Choosing the Step Size: Intuitive Line Search Algorithms with Efficient Convergence

Sara Fridovich-Keil SFK@EECS.BERKELEY.EDU
Benjamin Recht BRECHT@EECS.BERKELEY.EDU
University of California, Berkeley, CA 94720

Abstract

Iterative optimization algorithms generally consist of two components: a step direction and a step size. In this work, we focus on zeroth and first order methods for choosing the step size along either the negative gradient or a negative stochastic gradient. Line search methods are well-studied, with one of the most common being backtracking line search. Backtracking line search starts with an initial step size that is intended to be too large (greater than the inverse Lipschitz constant of the gradient), and then iteratively shrinks the step size until the Armijo condition for sufficient function decrease is satisfied. We propose two algorithms that match or improve upon the performance of backtracking line search in both theory (convergence rate bounds for deterministic gradient descent) and empirical performance (for both deterministic and stochastic gradient descent), depending on the initial step size, while providing an intuitive interpretation of each algorithm.

1. Introduction

Backtracking line search (as it is typically presented, e.g. in [17], [15], and [14]) begins with a user-specified initial step size, and adaptively *decreases* that step size until the Armijo condition [2]

$$f(x_k - t_k\nabla f(x_k)) \leq f(x_k) - \frac{t_k}{2}\|\nabla f(x_k)\|^2 \tag{1}$$

is satisfied in order to guarantee convergence. This procedure is theoretically well-motivated in the context of deterministic gradient descent with a sufficiently large initial step size, for use on convex objective functions with Lipschitz gradients. Each of these classical assumptions, however, is often violated in practice. Much prior work has managed to remove some of these assumptions in certain contexts; in this paper we continue their efforts.

1.1. Stochastic gradients

In practice, many optimization objectives include a sum over examples in a dataset of a per-example loss function, such as (optionally regularized) least squares and logistic regression. On large datasets, such objectives are typically optimized using stochastic gradient descent [18], in which the gradient at every iteration is computed on a subset (minibatch) of the dataset. Because the gradient and function evaluations in this minibatch setting are stochastic approximations of the objective, it is a priori unclear whether a line search method would be beneficial. Standard practice in stochastic gradient descent is to choose a fixed step size (often referred to as learning rate), a decreasing step size, or a hand-tuned schedule of step sizes in each iteration [4].

Many methods have been proposed for automatically adjusting the step size during the course of stochastic gradient descent, e.g. [19], [1], and [9]. One class of approaches combines a classical line search with a variance reduction method, typically by adaptively adjusting the minibatch size in each iteration to ensure sufficiently accurate stochastic gradients [16] [6] [10]. It is also possible to guarantee efficient convergence without increasing the minibatch size, e.g. using the stochastic average gradient method of [20] with a fixed step size or using vanilla stochastic gradients with an adaptive step size (changed at most once per iteration), as in [5]. In the case of interpolating classifiers, [21] prove that Armijo-based backtracking line search with a periodic step size reset converges at the same rate in stochastic (minibatch) and deterministic (full) gradient settings.

We build upon this line of work, particularly that of Berahas, Cao, Choromanski, and Scheinberg [5], by introducing two line search methods that are empirically competitive in the stochastic setting with the line search method whose convergence they guarantee.

1.2. Non-Lipschitz gradients and non-convexity

The Armijo condition is automatically satisfied by gradient descent on a convex objective with L-Lipschitz gradients and a step size no more than $\frac{1}{L}$. However, in practice some objective functions violate these conditions and yet have unique minima (e.g. $f(x) = \sqrt{|x|}$ for a one-dimensional example). Other objectives may satisfy these conditions but have a wide, flat minimum (e.g. $f(x) = x^{100}$ for a one-dimensional example) that is very slow to minimize using an Armijo line search. We introduce a line search method (Algorithm 2) that is independent of the Armijo condition, choosing a step size based on function evaluations only. This property additionally makes Algorithm 2 suitable for adaptation to derivative-free optimization, e.g. as in [11].

1.3. Initial step size too small

Although the traditional version of backtracking line search (e.g. in [17]) only decreases the step size, variants that increase and decrease the step size are also common: see e.g. [5], [16] [21], [22], [7], [13], and [12]. However, these methods either increase the step size by at most a constant factor in each iteration or require additional gradient evaluations to select the step size. We refer to these two versions of backtracking line search as traditional backtracking line search (Algorithm 3, never increases the step size) and adaptive backtracking line search (Algorithm 4, increases the initial step size by a constant factor in each iteration). Both algorithms we introduce can increase or decrease the step size as much as necessary in each iteration, with no additional gradient evaluations.

1.4. Contribution

Prior work has shown that line search methods can dramatically improve the performance of both deterministic and stochastic gradient descent. The latter is a highly nontrivial result, as we would not a priori expect that putting more effort into choosing a step size along a randomized direction would be beneficial. The contribution of this work is to speed up the line search component of these algorithms for both settings, by increasing the step size more consistently than prior methods. We also introduce a line search method similar to golden-section search [3] that does not rely on the Armijo condition, and is even more broadly applicable. We prove convergence rates in terms of both number of steps (gradient evaluations) and number of function evaluations per step in the convex and strongly convex settings under deterministic gradient descent, and present empirical results in the deterministic and stochastic (minibatch) settings.

2. Proposed algorithms

To guarantee efficient convergence for gradient descent, traditional backtracking line search assumes that the initial step size is larger than the inverse Lipschitz constant of the gradient [14]. In practice, this quantity is unknown, and estimating it is one of the purposes of line search algorithms. We propose a natural extension, forward-tracking line search (Algorithm 1), which iteratively grows or shrinks the initial step size as necessary to more tightly satisfy the Armijo condition and adaptively adjusts the initial step size across gradient steps to minimize unnecessary function evaluations. This procedure is designed to use the largest step size that the Armijo condition can guarantee converges, reducing the number of (stochastic) gradient steps, while keeping the number of function evaluations in each step small.

Backtracking line search and Algorithm 1, as well as [5] and most line search methods that avoid extra gradient evaluations, rely on the Armijo condition to choose a step size that guarantees sufficient function decrease. Although this approach tends to work well, it is specifically designed for convex objective functions with Lipschitz gradients. Accordingly, we propose approximately exact line search (Algorithm 2), which extends golden section search [3] to efficiently find a step size within a constant factor of the exact line search minimizer of *any* unimodal objective, using only function evaluations. The idea of Algorithm 2 is to increase the step size until the line search objective becomes nondecreasing, then decrease the step size to use the largest step size that does not exceed the exact line search minimizer.

Algorithm 1: Forward-Tracking Line Search

```
Input: $f, x_0, K, T_0 > 0, \beta \in (0,1)$
$f_{old} \leftarrow f(x_0)$
$t \leftarrow T_0$
for $k = 0,\ k < K,\ k = k+1$ do
    $t \leftarrow t/\beta$
    $f_{new} \leftarrow f(x_k - t\nabla f(x_k))$
    while $f_{new} < f_{old} - \frac{1}{2}t\|\nabla f(x_k)\|^2$ do
        $t \leftarrow t/\beta$
        $f_{new} \leftarrow f(x_k - t\nabla f(x_k))$
    end
    while $f_{new} > f_{old} - \frac{1}{2}t\|\nabla f(x_k)\|^2$ do
        $t \leftarrow \beta t$
        $f_{new} \leftarrow f(x_k - t\nabla f(x_k))$
    end
    $x_{k+1} \leftarrow x_k - t\nabla f(x_k)$
    $f_{old} \leftarrow f_{new}$
end
return $x_K$
```

Algorithm 2: Approximately Exact Line Search

```
Input: $f, x_0, K, T_0 > 0, \beta \in (0,1)$
$f_{old} \leftarrow f(x_0)$
$t \leftarrow T_0$
for $k = 0,\ k < K,\ k = k+1$ do
    $t \leftarrow t/\beta$
    $f_{new} \leftarrow f(x_k - t\nabla f(x_k))$
    while $f_{new} < f_{old}$ do
        $t \leftarrow t/\beta$
        $f_{old} \leftarrow f_{new}$
        $f_{new} \leftarrow f(x_k - t\nabla f(x_k))$
    end
    $t \leftarrow \beta t$
    $f_{old} \leftarrow f_{new}$
    $f_{new} \leftarrow f(x_k - t\nabla f(x_k))$
    while $f_{new} \leq f_{old}$ do
        $t \leftarrow \beta t$
        $f_{old} \leftarrow f_{new}$
        $f_{new} \leftarrow f(x_k - t\nabla f(x_k))$
    end
    $x_{k+1} \leftarrow x_k - t\nabla f(x_k)$
    $f_{old} \leftarrow f_{new}$
end
return $x_K$
```

3. Theoretical results

Proofs can be found in Section 5.

3.1. Forward-tracking line search

Invariant 1 *Let t be the step size used in a step of Algorithm 1. Then $t \in [\beta t_a, t_a]$, where t_a is the step size that satisfies the Armijo condition with equality ($f(x - t_a \nabla f(x)) = f(x) - \frac{t_a}{2} \|\nabla f(x)\|^2$).*

Theorem 2 *The iterates of Algorithm 1 on a weakly convex objective f with L-Lipschitz gradients satisfy*

$$\|\nabla f(x_k)\|^2 \leq \frac{2L}{\beta}(f(x_k) - f(x_{k+1}))$$

Corollary 3 *On weakly convex objectives with L-Lipschitz gradients, Algorithm 1 achieves optimality gap ϵ within*

$$\frac{L\|x_0 - x_*\|^2}{2\epsilon\beta}$$

steps (gradient evaluations).

This bound is an improvement compared to the corresponding bound of $\frac{L\|x_0 - x_*\|^2}{2\epsilon \min(LT_0, \beta)}$ steps for traditional backtracking line search (Theorem 20) or $\frac{L\|x_0 - x_*\|^2}{2\epsilon\beta} + \log_\beta(T_0 L)_+$ steps for adaptive backtracking line search (Theorem 22).

Theorem 4 *On m-strongly convex objectives with L-Lipschitz gradients, Algorithm 1 achieves optimality gap ϵ within*

$$\frac{L}{m\beta} \log\left(\frac{L\|x_0 - x_*\|^2}{\epsilon}\right)$$

steps, using no more than

$$k\left(2 + \log_\beta\left(\frac{m}{L}\right)\right) + \log_\beta\left(\min\left(\frac{\beta}{mT_0}, LT_0\right)\right)$$

function evaluations for k steps.

The corresponding bounds for traditional and adaptive backtracking line search are stated in Theorem 21 and Theorem 23, respectively.

3.2. Approximately exact line search

Invariant 5 *Let t be the step size used in a step of Algorithm 2. Then $t \in [\beta^2 t_*, t_*]$, where t_* is the step size used in exact line search.*

Theorem 6 *Let t_* be the step size used by exact line search starting at x_k. Then the iterates of Algorithm 2 on a weakly convex objective f with L-Lipschitz gradients satisfy*

$$\|\nabla f(x_k)\|^2 \leq \frac{2L}{\min(\beta^2 L t_*, 1)}(f(x_k) - f(x_{k+1}))$$

Theorem 7 *On m-strongly convex objectives with L-Lipschitz gradients, Algorithm 2 achieves optimality gap ϵ within*

$$\frac{1}{\beta^2\left(1-\sqrt{1-\frac{m}{L}}\right)}\log\left(\frac{L\|x_0-x_*\|^2}{\epsilon}\right)$$

steps, using no more than

$$k\left(5+\log_\beta\left(\frac{1-\sqrt{1-m/L}}{1+\sqrt{1-m/L}}\right)\right)+\log_\beta\left(\min\left(\frac{mT_0}{\beta(1-\sqrt{1-m/L})},\frac{\beta(1+\sqrt{1-m/L})}{mT_0}\right)\right)$$

function evaluations for k steps.

4. Empirical results

Figure 1 shows the empirical performance of Algorithms 1 (forward-tracking) and 2 (approximately exact) as compared to traditional and adaptive backtracking line search and the line search method of Berahas et al. [5] on a logistic regression objective (with $\lambda = \frac{1}{n}$):

$$f(w) = \frac{\lambda}{2}w^Tw + \frac{1}{n}\sum_{i=1}^{n}\log(1+\exp(-y_i(w^Tx_i)))$$

for the quantum dataset provided by [8], which has 50,000 examples x_i, each with 78 features.

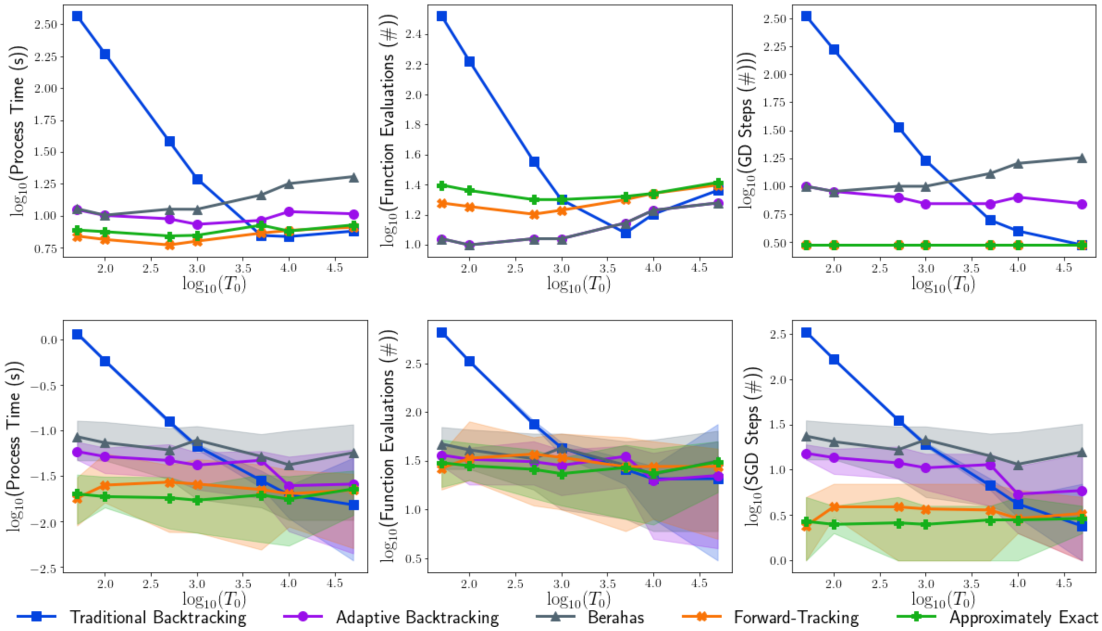

Figure 1: Performance on the quantum dataset to a relative error of 0.1%, as a function of user-specified initial step size T_0. The top row shows deterministic gradient descent, and the bottom row shows stochastic gradient descent with minibatches of 100 examples each. Lines indicate average performance, with error bars showing variation over 10 trials.

References

[1] Luís B. Almeida, Thibault Langlois, Jose D. Amaral, and Alexander Plakhov. *Parameter Adaptation in Stochastic Optimization*, pages 111–134. Publications of the Newton Institute. Cambridge University Press, 1999. doi: 10.1017/CBO9780511569920.007.

[2] Larry Armijo. Minimization of functions having lipschitz continuous first partial derivatives. *Pacific Journal of Mathematics*, 16(1):1–3, 1966.

[3] Mordecai Avriel and Douglass J Wilde. Golden block search for the maximum of unimodal functions. *Management Science*, 14(5):307–319, 1968.

[4] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. *Neural Networks: Tricks of the Trade*, pages 437–478, 2012. ISSN 1611-3349. doi: 10.1007/978-3-642-35289-8_26. URL http://dx.doi.org/10.1007/978-3-642-35289-8_26.

[5] Albert S Berahas, Liyuan Cao, Krzysztof Choromanski, and Katya Scheinberg. A theoretical and empirical comparison of gradient approximations in derivative-free optimization. *arXiv preprint arXiv:1905.01332*, 2019.

[6] Raghu Bollapragada, Richard Byrd, and Jorge Nocedal. Adaptive sampling strategies for stochastic optimization. *SIAM Journal on Optimization*, 28(4):3312–3343, Jan 2018. ISSN 1095-7189. doi: 10.1137/17m1154679. URL http://dx.doi.org/10.1137/17M1154679.

[7] Richard P. Brent. An algorithm with guaranteed convergence for finding a zero of a function. *The Computer Journal*, 14(4):422–425, 1971.

[8] Rich Caruana, Thorsten Joachims, and Lars Backstrom. Kdd-cup 2004: Results and analysis. *SIGKDD Explor. Newsl.*, 6(2):95–108, December 2004. ISSN 1931-0145. doi: 10.1145/1046456.1046470. URL http://doi.acm.org/10.1145/1046456.1046470.

[9] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159, July 2011. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1953048.2021068.

[10] Michael P. Friedlander and Mark Schmidt. Hybrid deterministic-stochastic methods for data fitting. *SIAM Journal on Scientific Computing*, 34(3):A1380A1405, Jan 2012. ISSN 1095-7197. doi: 10.1137/110830629. URL http://dx.doi.org/10.1137/110830629.

[11] Kevin G Jamieson, Robert Nowak, and Ben Recht. Query complexity of derivative-free optimization. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2012.

[12] Adrian S Lewis and Michael L Overton. Nonsmooth optimization via quasi-newton methods. *Mathematical Programming*, 141(1–2):135–163, 2013.

[13] John A Nelder and Roger Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, 1965.

[14] Yu Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

[15] Jorge Nocedal and Stephen J Wright. Line search methods. *Numerical Optimization*, pages 30–65, 2006.

[16] Courtney Paquette and Katya Scheinberg. A stochastic line search method with convergence rate analysis. *arXiv preprint arXiv:1807.07994*, 2018.

[17] Benjamin Recht and Stephen J. Wright. *Optimization for Modern Data Analysis*. 2019. Preprint available at http://eecs.berkeley.edu/~brecht/opt4mlbook.

[18] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951.

[19] Tom Schaul, Sixin Zhang, and Yann LeCun. No more pesky learning rates. *arXiv preprint arXiv:1206.1106*, 2012.

[20] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1–2):83–112, 2017.

[21] Sharan Vaswani, Aaron Mishkin, Issam Laradji, Mark Schmidt, Gauthier Gidel, and Simon Lacoste-Julien. Painless stochastic gradient: Interpolation, line-search, and convergence rates. *arXiv preprint arXiv:1905.09997*, 2019.

[22] Philip Wolfe. Convergence conditions for ascent methods. *SIAM review*, 11(2):226–235, 1969.

5. Appendix

In this section, we prove the results in Section 3.

5.1. Preliminaries

In this section, we introduce useful notation, definitions, and lemmas. We begin by defining notation:

- $\|\cdot\| \equiv \|\cdot\|_2$, the Euclidean norm
- $(\cdot)_+ \equiv \max(\cdot, 0)$

Definition 8 *A function f has L-Lipschitz gradient if:*

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$$

Definition 9 *A function f is (weakly) convex if:*

$$f(y) \geq f(x) + \nabla f(x)^T (y - x)$$

Definition 10 *A function f is m-strongly convex if:*

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) + \frac{m}{2}\|y - x\|^2$$

Definition 11 *Armijo condition [2]:*

$$f(x_k - t\nabla f(x_k)) \leq f(x_k) - \gamma t\|\nabla f(x_k)\|^2$$

where $\gamma \in (0,1)$, and typically $\gamma = \frac{1}{2}$ (we use $\gamma = \frac{1}{2}$ throughout).

Lemma 12 *For any step size t and any (not necessarily convex) function f with L-Lipschitz gradient, where $x_{k+1} = x_k - t\nabla f(x_k)$,*

$$f(x_{k+1}) \leq f(x_k) - t\left(1 - \frac{tL}{2}\right)\|\nabla f(x_k)\|^2$$

Proof *From Taylor's theorem (also using Cauchy-Schwarz and Lipschitz gradients),*

$$\begin{aligned} f(y) - f(x) - \nabla f(x)^T(y-x) &= \int_0^1 (\nabla f(ty + (1-t)x) - \nabla f(x))^T(y-x)dt \\ &\leq \|y-x\|\int_0^1 \|\nabla f(ty+(1-t)x) - \nabla f(x)\|dt \\ &\leq \|y-x\|\int_0^1 Lt\|y-x\|dt \\ &= \frac{L}{2}\|y-x\|^2 \end{aligned}$$

Therefore, $f(y) - f(x) - \nabla f(x)^T(y-x) \leq \frac{L}{2}\|y-x\|^2$. Taking $y = x_{k+1}$ and $x = x_k$:

$$\begin{aligned} & f(x_{k+1}) - f(x_k) - \nabla f(x_k)^T(x_{k+1} - x_k) \leq \frac{L}{2}\|x_{k+1} - x_k\|^2 \\ & \Rightarrow f(x_{k+1}) \leq f(x_k) + \nabla f(x_k)^T(-t\nabla f(x_k)) + \frac{L}{2}\|-t\nabla f(x_k)\|^2 \\ & \Rightarrow f(x_{k+1}) \leq f(x_k) - t\|\nabla f(x_k)\|^2 + \frac{Lt^2}{2}\|\nabla f(x_k)\|^2 \end{aligned}$$

■

Lemma 13 *Let t_i be the step size used in step i. If t_i satisfies the Armijo condition, then*

$$f(x_k) - f_* \leq \frac{\|x_0 - x_*\|^2}{2kt_{min}}$$

where $t_{min} \leq t_i\ \forall i = 1, 2, ...$

Proof *(based on http://seas.ucla.edu/~vandenbe/ee236c.html)*

$$\begin{aligned} f(x_{i+1}) &\leq f(x_i) - \frac{t_i}{2}\|\nabla f(x_i)\|^2 \text{ (Armijo condition)} \\ &\leq f_* + \nabla f(x_i)^T(x_i - x_*) - \frac{t_i}{2}\|\nabla f(x_i)\|^2 \text{ (convexity of } f\text{)} \\ &= f_* + \frac{1}{2t_i}(\|x_i - x_*\|^2 - \|x_{i+1} - x_*\|^2) \end{aligned}$$

This implies that $\|x_i - x_*\| \geq \|x_{i+1} - x_*\|$, so we can replace t_i with $t_{min} \leq t_i$ in our subsequent analysis:

$$\begin{aligned} f(x_k) - f_* &\leq \frac{1}{k}\sum_{i=1}^{k}(f(x_i) - f_*) \\ &\leq \frac{1}{k}\sum_{i=1}^{k}\frac{1}{2t_{i-1}}(\|x_{i-1} - x_*\|^2 - \|x_i - x_*\|^2) \\ &\leq \frac{1}{2kt_{min}}(\|x_0 - x_*\|^2 - \|x_k - x_*\|^2) \\ &\leq \frac{1}{2kt_{min}}\|x_0 - x_*\|^2 \end{aligned}$$

■

Lemma 13 allows us to directly translate a per-iteration function decrease of sufficient magnitude (compared to the norm of the gradient, based on the Armijo condition) into a $\frac{1}{k}$ convergence rate.

Corollary 14 *In order to achieve an objective value $f(x_k)$ that is within ϵ of the minimum f_*, given that t_i satisfies the Armijo condition for all steps i, the number of steps $k \leq \frac{\|x_0 - x_*\|^2}{2\epsilon t_{min}}$ is sufficient.*

Lemma 15 *If f is m-strongly convex, then*

$$f(x) - f_* \leq \frac{1}{2m}\|\nabla f(x)\|^2$$

Proof *(based on [17], chapter 3)*

$$\begin{aligned} f(y) &\geq f(x) + \nabla f(x)^T(y - x) + \frac{m}{2}\|y - x\|^2 \text{ \textit{(by strong convexity)}} \\ \min_y f(y) &\geq \min_y f(x) + \nabla f(x)^T(y - x) + \frac{m}{2}\|y - x\|^2 \\ f_* &\geq f(x) - \frac{1}{2m}\|\nabla f(x)\|^2 \text{ \textit{(}}y_* = x_* \text{ \textit{on the left and} } y_* = x - \frac{1}{m}\nabla f(x) \text{ \textit{on the right)}} \end{aligned}$$

■

Lemma 16 *Let f be m-strongly convex and have L-Lipschitz gradient, and let t_i be the step size used in step i. If t_i satisfies the Armijo condition and $t_i \leq \frac{1}{m}$ $\forall i$, then*

$$f(x_k) - f_* \leq L(1 - mt_{min})^k\|x_0 - x_*\|^2$$

Proof *(based on [17], chapter 3)*

$$\begin{aligned} f(x_{i+1}) - f_* &= f(x_i - t_i\nabla f(x_i)) - f_* \\ &\leq f(x_i) - f_* - \frac{t_i}{2}\|\nabla f(x_i)\|^2 \text{ \textit{(since} } t_i \text{ \textit{satisfies Armijo)}} \\ &\leq f(x_i) - f_* - mt_i(f(x_i) - f_*) \text{ \textit{(Lemma 15)}} \\ &\leq (1 - mt_{min})(f(x_i) - f_*) \end{aligned}$$

Applying this relation iteratively yields:

$$\begin{aligned} f(x_k) - f_* &\leq (1 - mt_{min})^k (f(x_0) - f_*) \\ &\leq L(1 - mt_{min})^k \|x_0 - x_*\|^2 \end{aligned}$$

where the last step follows from convexity and Lipschitz gradient. ■

Corollary 17 *In order to achieve an objective value $f(x_k)$ that is within ϵ of the minimum f_*, given that t_i satisfies the Armijo condition and $t_i \leq \frac{1}{m}$ $\forall i$, the number of steps $k \leq \frac{1}{mt_{min}} \log \left(\frac{L\|x_0 - x_*\|^2}{\epsilon} \right)$ is sufficient.*

Lemma 18 *If f is m-strongly convex with L-Lipschitz gradient, then*

$$\frac{1}{L} \leq t_a \leq \frac{1}{m}$$

where t_a solves the Armijo condition (Definition 11) with equality. Note that the first inequality requires only the Lipschitz gradient condition, and the second requires only the strong convexity condition.

Proof *We begin by proving the first inequality, which assumes that the objective has L-Lipschitz gradient. Using Lemma 12, we have:*

$$f(x_k - t_a \nabla f(x_k)) \leq f(x_k) - t_a \left(1 - \frac{t_a L}{2}\right) \|\nabla f(x_k)\|^2$$

Since t_a solves the Armijo condition with equality, we have:

$$f(x_k - t_a \nabla f(x_k)) = f(x_k) - \frac{t_a}{2} \|\nabla f(x_k)\|^2$$

Relating these, we have:

$$\begin{aligned} &\Rightarrow f(x_k) - \frac{t_a}{2} \|\nabla f(x_k)\|^2 \leq f(x_k) - t_a \left(1 - \frac{t_a L}{2}\right) \|\nabla f(x_k)\|^2 \\ &\Rightarrow t_a \geq \frac{1}{L} \end{aligned}$$

We proceed to prove the second inequality, which assumes that the objective is m-strongly convex. By strong convexity, we have:

$$f(x_k - t_a \nabla f(x_k)) \geq f(x_k) - t_a \left(1 - \frac{t_a m}{2}\right) \|\nabla f(x_k)\|^2$$

Since t_a solves the Armijo condition with equality, we have:

$$f(x_k - t_a \nabla f(x_k)) = f(x_k) - \frac{t_a}{2} \|\nabla f(x_k)\|^2$$

Relating these, we have:

$$\Rightarrow f(x_k) - \frac{t_a}{2}\|\nabla f(x_k)\|^2 \geq f(x_k) - t_a\left(1 - \frac{t_a m}{2}\right)\|\nabla f(x_k)\|^2$$
$$\Rightarrow t_a \leq \frac{1}{m}$$

■

Lemma 19 *If f is m-strongly convex with L-Lipschitz gradient, then*

$$\frac{1}{m}\left[1 - \sqrt{1 - \frac{m}{L}}\right] \leq t_* \leq \frac{1}{m}\left[1 + \sqrt{1 - \frac{m}{L}}\right]$$

where $t_* = \operatorname{argmin}_t f(x - t\nabla f(x))$, the step size used in exact line search starting from any point x.

Proof

$$\begin{aligned} f(x - t_*\nabla f(x)) &\leq f(x - \frac{1}{L}\nabla f(x)) \\ &\leq f(x) - \frac{1}{2L}\|\nabla f(x)\|^2 \end{aligned}$$

From strong convexity (Definition 10), we have:

$$\begin{aligned} f(x - t_*\nabla f(x)) &\geq f(x) - \nabla f(x)^T(t_*\nabla f(x)) + \frac{m}{2}\|t_*\nabla f(x)\|^2 \\ &= f(x) + \left(\frac{t_*^2 m}{2} - t_*\right)\|\nabla f(x)\|^2 \end{aligned}$$

Combining these:

$$f(x) + \left(\frac{t_*^2 m}{2} - t_*\right)\|\nabla f(x)\|^2 \leq f(x) - \frac{1}{2L}\|\nabla f(x)\|^2$$
$$\Rightarrow t_*^2 m - 2t_* + \frac{1}{L} \leq 0$$

which is satisfied for t_* in the interval $\left[\frac{1}{m} - \sqrt{\frac{1}{m^2} - \frac{1}{mL}}, \frac{1}{m} + \sqrt{\frac{1}{m^2} - \frac{1}{mL}}\right]$. ■

5.2. Traditional backtracking line search

Algorithm 3: Traditional Backtracking Line Search

```
Input: $f, x_0, K, T_0 > 0, \beta \in (0,1)$
$f_{old} \leftarrow f(x_0)$
for $k = 0$, $k < K$, $k = k + 1$ do
    $t \leftarrow T_0$
    $f_{new} \leftarrow f(x_k - t\nabla f(x_k))$
    while $f_{new} > f_{old} - \frac{1}{2}t\|\nabla f(x_k)\|^2$ do
        $t \leftarrow \beta t$
        $f_{new} \leftarrow f(x_k - t\nabla f(x_k))$
    end
    $x_{k+1} \leftarrow x_k - t\nabla f(x_k)$
    $f_{old} \leftarrow f_{new}$
end
return $x_K$
```

Theorem 20 *On weakly convex objectives with L-Lipschitz gradients, Algorithm 3 achieves optimality gap ϵ within*

$$\frac{L\|x_0 - x_*\|^2}{2\epsilon \min(LT_0, \beta)}$$

steps, using no more than

$$1 + k\left(1 + \log_\beta\left(\frac{1}{LT_0}\right)_+\right)$$

function evaluations for k steps.

Proof *We begin by proving the first statement, which bounds the number of steps. By construction, the step size t_k satisfies the Armijo condition. We can therefore apply Lemma 13 by calculating t_{min}. If no backtracking was required in step k, then $t_k = T_0$. If backtracking was required, then $t_k < T_0$ and $\beta^{-1}t_k$ satisfies the opposite of the Armijo condition:*

$$f(x_k - \frac{t_k}{\beta}\nabla f(x_k)) > f(x_k) - \frac{t_k}{2\beta}\|\nabla f(x_k)\|^2$$

Using Lemma 12, we have:

$$\begin{aligned}
&f(x_k - \frac{t_k}{\beta}\nabla f(x_k)) \le f(x_k) - \frac{t_k}{\beta}\left(1 - \frac{t_k L}{2\beta}\right)\|\nabla f(x_k)\|^2 \\
&\Rightarrow f(x_k) - \frac{t_k}{2\beta}\|\nabla f(x_k)\|^2 \le f(x_k) - \frac{t_k}{\beta}\left(1 - \frac{t_k L}{2\beta}\right)\|\nabla f(x_k)\|^2 \\
&\Rightarrow t_k \ge \frac{\beta}{L}
\end{aligned}$$

Therefore, $t_k \ge \min(T_0, \frac{\beta}{L})$, so we can apply Lemma 13 with $t_{min} = \min(T_0, \frac{\beta}{L})$. From Corollary 14, we can conclude that Backtracking Line Search requires no more than

$$\frac{\|x_0 - x_*\|^2}{2\epsilon \min(T_0, \frac{\beta}{L})} \tag{2}$$

steps to achieve an objective value within ϵ of the global minimum.

Next, we prove the second statement, which bounds the number of function evaluations for k steps. In a single step, Algorithm 3 makes one initial function evaluation (to compute $f(x_k - T_0\nabla f(x_k))$), then one additional function evaluation per iteration of the While loop. Let t_k be the step size found by Algorithm 3, t_a be the step size that satisfies Definition 11 with equality in step k, and q be the number of iterations of the While loop. Then we can write $t_k = T_0\beta^q \leq t_a$.

Consider two cases:

1. *$T_0 > t_a$, in which case$q \leq \log_\beta(\frac{t_a}{T_0})$*
2. *$T_0 \leq t_a$, so $q = 0$*

Combining these cases, the total number of function evaluations in a single step k of Algorithm 3 is

$$n_k = q + 1 \leq 1 + \log_\beta \left(\frac{t_a}{T_0}\right)_+ \leq 1 + \log_\beta \left(\frac{1}{T_0 L}\right)_+$$

where the final inequality follows from Lemma 18.

We include $n_0 = 1$ to count the initial evaluation of $f(x_0)$. The total number of function evaluations required for k steps is therefore upper bounded by

$$1 + k\left(1 + \log_\beta \left(\frac{1}{T_0 L}\right)_+\right) \tag{3}$$

■

Theorem 21 *On m-strongly convex objectives with L-Lipschitz gradients, Algorithm 3 achieves optimality gap ϵ within*

$$\frac{L}{m\min(LT_0, \beta)} \log\left(\frac{L\|x_0 - x_*\|^2}{\epsilon}\right)$$

steps. The number of function evaluations for k steps is bounded in Theorem 20.

Proof *Since $t_k \leq t_a \leq \frac{1}{m}$ (by Lemma 18), we can apply Lemma 16 with $t_{min} = \min(T_0, \frac{\beta}{L})$ (using the reasoning from Theorem 20). The bound then follows from Corollary 17.* ■

5.3. Adaptive backtracking line search

Algorithm 4: Adaptive Backtracking Line Search

```
Input: $f, x_0, K, T_0 > 0, \beta \in (0, 1)$
$f_{old} \leftarrow f(x_0)$
$t \leftarrow T_0$
for $k = 0$, $k < K$, $k = k + 1$ do
    $t \leftarrow t/\beta$
    $f_{new} \leftarrow f(x_k - t\nabla f(x_k))$
    while $f_{new} > f_{old} - \frac{1}{2}t\|\nabla f(x_k)\|^2$ do
        $t \leftarrow \beta t$
        $f_{new} \leftarrow f(x_k - t\nabla f(x_k))$
    end
    $x_{k+1} \leftarrow x_k - t\nabla f(x_k)$
    $f_{old} \leftarrow f_{new}$
end
return $x_K$
```

Theorem 22 *On weakly convex objectives with L-Lipschitz gradients, Algorithm 4 achieves optimality gap ϵ within*

$$\frac{L\|x_0 - x_*\|^2}{2\epsilon\beta} + \log_\beta(T_0 L)_+$$

steps, using no more than

$$2(k+1) + \max\left(\log_\beta\left(\frac{1}{LT_0}\right), -(k+1)\right)$$

function evaluations for k steps.

Proof *We begin by proving the first statement, which bounds the number of steps. Let T_k denote the initial step size at step k, and t_k denote the step size used at step k. By construction, $T_{k+1} = \frac{t_k}{\beta}$ and $T_1 = \frac{T_0}{\beta}$.*

In a given step k, one of the following is true:

1. *Backtracking occurred at some prior step $i < k$, so $T_i \geq \frac{1}{L}$. In this case, we know that $t_{k-1} \geq \frac{\beta}{L}$, so $T_k \geq \frac{1}{L}$*
2. *No backtracking has yet occurred, so $t_{k-1} \leq \frac{1}{L}$. In this case, $T_k = \frac{T_0}{\beta^k}$*

Combining these cases, we have that $T_k \geq \min(\frac{1}{L}, \frac{T_0}{\beta^k})$.

By construction, the step size t_k satisfies the Armijo condition. We can therefore apply Lemma 13 by calculating t_{min}. If no backtracking was required in step k, then $t_k = T_k$. If backtracking was required, then $t_k < T_k$. By construction, we have that $\beta^{-1}t_k$ satisfies the opposite of the Armijo condition:

$$f(x_k - \frac{t_k}{\beta}\nabla f(x_k)) > f(x_k) - \frac{t_k}{2\beta}\|\nabla f(x_k)\|^2$$

Using Lemma 12, we have:

$$\begin{aligned}
&f(x_k - \frac{t_k}{\beta}\nabla f(x_k)) \leq f(x_k) - \frac{t_k}{\beta}\left(1 - \frac{t_k L}{2\beta}\right)\|\nabla f(x_k)\|^2 \\
&\Rightarrow f(x_k) - \frac{t_k}{2\beta}\|\nabla f(x_k)\|^2 \leq f(x_k) - \frac{t_k}{\beta}\left(1 - \frac{t_k L}{2\beta}\right)\|\nabla f(x_k)\|^2 \\
&\Rightarrow t_k \geq \frac{\beta}{L}
\end{aligned}$$

Therefore, $t_k \geq \min(T_k, \frac{\beta}{L}) \geq \min(\frac{T_0}{\beta^k}, \frac{\beta}{L})$. From this bound, we can see that there are two potential phases of Algorithm 4: one in which the step size monotonically increases, and another in which the step size is always $\geq \frac{\beta}{L}$. The first phase requires no more than $\log_\beta(T_0 L)_+$ steps, and the second requires no more than $\frac{L\|x_0 - x_*\|^2}{2\epsilon\beta}$ (using Corollary 14).

We now proceed to bound the number of function evaluations required in the first k steps. Let $n_k \geq 1$ be the number of function evaluations in step k and N_k be the total number of function evaluations up to and including step k.

We begin by noting that $T_{k+1} = \frac{t_k}{\beta}$. We have that $t_k = T_k\beta^{n_k - 1}$, since each iteration of the While loop multiplies by β (and one additional function evaluation is needed before the loop). Also note that one function evaluation is needed before the first step is taken; we consider this as $n_0 = 1$, which is consistent with $t_0 = T_0$. Combined, we have:

$$\begin{aligned}
T_{k+1} &= \frac{1}{\beta}T_k\beta^{n_k - 1} \\
&= T_k\beta^{n_k - 2} \\
&\Rightarrow n_k = 2 + \log_\beta\left(\frac{T_{k+1}}{T_k}\right) \\
&\Rightarrow N_k = \sum_{i=0}^{k} n_i = 2(k+1) + \log_\beta\left(\frac{T_{k+1}}{T_0}\right) \\
&\Rightarrow N_k \leq 2(k+1) + \log_\beta\left(\min\left(\frac{1}{LT_0}, \frac{1}{\beta^{k+1}}\right)\right)
\end{aligned}$$

$$\Rightarrow N_k \leq 2(k+1) + \max(\log_\beta\left(\frac{1}{LT_0}\right), -(k+1)) \tag{4}$$

To bound the number of function evaluations for a desired accuracy ϵ, we simply plug in our upper bound for the number of steps k to compute the corresponding bound for N_k. ■

Theorem 23 *On m-strongly convex objectives with L-Lipschitz gradients, Algorithm 4 achieves optimality gap ϵ within*

$$\frac{L}{m\beta}\log\left(\frac{L\|x_0 - x_*\|^2}{\epsilon}\right) + \log_\beta(T_0 L)_+$$

steps. The number of function evaluations for k steps is bounded in Theorem 22.

Proof *Since $t_k \leq t_a \leq \frac{1}{m}$ (by Lemma 18), we can apply Lemma 16 by calculating t_{min}. From the reasoning in Theorem 22, $t_k \geq \min(\frac{T_0}{\beta^k}, \frac{\beta}{L})$. From this bound, we can see that there are two potential phases of Algorithm 4: one in which the step size monotonically increases, and another in which the step size is always $\geq \frac{\beta}{L}$. The first phase requires no more than $\log_\beta(T_0 L)_+$ steps, and the second requires no more than $\frac{L}{m\beta}\log\left(\frac{L\|x_0 - x_*\|^2}{\epsilon}\right)$ steps (by Corollary 17).* ■

5.4. Forward-tracking line search

Proof [of Invariant 1] In each step, Algorithm 1 iteratively increases the step size by factors of β^{-1} until the Armijo condition is not satisfied and then decreases the step size by factors of β until the Armijo condition is satisfied. This guarantees that $t \leq t_a$ and $\beta^{-1}t > t_a$. ■

Proof [of Theorem 2] Consider a single gradient step k. Since backtracking was required, $\beta^{-1}t_k$ satisfies the opposite of the Armijo condition:

$$f(x_k - \frac{t_k}{\beta}\nabla f(x_k)) > f(x_k) - \frac{t_k}{2\beta}\|\nabla f(x_k)\|^2$$

Using Lemma 12, we have:

$$\begin{aligned}
&f(x_k - \frac{t_k}{\beta}\nabla f(x_k)) \leq f(x_k) - \frac{t_k}{\beta}(1 - \frac{t_k L}{2\beta})\|\nabla f(x_k)\|^2 \\
&\Rightarrow f(x_k) - \frac{t_k}{2\beta}\|\nabla f(x_k)\|^2 \leq f(x_k) - \frac{t_k}{\beta}(1 - \frac{t_k L}{2\beta})\|\nabla f(x_k)\|^2 \\
&\Rightarrow t_k \geq \frac{\beta}{L}
\end{aligned}$$

Since this t_k must satisfy the Armijo condition, we have:

$$\begin{aligned}
&f(x_k - t_k\nabla f(x_k)) \leq f(x_k) - \frac{t_k}{2}\|\nabla f(x_k)\|^2 \\
&\Rightarrow f(x_{k+1}) \leq f(x_k) - \frac{\beta}{2L}\|\nabla f(x_k)\|^2
\end{aligned}$$

$$\Rightarrow \|\nabla f(x_k)\|^2 \leq \frac{2L}{\beta}(f(x_k) - f(x_{k+1})) \tag{5}$$

■

Proof [of Corollary 3] Since t_k satisfies the Armijo condition, we can apply Lemma 13 with $t_{min} = \frac{\beta}{L}$. The result then follows from Corollary 14. ■

Proof [of Theorem 4] Let T_k denote the step size at the beginning of the kth step of Algorithm 1, and t_k denote the step size used in the kth step. Let t_a denote the step size (in step k) that would satisfy the Armijo condition with equality.

We begin by proving the first statement, which bounds the number of gradient steps to achieve a desired objective gap ϵ. Since $t_k \leq t_a \leq \frac{1}{m}$ (by Lemma 18), we can apply Lemma 16 with $t_{min} = \frac{\beta}{L}$. The bound then follows from Corollary 17.

We now proceed to prove the second statement, which bounds the number of function evaluations in the first k gradient steps. In a single step, Algorithm 1 makes one initial function evaluation (to compute $f(x_k - T_k \nabla f(x_k))$), then one additional function evaluation per iteration of each of the two While loops. Let p be the number of iterations of the first While loop and q be the number of iterations of the second While loop. Then we can write $t_k = T_k \beta^{q-p} \leq t_a$.

Consider two cases for a given step $k > 1$. In the first case, $T_k > t_a$, so $p = 0$. Then $q \leq \log_\beta(\frac{t_a}{T_k})$.

In the second case, $T_k \leq t_a$ and $T_k \beta^{1-p} \leq t_a$, so $p \leq 1 + \log_\beta(\frac{T_k}{t_a})$. Since we still have $T_k \beta^{q-p} \leq t_a$,

$$\begin{aligned} q &\leq \log_\beta\left(\frac{t_a \beta^p}{T_k}\right) \\ &\leq \log_\beta\left(\frac{t_a \beta^{1+\log_\beta(T_k/t_a)}}{T_k}\right) \\ &\leq \log_\beta\left(\frac{t_a \beta \frac{T_k}{t_a}}{T_k}\right) \\ &= 1 \end{aligned}$$

We can then combine these two cases to conclude that the total number of function evaluations in step $k > 1$ of Algorithm 1 is

$$n_k = p + q + 1 \leq \max\left(1 + \log_\beta\left(\frac{t_a}{T_k}\right), 2 + \log_\beta\left(\frac{T_k}{t_a}\right)\right)$$

For all $k \geq 1$:

$$\begin{aligned} T_{k+1} &= \beta^{-1} t_k \\ &\in [t_a, \beta^{-1} t_a] \quad \text{(Invariant 1)} \\ &\in \left[\frac{1}{L}, \frac{1}{\beta m}\right] \quad \text{(Lemma 18)} \end{aligned}$$

Using this range for T_k (for $k > 1$) and Lemma 18):

$$\begin{aligned} n_k &\leq \max\left(1 + \log_\beta\left(\frac{t_a}{T_k}\right), 2 + \log_\beta\left(\frac{T_k}{t_a}\right)\right) \\ &\leq \max\left(1 + \log_\beta\left(\frac{1}{LT_k}\right), 2 + \log_\beta(mT_k)\right) \\ &\leq \max\left(1 + \log_\beta\left(\frac{\beta m}{L}\right), 2 + \log_\beta\left(\frac{m}{L}\right)\right) \\ &= 2 + \log_\beta\left(\frac{m}{L}\right) \end{aligned}$$

Finally, the total number of function evaluations in the first k steps is:

$$\begin{aligned}
N_k &= \sum_{i=0}^{k} n_i \\
&\leq 1 + \max\left(1 + \log_\beta\left(\frac{t_a}{T_1}\right), 2 + \log_\beta\left(\frac{T_1}{t_a}\right)\right) + \sum_{i=2}^{k}\left(2 + \log_\beta\left(\frac{m}{L}\right)\right) \\
&\leq 1 + \max\left(2 + \log_\beta\left(\frac{1}{LT_0}\right), 1 + \log_\beta(mT_0)\right) + (k-1)\left(2 + \log_\beta\left(\frac{m}{L}\right)\right) \\
&= 2 + \max\left(\log_\beta\left(\frac{\beta}{LT_0}\right), \log_\beta(mT_0)\right) + (k-1)\left(2 + \log_\beta\left(\frac{m}{L}\right)\right) \\
&= k\left[2 + \log_\beta\left(\frac{m}{L}\right)\right] + \max\left(\log_\beta\left(\frac{\beta}{mT_0}\right), \log_\beta(LT_0)\right)
\end{aligned}$$

■

5.5. Approximately exact line search

Proof [of Invariant 5] We start with some notation and observations. With T_k as the initial step size of Algorithm 2 starting from x_k, let t_1 be the step size after the first While loop, $t_2 = \beta t_1$ be the step size immediately prior to the second While loop, and t_3 be the final step size upon exiting the second While loop. Let $g(s) = f(x_k - s\nabla f(x_x))$, the value of f evaluated along the direction of the negative gradient. Since f is convex, so is g. Let $t_* = \arg\min_t f(x_k - t\nabla f(x_k)) = \arg\min_s g(s)$, a step size used in exact line search starting from x_k. We can observe (using convexity of g and the definition of t_*) that $g(s)$ is nonincreasing for $s \in [0, t_*)$ and $g(s)$ is nondecreasing for $s \in (t_*, \infty)$. If t_* is not unique, then the present invariant may be taken as showing that $\exists$ some t_* that satisfies the relevant conditions.

We begin by showing that $t_1 \geq t_*$. Consider the first While loop. One of the following must be true:

1. $g(T_k) \geq g(0)$, in which case we do not enter the first While loop at all, so $t_1 = T_k$.

2. $g(t_1) \geq g(\beta t_1)$.

In either case, $g(s)$ is nondecreasing at $s = t_1$, so $t_1 \geq t_*$.

We now proceed to prove the present invariant, that $t_3 \in [\beta^2 t_*, t_*]$. Let p be the number of times the second While loop executes, so $t_3 = \beta^p t_2 = \beta^{p+1} t_1$. Since $t_1 \geq t_*$, $t_3 \geq \beta^{p+1} t_*$. One of the following must be true:

1. $p = 0$, in which case $g(\beta t_1) \geq g(t_1)$, so $g(s)$ is nonincreasing at $s = \beta t_1$. Therefore $t_* \geq \beta t_1 = t_3$. Since $t_1 \geq t_*$, we have that $\beta t_* \leq t_3 \leq t_*$, a stricter bound than required for the present invariant.

2. $p \geq 1$: From the conditions of the second While loop, which executes at least once, we have that $g(\beta^{p+1} t_1) \geq g(\beta^p t_1)$ and $g(\beta^k t_1) < g(\beta^{k-1} t_1)$ $\forall k = 1, 2, ..., p$. This implies that $g(s)$ is nonincreasing at $s = \beta^{p+1} t_1$ and nondecreasing at $s = \beta^{p-1} t_1$. Therefore, $\beta^{p+1} t_1 \leq t_* \leq \beta^{p-1} t_1$. Since $t_3 = \beta^{p+1} t_1$, we have that $t_3 \leq t_* \leq \beta^{-2} t_3$, proving the present invariant.

■

Proof [of Theorem 6] Let t_k be the step size used by Algorithm 2 in step k, t_a be the step size that would satisfy the Armijo condition with equality in step k, and t_* be the step size that would be used in an exact line search in step k. We begin with the observation that

$$f(x_k - t_k\nabla f(x_k)) \leq \max(f(x_k) - \frac{t_k}{2}\|\nabla f(x_k)\|^2, f(x_k) - \frac{t_a}{2}\|\nabla f(x_k)\|^2)$$

which holds because one of the following cases must be true:

1. $t_k > t_a$: In this case, by construction it must be that $f(x_k - t_k\nabla f(x_k)) \leq f(x_k - t_a\nabla f(x_k)) = f(x_k) - \frac{t_a}{2}\|\nabla f(x_k)\|^2$.
2. $t_k \leq t_a$: In this case, by definition t_k satisfies the Armijo condition: $f(x_k - t_k\nabla f(x_k)) \leq f(x_k) - \frac{t_k}{2}\|\nabla f(x_k)\|^2$.

Combining this observation with Invariant 5, we have that

$$f(x_k - t_k\nabla f(x_k)) \leq \max\left(f(x_k) - \frac{\beta^2 t_*}{2}\|\nabla f(x_k)\|^2, f(x_k) - \frac{t_a}{2}\|\nabla f(x_k)\|^2\right),$$

Therefore,

$$f(x_k) - f(x_{k+1}) \geq \min\left(\frac{\beta^2 t_*}{2}, \frac{t_a}{2}\right)\|\nabla f(x_k)\|^2$$

From here, we can apply Lemma 18:

$$f(x_k) - f(x_{k+1}) \geq \min\left(\frac{\beta^2 t_*}{2}, \frac{1}{2L}\right)\|\nabla f(x_k)\|^2$$

Rearranging this, we have:

$$\|\nabla f(x_k)\|^2 \leq \frac{2L}{\min(\beta^2 L t_*, 1)}(f(x_k) - f(x_{k+1}))$$

■

Proposition 24 *In each step of Algorithm 2, the first While loop terminates (finding t_1) after at most $\lceil\log_\beta(\frac{T_0}{t_*})\rceil_+ + 1$ iterations.*

Proof *$t_1 = \beta^{-k}T_0 \geq t_*$, where the first While loop executed k times. Rearranging, we can see that $\lceil\log_\beta(\frac{T_0}{t_*})\rceil_+$ satisfies the inequality (where the ceiling and + are required to handle the cases where $T_0 > t_*$ and $T_0 \leq t_*$). The $+1$ is necessary because in some cases $\beta^{-k+1}T_0 \geq t_*$, but in such cases we still have that $\beta^{-k+2}T_0 \leq t_*$.* ■

Proposition 25 *In each step of Algorithm 2, the second While loop executes at most $\lceil\log_\beta(\frac{t_*}{T_0})\rceil_+ + 1$ times.*

Proof *There are two cases:*

1. *The first While loop executed at least twice: In this case, the second While loop executes exactly once (satisfying the proposition).*
2. *The first While loop executed no more than once: In this case, we have that $\alpha T_0 > t_*$. The starting value for the second While loop is no more than T_0. Then the second While loop executes at most $\lceil \log_\beta(\frac{t_*}{T_0}) \rceil_+ + 1$ times (using the same reasoning as in Proposition 24).*

Combining these cases completes the proof. ■

Proof [of Theorem 7] We begin by proving the first statement, which bounds the number of steps.

$$\begin{aligned} \|\nabla f(x_k)\|^2 &\leq \frac{2L}{\min(\beta^2 L t_*, 1)}(f(x_k) - f(x_{k+1})) \\ &\leq \frac{2m}{\beta^2(1-\sqrt{1-m/L})}(f(x_k) - f(x_{k+1})) \quad \text{(Lemma 19)} \end{aligned}$$

Combining this with Lemma 15, we have that:

$$\begin{aligned} 2m[f(x_k) - f_*] &\leq \frac{2m}{\beta^2(1-\sqrt{1-m/L})}(f(x_k) - f(x_{k+1})) \\ \Rightarrow f(x_k) - f_* &\leq \frac{1}{\beta^2(1-\sqrt{1-m/L})}((f(x_k) - f_*) - (f(x_{k+1}) - f_*)) \\ \Rightarrow f(x_{k+1}) - f_* &\leq [1 - \beta^2(1-\sqrt{1-m/L})](f(x_k) - f_*) \end{aligned}$$

Applying this relation iteratively,

$$\begin{aligned} f(x_k) - f_* &\leq [1 - \beta^2(1-\sqrt{1-m/L})]^k(f(x_0) - f_*) \\ &\leq L[1 - \beta^2(1-\sqrt{1-m/L})]^k \|x_0 - x_*\|^2 \end{aligned}$$

where the last step follows from convexity and Lipschitz gradient. To achieve optimality gap ϵ, we therefore need

$$k \leq \frac{1}{\beta^2\left(1-\sqrt{1-\frac{m}{L}}\right)} \log\left(\frac{L\|x_0 - x_*\|^2}{\epsilon}\right).$$

steps (using Corollary 17).

We now proceed to prove the second statement, which bounds the number of function evaluations needed for the first k steps. Let T_k denote the step size at the beginning of the kth step of Algorithm 2, and t_k denote the step size used in the kth step. Let t_* denote the step size (in step k) that would be used in an exact line search.

Combining Propositions 24 and 25, we have that

$$\begin{aligned} n_k &\leq 4 + \log_\beta\left(\min\left(\frac{T_k}{t_*}, \frac{t_*}{T_k}\right)\right) \\ &\leq 4 + \log_\beta\left(\min\left(\frac{mT_k}{1+\sqrt{1-m/L}}, \frac{1-\sqrt{1-m/L}}{mT_k}\right)\right) \quad \text{(Lemma 19)} \end{aligned}$$

In particular,

$$\begin{aligned} n_1 &\leq 4 + \log_\beta\left(\min\left(\frac{mT_1}{1+\sqrt{1-m/L}}, \frac{1-\sqrt{1-m/L}}{mT_1}\right)\right) \\ &\leq 4 + \log_\beta\left(\min\left(\frac{mT_0}{\beta(1+\sqrt{1-m/L})}, \frac{\beta(1-\sqrt{1-m/L})}{mT_0}\right)\right) \end{aligned}$$

For $k > 1, T_k = \beta^{-1}t_{k-1} \in [\beta t_*, \beta^{-1}t_*]$ by Invariant 5. Combining this with Lemma 19, for $k > 1$ we have that $T_k \in \left[\frac{\beta}{m}(1-\sqrt{1-m/L}), \frac{1}{\beta m}(1+\sqrt{1-m/L})\right]$. Therefore,

$$n_{k>1} \leq 5 + \log_\beta\left(\frac{1-\sqrt{1-m/L}}{1+\sqrt{1-m/L}}\right)$$

Summing over the k steps:

$$\begin{aligned} N_k &= \sum_{i=0}^{k} n_i \\ &= 1 + n_1 + \sum_{i=2}^{k} n_i \\ &= \log_\beta\left(\min\left(\frac{mT_0}{\beta(1+\sqrt{1-m/L})}, \frac{\beta(1-\sqrt{1-m/L})}{mT_0}\right)\right) \\ &\qquad + 5k + (k-1)\log_\beta\left(\frac{1-\sqrt{1-m/L}}{1+\sqrt{1-m/L}}\right) \\ &= k\left(5 + \log_\beta\left(\frac{1-\sqrt{1-m/L}}{1+\sqrt{1-m/L}}\right)\right) \\ &\qquad + \log_\beta\left(\min\left(\frac{mT_0}{\beta(1-\sqrt{1-m/L})}, \frac{\beta(1+\sqrt{1-m/L})}{mT_0}\right)\right) \end{aligned}$$

■