SIAM J. OPTIM.
Vol. 31, No. 2, pp. 1489–1518

GLOBAL CONVERGENCE RATE ANALYSIS OF A GENERIC LINE SEARCH ALGORITHM WITH NOISE*

A. S. BERAHAS†, L. CAO‡, AND K. SCHEINBERG§

Abstract. In this paper, we develop convergence analysis of a modified line search method for objective functions whose value is computed with noise and whose gradient estimates are inexact and possibly random. The noise is assumed to be bounded in absolute value without any additional assumptions. We extend the framework based on stochastic methods from [C. Cartis and K. Scheinberg, *Math. Program.*, 169 (2018), pp. 337–375] which was developed to provide analysis of a standard line search method with exact function values and random gradients to the case of noisy functions. We introduce two alternative conditions on the gradient which, when satisfied with some sufficiently large probability at each iteration, guarantees convergence properties of the line search method. We derive expected complexity bounds to reach a near optimal neighborhood for convex, strongly convex and nonconvex functions. The exact dependence of the convergence neighborhood on the noise is specified.

Key words. nonlinear optimization, line search, convergence rates, derivative-free optimization

AMS subject classification. 90C30

DOI. 10.1137/19M1291832

1. Introduction.

We consider an unconstrained optimization problem of the form

$$\min_{x\in\mathbb{R}^n} \phi(x), \tag{1.1}$$

where $f(x,\xi) = \phi(x) + e(x,\xi)$ is computable, while $\phi(x)$ is not, and ξ is a random variable with associated probability space $(\Xi, \mathcal{F}, P)$. In other words $f : \mathbb{R}^n \times \Xi \to \mathbb{R}$ is a possibly noisy approximation of a smooth function $\phi : \mathbb{R}^n \to \mathbb{R}$, and the goal is to minimize ϕ. Alternatively, $f(x)$ may be a nonsmooth function and $\phi(x)$ its smooth approximation; see, for instance, [12, 16]. Such problems arise in a plethora of fields such as derivative-free optimization (DFO) [7, 11], simulation optimization [19], and machine learning. There has been a lot of work analyzing the case when $e : \mathbb{R}^n \times \Xi \to \mathbb{R}$ is a random function with zero mean. Here, we take a different research direction, allowing $e(x,\xi)$ to be stochastic, deterministic, or adversarial, but assuming that $|e(x,\xi)| \leq \epsilon_f$ for all $x \in \mathbb{R}^n$ and all realizations of ξ. While this is a strong assumption, it is often satisfied in practice when $f(x,\xi)$ is a result of a computer code aimed at computing (or approximating) $\phi(x)$, but has inaccuracies due to internal discretization [13, 14]. It will be evident from our analysis that the modified line search method makes progress as long as $\|\nabla\phi(x)\|$ is sufficiently large compared to the noise.

*Received by the editors October 7, 2019; accepted for publication (in revised form) March 16, 2021; published electronically June 8, 2021.
https://doi.org/10.1137/19M1291832
Funding: This work was partially supported by NSF grants CCF 16-18717 and TRIPODS 17-40796, by DARPA Lagrange award HR-001117S0039, and by a Google Faculty Award.
†Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI 48109 USA (albertberahas@gmail.com).
‡Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA 18015 USA (liyuan@lehigh.edu).
§School of Operations Research and Information Engineering, Cornell University, Ithaca, NY 14850 USA (katyas@cornell.edu).

Line searches are classical and well-known techniques for improving the performance of optimization algorithms [17]. They allow algorithms to be more robust, less dependent on the choices of hyperparameters and typically ensure faster practical convergence rates. However, in their original form they rely on exact function and gradient information. Many modern applications give rise to functions for which computing accurate function values and/or gradients is either impossible or prohibitively expensive. Thus, it is desirable to extend the line search paradigm and its analysis to such functions. In [6] a general line search algorithm was analyzed under the conditions that the function values are *exact* but the gradient estimates are *inexact* and *random*. It is shown that under certain (realizable) probabilistic conditions on the accuracy of the gradient estimates, the resulting line search has the same expected complexity (up to constants) as the line search based on exact gradients. In [3] a general framework for complexity analysis of stochastic optimization methods is proposed and applied to a trust region method. The same framework is used in [18] to analyze a line search method applied to *stochastic* functions. This framework is significantly more complicated than that in [6], but it also relies on casting the algorithm as a stochastic process (a submartingale), and it is again shown that the expected complexity is the same (up to constants) as that of regular deterministic gradient descent, under certain probabilistic, and realizable, conditions on stochastic function values and gradient estimates.

In this paper, we extend the analysis in [6] to apply to (1.1). In particular, we assume that the gradient estimates are random and the function values are noisy. Since the function values are noisy (unlike in [6]), the line search is modified to accept steps that may potentially increase the current estimated value. This modification causes significant changes in the analysis of the expected complexity rates, as the analysis in [6] heavily relies on the fact that an objective function can never be increased by the algorithm. Nevertheless, we are able to extend the results in [6] recovering expected complexity bounds for the cases of convex, strongly convex, and nonconvex objective functions. We note here that we derived the complexity bounds for the condition on the gradient accuracy used in [6], as well as for the so-called *norm condition* used, for example, in [4]. While, as we discuss later, each gradient accuracy condition can have advantages over the other, depending on the setting, they can be used interchangeably with relatively small adjustments to the analysis. Specifically, the analysis of the supermartingale is not affected by the choice of this condition, and the key steps of the analysis of the line search method itself are analogous; however, the constants stemming from the gradient condition appear differently in the final complexity bounds.

The conditions we impose on the line search algorithm are essentially the same as in [6], while the conditions required for the analysis of the stochastic line search [18], where the noise is unbounded, are more restrictive, and thus that analysis does not apply to the case we consider here. In particular, while the function value noise is allowed to be unbounded in [18], it is assumed that it is possible to reduce its variance below any given threshold, for example, by sample averaging. In contrast here we do not assume that the noise is stochastic, and thus we do not assume it can be reduced or controlled. Also, the algorithm itself in [18] is more complicated than a simple line search in order to handle unbounded noise. Moreover, the resulting bounds in [18] have worse dependence on constants than those in [6] and the bounds we derive in this paper. Finally, in both [3] and [18] the expected complexity bound is derived for any ϵ, arbitrarily small, under the assumption that the noise can be made arbitrarily small accordingly (at least with sufficient probability). Here we establish a

connection between the level of noise and the convergence neighborhood. Obtaining similar results for the setting in [3, 18] is nontrivial and is the subject of future work.

Our main motivation for this analysis is the recent popularity of smoothing methods for gradient estimates of black-box functions. Stochastic gradient approximations can be computed at relatively low costs, e.g., via Gaussian smoothing [9, 16, 20] and smoothing on a unit sphere [8], and used within a gradient descent algorithm. This approach has been analyzed in [16] and more recently used in several papers for the specific cases of policy optimization in reinforcement learning and online learning [8, 9, 20]. All of these papers employ specific fixed step length gradient descent schemes within limited settings (e.g., convex functions). Our goal is to develop convergence rate analyses for convex, strongly convex, and nonconvex functions, for a generic line search algorithm based on gradient approximations, that can apply not only to gradient descent, but also to quasi-Newton methods such as L-BFGS [17].

It turns out that the variance of the stochastic gradients computed via Gaussian and unit sphere smoothing can be bounded from above by the squared norm of the expectation, that is, $\|\nabla\phi(x)\|^2$, when ϕ is the smoothing function [2]. This motivates us to consider a simpler probabilistic condition on the accuracy of the gradient estimates in addition to the one used in [6].

Assumptions. Throughout the paper we make the following assumptions.

ASSUMPTION 1.1 (Lipschitz continuity of the gradients of $\boldsymbol{\phi}$). *The function ϕ is continuously differentiable, and the gradient of ϕ is L-Lipschitz continuous for all $x \in \mathbb{R}^n$.*

ASSUMPTION 1.2 (lower bound on $\boldsymbol{\phi}$). *The function ϕ is bounded below by a scalar $\hat{\phi}$.*

ASSUMPTION 1.3 (boundedness of noise in the function). *There is a constant $\epsilon_f \geq 0$ such that $|f(x,\xi) - \phi(x)| = |e(x,\xi)| \leq \epsilon_f$ for all $x \in \mathbb{R}^n$ and all realizations of ξ.*

Assumption 1.3 may seem very strong; however, we will show that under this assumption the modified line search algorithm converges to a neighborhood of the optimal solution whose size is defined by ϵ_f. Thus, if it is possible to control the value of ϵ_f, then one can tighten the convergence neighborhood. This is possible in many applications where, for example, values of $\phi(x)$ are obtained as a limit to some discretized computation and the error is controlled by the fineness of the discrete grid [13, 14] or if $\phi(x)$ is a smoothed approximation of $f(x,\xi)$ where the smoothing parameter controls the error between $f(x,\xi)$ and $\phi(x)$ [12, 16]. We stress here that our algorithms and analysis do *not* assume that the noise is stochastic or that the bound ϵ_f is controllable, just that it is known.

Summary of results. While we are motivated by some specific methods for computing gradient estimates, in the remainder of the paper, we simply aim to establish complexity bounds on a generic modified line search algorithm applied to the minimization of convex, strongly convex, and nonconvex functions, under the condition that the gradient estimate $g : \mathbb{R}^n \to \mathbb{R}^n$ satisfies

$$\|g(x) - \nabla\phi(x)\| \leq \theta\|\nabla\phi(x)\| \tag{1.2}$$

for sufficiently small θ with some probability $1 - \delta$.[1] The bound (1.2), known as the *norm condition*, was first introduced in [5] and consequently used in a variety

[1] The norms used in this paper are Euclidean norms.

of works (see, e.g., [4]). This bound is generally not realizable for generic stochastic gradient estimates; however, it can be made to hold for several deterministic and stochastic gradient estimates such as those used in [1, 7, 8, 16]. We establish expected complexity bounds similar to those in [6], where the line search is analyzed under a more complicated bound on $\|g(x) - \nabla\phi(x)\|$ using exact evaluations of ϕ (i.e., no noise in the function evaluations). The expected complexity bounds are established in terms of desired accuracy ϵ, under the assumption that ϵ is sufficiently big compared to the error level ϵ_f. We derive specific bounds on ϵ with respect to ϵ_f for convex, strongly convex, and nonconvex cases first for a gradient descent-type algorithm, and then for an algorithm that uses any general descent direction. For completeness, we derive complexity bounds for the condition on the gradient accuracy presented in [6], but, due to the presence of noise, this condition is somewhat modified.

Organization. The paper is organized as follows. In section 2 we describe a general line search algorithm that uses gradient approximations in lieu of the true gradient, and noisy function evaluations of the objective function. We present the stochastic analysis that allows us to bound the expected number of steps required by our generic scheme to reach a desired accuracy in section 3. This analysis is an extension of the results in [6] that accounts for noise in the objective function. In section 4, we apply the results of section 3 to derive global convergence rates and bounds on ϵ in terms of ϵ_f when the generic line search method is applied to convex, strongly convex, and nonconvex functions. Finally, in section 5 we make some concluding remarks and discuss avenues for future research.

2. A generic modified line search algorithm. In this section, we describe a generic line search algorithm that uses gradient approximations in lieu of the true gradient and that operates in the noisy regime. In general, line search algorithms construct a possibly noisy approximation of the gradient at the current iterate x_k, $g_k = g(x_k)$, and compute a search direction using this gradient estimate and possibly additional information, e.g., a quasi-Newton search direction. The step size parameter is then chosen; this could be constant, selected from a predetermined sequence of step lengths (e.g., diminishing) or adaptive (e.g., via a back-tracking Armijo line search [17, Chapter 3]). The framework of the generic line search method we analyze is given in Algorithm 2.1. As is clear from Algorithm 2.1, the key components of this method are (i) the construction of the gradient approximation (step 2), (ii) the choice of the search direction (step 3), and (iii) the choice of the step size parameter and the iterate update (step 4).

Algorithm 2.1 Generic Line Search Algorithm

Inputs: Starting point x_0, initial step size parameter $\alpha_0 > 0$.

1: **for** $k = 0, 1, 2, \ldots$ **do**
2: **Construct a gradient approximation g_k:**
Construct an approximation g_k of $\nabla\phi(x_k)$.
3: **Construct a search direction d_k:**
Construct a search direction d_k, e.g., $d_k = -g_k$ or $d_k = -H_k g_k$.
4: **Compute step size α_k and update the iterate.**

Algorithm 2.1 is a generic line search algorithm. We perform the analysis in section 4 for the case where $d_k = -g_k$ and then outline how the analysis can be easily modified to the case of a more general search direction d_k, under additional

assumptions on d_k. In order to prove theoretical convergence guarantees, we need to fully specify the manner in which the step size parameter is selected at every iteration and how a new iterate is computed (line 4). We consider Algorithm 2.1 for which the step size parameter α_k varies under the condition that α_k is chosen to satisfy a *modified* version of the sufficient decrease Armijo condition,

$$f(x_k + \alpha_k d_k, \xi) \leq f(x_k, \xi) + c_1 \alpha_k d_k^T g_k + 2\epsilon_f, \tag{2.1}$$

where $c_1 \in (0, 1)$ is the Armijo parameter, and ϵ_f is the upper bound on the noise in the objective function. Note that the random variable ξ may have two different realizations when computing $f(x_k + \alpha_k d_k, \xi)$ and $f(x_k, \xi)$; however, these realizations may be dependent, independent, or identical. This does not affect our analysis, and thus for simplicity we do not assign specific notation to different realizations of ξ. If a trial value α_k does not satisfy (2.1) for some particular realizations of ξ, then the iteration is called *unsuccessful*; the new iterate is set to the previous iterate, i.e., $x_{k+1} = x_k$, and the step size parameter is set to a (fixed) fraction $\tau \leq 1$ of the previous value, i.e., $\alpha_{k+1} \leftarrow \tau \alpha_k$. This step makes sense particularly when g_k (and thus d_k) are random vectors and thus can be different even for the same x_k. If the trial value satisfies (2.1), then the iteration is called *successful*, the new iterate is updated based on the search direction d_k, i.e., $x_{k+1} = x_k + \alpha_k d_k$, and the step size parameter is set to $\alpha_{k+1} \leftarrow \tau^{-1} \alpha_k$. Algorithm 2.2 fully specifies a subroutine for computing the step size parameter and taking a step. Note that if $\tau = 1$, Algorithm 2.1 is a constant step size parameter line search algorithm. Algorithm 2.2 receives ϵ_f as input from Algorithm 2.1. We do not specify here if Algorithm 2.1 receives this quantity as input from the user or has an ability to estimate it, as it may depend on a particular case.

Algorithm 2.2 Line Search Subroutine

Inputs: Current iterate x_k, current gradient estimate g_k, current search direction d_k, current step size parameter α_k, backtracking factor $\tau \in (0, 1]$, Armijo parameter $c_1 \in (0, 1)$, bound on the noise ϵ_f.

1: **for** $k = 0, 1, 2, \ldots$ **do**
2: **Check sufficient decrease:**
Check if (2.1) is satisfied.
3: **if** Condition Satisfied **(*successful* step) then**
$x_{k+1} = x_k + \alpha_k d_k$ and $\alpha_{k+1} \leftarrow \tau^{-1} \alpha_k$.
4: **else**
$x_{k+1} = x_k$ and $\alpha_{k+1} \leftarrow \tau \alpha_k$.
5: **Outputs:** New iterate x_{k+1}, new step size parameter α_{k+1}.

The modified Armijo condition has been used in [1]. The addition of the term $2\epsilon_f$ ensures that a step is *successful* if α_k is small enough and $d_k^T g_k$ is large enough. In [1] the case of functions with arbitrary but bounded noise, such as the ones considered here, were considered. However, unlike this paper the error of the gradient estimates was also assumed to be bounded by a constant, and convergence rates were derived for strongly convex objectives only.

3. Analysis of the underlying stochastic process. In this section, we describe the general mechanism that is used to provide the theoretical results of the paper. This analysis is an extension of the analysis provided in [6] that accounts for possible noise in the function evaluations, i.e., $e(x) \neq 0$.

We begin by introducing several definitions, key assumptions, and theoretical results, similar to those in [6] but suitably modified as required for the analysis in this paper. In particular, similar to [6], we view Algorithm 2.1 as a stochastic process, generated from a sequence of random function realizations $f(x_k, \xi)$ and gradient estimates G_k. With some abuse of notation and for simplicity of presentation, we introduce the new probability space $(\Omega, \mathcal{F}, P)$, which includes the randomness in both the function and the gradient realizations. Since the function realizations used by the line search are essentially replaced by their upper and lower bounds in our analysis, the nature of Ξ has no effect on it.

The following quantities are random and are important in the analysis: the gradient estimate G_k, the step size parameter $\mathcal{A}_k$, and the search direction $\mathcal{D}_k$. Realizations of these random quantities are denoted by $g_k = G_k(\omega_k)$, $x_k = X_k(\omega_k)$, $\alpha_k = \mathcal{A}_k(\omega_k)$, and $d_k = \mathcal{D}_k(\omega_k)$, respectively. For brevity, we will omit the ω_k in the notation. The iterate X_k, given X_{k-1} and $\mathcal{A}_{k-1}$, is fully determined by G_{k-1} and the noise in the function value estimation during iteration $k-1$. The noise may be stochastic or deterministic; let $\mathcal{E}_{k-1}$ denote all noise history up to iteration $k-1$. Note that our algorithm and its analysis are independent of the nature of the noise, but we include $\mathcal{E}_{k-1}$ in the algorithm history for completeness. We use $\mathcal{F}_{k-1}^{G,\mathcal{E}} = \sigma(G_0, \ldots, G_{k-1}, \mathcal{E}_{k-1})$ to denote the σ-algebra generated by $G_0, \ldots, G_{k-1}$ and $\mathcal{E}_{k-1}$, that is to say, generated by Algorithm 2.1 up to the start of iteration k.

Sufficiently accurate gradients. We assume that the random gradient approximations G_k satisfy some notion of *good quality* with probability $1-\delta$. We use the following general notion of *sufficiently accurate* gradients, similar to that presented in [6].

DEFINITION 3.1. *A sequence of random gradients* $\{G_k\}$ *is* $(1-\delta)$*-probabilistically "sufficiently accurate" for Algorithm* 2.1 *if the indicator variables*

$$I_k = \mathbb{1}\{G_k \text{ is a sufficiently accurate gradient of } \phi \text{ for the given } \mathcal{A}_k, X_k, \text{ and } \mathcal{D}_k\}$$

satisfy the submartingale condition

$$\mathbb{P}(I_k = 1 | \mathcal{F}_{k-1}^{G,\mathcal{E}}) \geq 1 - \delta \tag{3.1}$$

for all realizations of $\mathcal{F}_{k-1}^{G,\mathcal{E}}$*, where* $\mathcal{F}_{k-1}^{G,\mathcal{E}} = \sigma(G_0, \ldots, G_{k-1}, \mathcal{E}_{k-1})$ *is the* σ*-algebra generated by* $G_0, \ldots, G_{k-1}$ *and* $\mathcal{E}_{k-1}$*. Moreover, we say that iteration* k *is a* true *iteration if the event* $I_k = 1$ *occurs; otherwise the iteration is called* false*.*

Definition 3.1 is generic, but somewhat less so than the equivalent definition in [6, Definition 2.1], where second order models are also considered and as a result the definition of *"sufficient accuracy"* is not restricted to gradients. The reason Definition 3.1 is generic is because it can be particularized differently depending on the way the gradient estimates are generated. Specifically, in section 4 we define *sufficiently accurate* in two different ways and derive expected complexity bounds for Algorithm 2.1. The first definition is motivated by the specific setting where estimates g_k are computed via finite differences, interpolation, or smoothing [1, 2, 7, 8, 16]. The second definition is similar to that presented in [6].

Number of iterations N_ϵ to reach ϵ accuracy. The main goal of this section is to derive bounds on the expected number of iterations $\mathbb{E}[N_\epsilon]$ required to reach a desired level of accuracy ϵ. We formally define N_ϵ as follows.

DEFINITION 3.2.

- *If ϕ is convex or strongly convex: N_ϵ is the number of iterations required until $\phi(X_k) - \phi^\star \leq \epsilon$ occurs for the first time. Note that $\phi^\star = \phi(x^\star)$, where $x^\star$ is a global minimizer of ϕ.*
- *If ϕ is nonconvex: N_ϵ is the number of iterations required until $\|\nabla\phi(X_k)\| \leq \epsilon$ occurs for the first time.*

Thus N_ϵ is a random variable with the property $\sigma(\mathbb{1}\{N_\epsilon > k\}) \subset \mathcal{F}_{k-1}^{G,\mathcal{E}}$, and thus it is a stopping time for our stochastic process; see [6, section 2]. To bound $\mathbb{E}[N_\epsilon]$ we assume that while $k < N_\epsilon$ the stochastic process induced by Algorithm 2.1 behaves in a certain way. Specifically, it tends to make a certain amount of progress towards optimality.

Measure of progress towards optimality and upper bound. As is done in [6, section 2], let Z_k denote a measure of progress towards optimality (from any starting point $x_0 \in \mathbb{R}^n$), and let Z_ϵ be an upper bound for Z_k for $k < N_\epsilon$.[2] In particular, our analysis will use the definitions of Z_k and Z_ϵ as described in Table 3.1.

TABLE 3.1
Definitions of Z_k and Z_ϵ for convex, strongly convex, and nonconvex functions.

Function	Z_k	Z_ϵ
Convex	$\frac{1}{\phi(X_k)-\phi^\star} - \frac{1}{\phi(X_0)-\phi^\star}$	$\frac{1}{\epsilon} - \frac{1}{\phi(X_0)-\phi^\star}$
Strongly convex	$\log\left(\frac{\phi(X_0)-\phi^\star}{\phi(X_k)-\phi^\star}\right)$	$\log\left(\frac{\phi(X_0)-\phi^\star}{\epsilon}\right)$
Nonconvex	$\phi(X_0) - \phi(X_k)$	$\phi(X_0) - \hat{\phi}$

We are now ready to introduce the key assumption of the behavior of the stochastic process $\{\mathcal{A}_k, Z_k\}$ generated by Algorithm 2.1 under which we derive a bound on $\mathbb{E}[N_\epsilon]$. In section 4, we show that this assumption holds for our generic line search algorithm, under a particular definition of *sufficiently accurate* gradient estimates, and thus we will be able to derive the expected complexity bound.

Recall that when the gradient estimate g_k is *sufficiently accurate*, the iteration is called *true*, and this is assumed to happen with probability at least $1-\delta$, conditioned on the past. The following assumption is a modification of the assumption in [6, section 2.4, Assumption 2.1]. Let $z_k = Z_k(\omega_k)$ be a realization of the random quantity Z_k. Note that $z_k = Z_k(\omega_k)$ is a measure of progress towards optimality.

ASSUMPTION 3.3. *There exist a constant $\bar{\alpha} > 0$, a nondecreasing function $h(\alpha) : \mathbb{R} \to \mathbb{R}$, which satisfies $h(\alpha) > 0$ for any $\alpha > 0$, and a nondecreasing function $r(\epsilon_f) : \mathbb{R} \to \mathbb{R}$, which satisfies $r(\epsilon_f) \geq 0$ for any $\epsilon_f \geq 0$, such that for any realization of Algorithm* 2.1 *the following hold for all $k < N_\epsilon$:*

(i) *If iteration k is* true *(i.e., $I_k = 1$) and* successful*, then $z_{k+1} \geq z_k + h(\alpha_k) - r(\epsilon_f)$.*
(ii) *If $\alpha_k \leq \bar{\alpha}$ and iteration k is* true*, then iteration k is also* successful*, which implies $\alpha_{k+1} = \tau^{-1}\alpha_k$.*
(iii) *$z_{k+1} \geq z_k - r(\epsilon_f)$ for all* successful *iterations k and $z_{k+1} \geq z_k$ for every* unsuccessful *iteration k.*
(iv) *The ratio $r(\epsilon_f)/h(\bar{\alpha})$ is bounded from above by some $\gamma \in (0,1)$.*

[2] F_k and F_ϵ is the notation used in [6].

Assumption 3.3 provides guarantees of progress for the process Z_k, using guaranteed increase $h(\alpha_k)$ and possible decrease $r(\epsilon_f)$. These quantities will be specified for each case (convex, strongly convex, nonconvex) in section 4. The key difference between Assumption 3.3 and the corresponding assumption in [6, section 2.4, Assumption 2.1] is that on each *successful* iteration Z_k may decrease by up to $r(\epsilon_f)$. When $r(\epsilon_f) = 0$, Assumption 3.3 reduces to the assumption in [6, section 2.4, Assumption 2.1] and in this case γ can be chosen arbitrarily close to 0. When $r(\epsilon_f) > 0$, the process Z_k may decrease on some *successful* iterations; see Assumption 3.3(iii). Assumption 3.3(i) states that Z_k is guaranteed to increase on *true successful* iterations by at least the quantity $h(\alpha_k) - r(\epsilon_f)$, which is positive due to Assumption 3.3(iv). The constant γ serves as a parameter that measures how much $h(\alpha_k)$ dominates $r(\epsilon_f)$. As we will see, γ can be chosen to be fixed, for example, $\gamma = \frac{1}{2}$, and Assumption 3.3(iv) then simply dictates that $h(\alpha_k) \geq 2r(\epsilon_f)$. The guaranteed value of progress $h(\alpha_k)$ is larger when the target accuracy ϵ is larger, which in turn implies the connection between the level of noise ϵ_f and the target accuracy ϵ. In other words, γ is not an algorithmic parameter, it is simply a parameter whose value implies a particular bound on the neighborhood of convergence.

As in [6] we define the following additional indicator random variables:

$$\Lambda_k = \mathbb{1}\{\mathcal{A}_k > \bar{\alpha}\}, \qquad \bar{\Lambda}_k = \mathbb{1}\{\mathcal{A}_k \geq \bar{\alpha}\},$$

$$\Theta_k = \mathbb{1}\{\text{Iteration } k \text{ is } \textit{successful}, \text{ i.e., } \mathcal{A}_{k+1} = \tau^{-1}\mathcal{A}_k\}.$$

Note that $\sigma(\Lambda_k) \subset \mathcal{F}_{k-1}^{G,\mathcal{E}}$, $\sigma(\bar{\Lambda}_k) \subset \mathcal{F}_{k-1}^{G,\mathcal{E}}$, and $\sigma(\Theta_k) \subset \mathcal{F}_k^{G,\mathcal{E}}$, that is, the random variables Λ_k and $\bar{\Lambda}_k$ are fully determined by the first $k-1$ steps of the algorithm, while Θ_k is fully determined by the first k steps.

Without loss of generality, we assume that $\bar{\alpha} = \tau^c \alpha_0$ for some positive integer c. In other words, $\bar{\alpha}$ is the largest value that the step size $\mathcal{A}_k$ actually achieves for which part (ii) of Assumption 3.3 holds. Note that if $\tau = 1$, the algorithm uses a constant step size and hence has to start with the value for which Assumption 3.3 holds, i.e., $\alpha \leq \bar{\alpha}$, in order to converge.

In summary, under Assumption 3.3, recalling the update rules for α_k in Algorithm 2.1, we can write the stochastic process $\{\mathcal{A}_k, Z_k\}$ as obeying the expressions below:

(3.2)
$$\mathcal{A}_{k+1} = \begin{cases} \tau^{-1}\mathcal{A}_k & \text{if } \Theta_k = 1, \\ \tau\mathcal{A}_k & \text{if } \Theta_k = 0, \end{cases} = \begin{cases} \tau^{-1}\mathcal{A}_k & \text{if } I_k = 1 \text{ and } \Lambda_k = 0, \\ \tau^{-1}\mathcal{A}_k & \text{if } \Theta_k = 1,\ I_k = 0, \text{ and } \Lambda_k = 0, \\ \tau\mathcal{A}_k & \text{if } \Theta_k = 0,\ I_k = 0, \text{ and } \Lambda_k = 0, \\ \tau^{-1}\mathcal{A}_k & \text{if } \Theta_k = 1 \text{ and } \Lambda_k = 1, \\ \tau\mathcal{A}_k & \text{if } \Theta_k = 0 \text{ and } \Lambda_k = 1, \end{cases}$$

(3.3)
$$Z_{k+1} \geq \begin{cases} Z_k + h(\mathcal{A}_k) - r(\epsilon_f) & \text{if } \Theta_k = 1 \text{ and } I_k = 1, \\ Z_k - r(\epsilon_f) & \text{if } \Theta_k = 1 \text{ and } I_k = 0, \\ Z_k & \text{if } \Theta_k = 0. \end{cases}$$

3.1. Analysis of the stochastic processes. We now present the derivation of the bounds on $\mathbb{E}[N_\epsilon]$ under Assumption 3.3 by modifying the analysis in [6]. We start by introducing a useful lemma from [6].

LEMMA 3.4. *Let N_ϵ denote the stopping time. For all $k < N_\epsilon$, let I_k be the sequence of random variables in Definition* 3.1 *so that* (3.1) *holds. Let W_k be a nonnegative stochastic process such that $\sigma(W_k) \subset \mathcal{F}_{k-1}^{G,\mathcal{E}}$ for any $k \geq 0$. Then*

$$\mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1} W_k I_k\right] \geq (1-\delta)\mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1} W_k\right].$$

Similarly,

$$\mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1} W_k(1-I_k)\right] \leq \delta\mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1} W_k\right].$$

For brevity, we omit the proof of Lemma 3.4; see [6, Lemma 2.3].

The following lemma from [6] bounds the number of steps for which $\alpha_k \leq \bar{\alpha}$. The proof depends only on the probabilities of different outcomes and not on the changes in Z_k; thus the proof from [6] applies directly.

LEMMA 3.5. *The expected number of steps for which $\alpha_k \leq \bar{\alpha}$ can be bounded as*

$$\mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1}(1-\Lambda_k)\right] \leq \frac{1}{2(1-\delta)}\mathbb{E}[N_\epsilon].$$

Proof. The proof uses Lemma 3.4 with $W_k = 1-\Lambda_k$ and is the same as in [6]. ∎

We now turn to the derivation of the bound on

$$\mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1}\Lambda_k\right],$$

which requires a substantially more elaborate analysis than that in [6] but is similar in spirit. The key difference is that, while in [6] Z_k never decreases, here we have to account for all iterations where Z_k may decrease, and bound their expected number. For brevity of notation, we define the following quantities:

- $N_{FS} = \sum_{k=0}^{N_\epsilon-1}\bar{\Lambda}_k(1-I_k)\Theta_k$: the number of *false successful* iterations with $\mathcal{A}_k \geq \bar{\alpha}$.
- $N_{TS} = \sum_{k=0}^{N_\epsilon-1}\bar{\Lambda}_k I_k\Theta_k$: the number of *true successful* iterations with $\mathcal{A}_k \geq \bar{\alpha}$.
- $N_F = \sum_{k=0}^{N_\epsilon-1}\bar{\Lambda}_k(1-I_k)$: the number of *false* iterations with $\mathcal{A}_k \geq \bar{\alpha}$.
- $N_T = \sum_{k=0}^{N_\epsilon-1}\bar{\Lambda}_k I_k$: the number of *true* iterations with $\mathcal{A}_k \geq \bar{\alpha}$.
- $N_{TU} = \sum_{k=0}^{N_\epsilon-1}\Lambda_k I_k(1-\Theta_k)$: the number of *true unsuccessful* iterations with $\mathcal{A}_k > \bar{\alpha}$.
- $N_U = \sum_{k=0}^{N_\epsilon-1}\Lambda_k(1-\Theta_k)$: the number of *unsuccessful* iterations with $\mathcal{A}_k > \bar{\alpha}$.
- $N_{SS} = \sum_{k=0}^{N_\epsilon-1}(1-\bar{\Lambda}_k)\Theta_k$: the number of *successful* iterations with $\mathcal{A}_k < \bar{\alpha}$ (*small* $\mathcal{A}_k$).

Since

$$\mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1}\Lambda_k\right] = \mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1}\Lambda_k(1-I_k)\right] + \mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1}\Lambda_k I_k\right] \leq \mathbb{E}[N_F] + \mathbb{E}[N_T], \tag{3.4}$$

our goal is to bound $\mathbb{E}[N_F] + \mathbb{E}[N_T]$.

We now establish several inequalities relating the quantities we just defined. We begin with

$$N_T = N_{TS} + N_{TU} \leq N_{TS} + N_U. \tag{3.5}$$

The equality above holds because by Assumption 3.3(ii) there are no *true unsuccessful* iterations when $\mathcal{A}_k = \bar{\alpha}$.

LEMMA 3.6. *For any $l \in \{0, \ldots, N_\epsilon - 1\}$ and for all realizations of Algorithm* 2.1*, we have*

$$\sum_{k=0}^{l} \Lambda_k (1 - \Theta_k) \leq \sum_{k=0}^{l} \bar{\Lambda}_k \Theta_k + \log_\tau \left(\frac{\bar{\alpha}}{\alpha_0} \right),$$

and hence when $l = N_\epsilon - 1$,

$$N_T \leq N_{FS} + 2N_{TS} + \log_\tau \left(\frac{\bar{\alpha}}{\alpha_0} \right). \tag{3.6}$$

Proof. On *successful* iterations $\mathcal{A}_k$ is increased and on *unsuccessful* iterations $\mathcal{A}_k$ is decreased. Hence, the total number of steps when $\mathcal{A}_k > \bar{\alpha}$ and $\mathcal{A}_k$ is decreased is bounded by the total number of steps when $\mathcal{A}_k \geq \bar{\alpha}$ is increased plus the number of steps required to reduce $\mathcal{A}_k$ from its initial value α_0 to $\bar{\alpha}$. The first inequality of the lemma is a simple consequence of this observation.

Now for $l = N_\epsilon - 1$ this inequality becomes

$$N_U \leq N_{TS} + N_{FS} + \log_\tau \left(\frac{\bar{\alpha}}{\alpha_0} \right),$$

which, combined with (3.5), gives us (3.6). ∎

LEMMA 3.7. *The expected number of* false *iterations with $\mathcal{A}_k \geq \bar{\alpha}$ can be bounded as*

$$\mathbb{E}[N_F] \leq \frac{\delta}{1 - \delta} \mathbb{E}[N_T].$$

Proof. The proof uses Lemma 3.4 and is the same as in [6]. ∎

Hence, by (3.5) and Lemmas 3.6 and 3.7, we have

$$\begin{aligned} \mathbb{E}[N_F] + \mathbb{E}[N_T] &\leq \frac{1}{1 - \delta} \mathbb{E}[N_T] \\ &\leq \frac{1}{1 - \delta} \left(\mathbb{E}[N_{TS}] + \mathbb{E}[N_U] \right) \\ &\leq \frac{1}{1 - \delta} \left(\mathbb{E}[N_{FS}] + 2\mathbb{E}[N_{TS}] + \log_\tau \left(\frac{\bar{\alpha}}{\alpha_0} \right) \right). \end{aligned} \tag{3.7}$$

We now bound $\mathbb{E}[N_{SS}]$, the number of *successful* iterations with $\mathcal{A}_k < \bar{\alpha}$.

LEMMA 3.8. *The expected number of* successful *iterations with $\mathcal{A}_k < \bar{\alpha}$ can be bounded as*

$$\mathbb{E}\left[N_{SS}\right] = \mathbb{E}\left[\sum_{k=0}^{N_\epsilon - 1} (1 - \bar{\Lambda}_k) \Theta_k \right] \leq \frac{\delta}{2(1 - \delta)} \mathbb{E}[N_\epsilon].$$

Proof. We want to bound the expected number of *successful* iterations for which $\alpha_k < \bar{\alpha}$. Since on all *successful* iterations α_k is increased, and $\alpha_0 \geq \bar{\alpha}$, then for each such *successful* iteration there has to be an *unsuccessful* iteration with $\alpha_k \leq \bar{\alpha}$. Hence,

$$\sum_{k=0}^{N_\epsilon-1}(1-\bar{\Lambda}_k)\Theta_k \leq \sum_{k=0}^{N_\epsilon-1}(1-\Lambda_k)(1-\Theta_k) \leq \sum_{k=0}^{N_\epsilon-1}(1-\Lambda_k)(1-I_k).$$

The last inequality follows from the fact that when $\alpha_k \leq \bar{\alpha}$, all *true* iterations are *successful*, which implies $(1-\Lambda_k)I_k \leq (1-\Lambda_k)\Theta_k$. Now applying Lemmas 3.4 and 3.5 we have

$$\mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1}(1-\Lambda_k)(1-I_k)\right] \leq \delta\mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1}(1-\Lambda_k)\right] \leq \frac{\delta}{2(1-\delta)}\mathbb{E}[N_\epsilon],$$

from which the result follows. ∎

The next observation is central to our analysis. It reflects the fact that the total gain minus the total loss in Z_k is bounded from above by Z_ϵ. We observe that when $\mathcal{A}_k \geq \bar{\alpha}$ on *true successful* iterations this gain is bounded from below away from zero by $h(\bar{\alpha}) - r(\epsilon_f) \geq (1-\gamma)h(\bar{\alpha})$, and at other *successful* iterations the loss is bounded above by $r(\epsilon_f)$. This will allow us to bound $\mathbb{E}[N_{TS}]$.

LEMMA 3.9. *The number of* true successful *iterations with* $\mathcal{A}_k \geq \bar{\alpha}$ *can be bounded as*

$$N_{TS} \leq \frac{Z_\epsilon}{(1-\gamma)h(\bar{\alpha})} + \frac{\gamma}{1-\gamma}(N_{FS}+N_{SS}) \tag{3.8}$$

and, hence,

$$\mathbb{E}\left[N_{TS}\right] \leq \frac{Z_\epsilon}{(1-\gamma)h(\bar{\alpha})} + \frac{\gamma}{1-\gamma}\mathbb{E}\left[N_{FS}\right] + \frac{\gamma}{1-\gamma}\frac{\delta}{2(1-\delta)}\mathbb{E}[N_\epsilon]. \tag{3.9}$$

Proof. The proof follows directly from (3.3) and Assumption 3.3. Z_k is increased by at least $h(\bar{\alpha}) - r(\epsilon_f)$ at each *true successful* iteration when $\alpha_k \geq \bar{\alpha}$, and it may be decreased by at most $r(\epsilon_f)$ at each *false successful* iteration when $\alpha_k \geq \bar{\alpha}$ and at each *successful* iteration when $\alpha_k < \bar{\alpha}$. Thus, we have

$$Z_\epsilon \geq Z_k \geq N_{TS}(h(\bar{\alpha}) - r(\epsilon_f)) - r(\epsilon_f)(N_{FS}+N_{SS}).$$

Recalling that, by Assumption 3.3, $r(\epsilon_f) \leq \gamma h(\bar{\alpha})$ and $\gamma \in (0,1)$ we obtain (3.8), while (3.9) follows further from Lemma 3.8. ∎

LEMMA 3.10. *Under the condition that* $\delta < \frac{1}{2} - \frac{\gamma}{2}$, *the number of* false successful *iterations with* $\mathcal{A}_k \geq \bar{\alpha}$ *can be bounded as*

$$\mathbb{E}\left[N_{FS}\right] \leq \frac{2\delta}{1-2\delta-\gamma}\frac{Z_\epsilon}{h(\bar{\alpha})} + \frac{(1-\gamma)\delta}{1-2\delta-\gamma}\log_\tau\left(\frac{\bar{\alpha}}{\alpha_0}\right) + \frac{\delta^2\gamma}{(1-\delta)(1-2\delta-\gamma)}\mathbb{E}[N_\epsilon].$$

Proof. From (3.6) and Lemma 3.7, we have

$$\mathbb{E}\left[N_{FS}\right] \leq \mathbb{E}\left[N_F\right] \leq \frac{\delta}{1-\delta}\left[\mathbb{E}\left[N_{FS}\right] + 2\mathbb{E}\left[N_{TS}\right] + \log_\tau\left(\frac{\bar{\alpha}}{\alpha_0}\right)\right].$$

Then from Lemma 3.9 if follows that

$$\mathbb{E}\left[N_{FS}\right] \leq \frac{\delta}{1-\delta}\left[\frac{1+\gamma}{1-\gamma}\mathbb{E}\left[N_{FS}\right] + \frac{2Z_\epsilon}{(1-\gamma)h(\bar{\alpha})} + \frac{\gamma}{1-\gamma}\frac{\delta}{1-\delta}\mathbb{E}[N_\epsilon] + \log_\tau\left(\frac{\bar{\alpha}}{\alpha_0}\right)\right].$$

Collecting the terms involving $\mathbb{E}\left[N_{FS}\right]$ on the left and observing that $1 - \frac{1+\gamma}{1-\gamma}\frac{\delta}{1-\delta} = \frac{1-2\delta-\gamma}{(1-\gamma)(1-\delta)}$ we can derive the bound

$$\mathbb{E}\left[N_{FS}\right] \leq \frac{(1-\gamma)\delta}{1-2\delta-\gamma}\left[\frac{2Z_\epsilon}{(1-\gamma)h(\bar{\alpha})} + \frac{\gamma}{1-\gamma}\frac{\delta}{1-\delta}\mathbb{E}[N_\epsilon] + \log_\tau\left(\frac{\bar{\alpha}}{\alpha_0}\right)\right],$$

from which the result follows. ∎

We can now derive the bound for $\mathbb{E}\left[N_{TS}\right]$ using Lemmas 3.9 and 3.10 and collecting the appropriate terms.

LEMMA 3.11. *Under the condition that* $\delta < \frac{1}{2} - \frac{\gamma}{2}$*, the number of* true successful *iterations with* $\mathcal{A}_k \geq \bar{\alpha}$ *can be bounded as*

$$\mathbb{E}\left[N_{TS}\right] \leq \frac{1-2\delta}{1-2\delta-\gamma}\frac{Z_\epsilon}{h(\bar{\alpha})} + \frac{\gamma\delta}{1-2\delta-\gamma}\log_\tau\left(\frac{\bar{\alpha}}{\alpha_0}\right) + \frac{\gamma(1-2\delta)\delta}{2(1-\delta)(1-2\delta-\gamma)}\mathbb{E}[N_\epsilon].$$

Proof. From Lemma 3.9, we have

$$\mathbb{E}\left[N_{TS}\right] \leq \frac{Z_\epsilon}{(1-\gamma)h(\bar{\alpha})} + \frac{\gamma}{1-\gamma}\mathbb{E}\left[N_{FS}\right] + \frac{\gamma}{1-\gamma}\frac{\delta}{2(1-\delta)}\mathbb{E}[N_\epsilon].$$

Using the result from Lemma 3.10, it follows that

$$\begin{aligned}
&\mathbb{E}\left[N_{TS}\right] \\
\leq &\frac{Z_\epsilon}{(1-\gamma)h(\bar{\alpha})} \\
&+ \frac{\gamma}{1-\gamma}\left[\frac{2\delta}{1-2\delta-\gamma}\frac{Z_\epsilon}{h(\bar{\alpha})} + \frac{(1-\gamma)\delta}{1-2\delta-\gamma}\log_\tau\left(\frac{\bar{\alpha}}{\alpha_0}\right) + \frac{\delta^2\gamma}{(1-\delta)(1-2\delta-\gamma)}\mathbb{E}[N_\epsilon]\right] \\
&+ \frac{\gamma}{1-\gamma}\frac{\delta}{2(1-\delta)}\mathbb{E}[N_\epsilon] \\
= &\frac{1-2\delta}{1-2\delta-\gamma}\frac{Z_\epsilon}{h(\bar{\alpha})} + \frac{\gamma\delta}{1-2\delta-\gamma}\log_\tau\left(\frac{\bar{\alpha}}{\alpha_0}\right) + \frac{\gamma(1-2\delta)\delta}{2(1-\delta)(1-2\delta-\gamma)}\mathbb{E}[N_\epsilon],
\end{aligned}$$

which completes the proof. ∎

LEMMA 3.12. *Under the condition that* $\delta < \frac{1}{2} - \frac{\gamma}{2}$*, the number of iterations with* $\mathcal{A}_k > \bar{\alpha}$ *can be bounded as*

$$\mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1}\Lambda_k\right] \leq \frac{2}{1-2\delta-\gamma}\frac{Z_\epsilon}{h(\bar{\alpha})} + \frac{(1-\gamma)}{1-2\delta-\gamma}\log_\tau\left(\frac{\bar{\alpha}}{\alpha_0}\right) + \frac{\gamma\delta}{(1-\delta)(1-2\delta-\gamma)}\mathbb{E}[N_\epsilon].$$

Proof. By (3.4), (3.7), and Lemmas 3.10 and 3.11, we have

$$\begin{aligned}
&\mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1}\Lambda_k\right] \le \mathbb{E}[N_F] + \mathbb{E}[N_T] \\
\le&\frac{1}{1-\delta}\left[\mathbb{E}[N_{FS}] + 2\mathbb{E}[N_{TS}] + \log_\tau(\bar\alpha/\alpha_0)\right] \\
\le&\frac{1}{1-\delta}\left[\frac{2\delta}{1-2\delta-\gamma}\frac{Z_\epsilon}{h(\bar\alpha)} + \frac{(1-\gamma)\delta}{1-2\delta-\gamma}\log_\tau\left(\frac{\bar\alpha}{\alpha_0}\right) + \frac{\delta^2\gamma}{(1-\delta)(1-2\delta-\gamma)}\mathbb{E}[N_\epsilon]\right] \\
&+\frac{2}{1-\delta}\left[\frac{1-2\delta}{1-2\delta-\gamma}\frac{Z_\epsilon}{h(\bar\alpha)} + \frac{\gamma\delta}{1-2\delta-\gamma}\log_\tau\left(\frac{\bar\alpha}{\alpha_0}\right) + \frac{\gamma(1-2\delta)\delta}{2(1-\delta)(1-2\delta-\gamma)}\mathbb{E}[N_\epsilon]\right] \\
&+\frac{1}{1-\delta}\log_\tau\left(\frac{\bar\alpha}{\alpha_0}\right) \\
=&\frac{2}{1-2\delta-\gamma}\frac{Z_\epsilon}{h(\bar\alpha)} + \frac{(1-\gamma)}{1-2\delta-\gamma}\log_\tau\left(\frac{\bar\alpha}{\alpha_0}\right) + \frac{\gamma\delta}{(1-\delta)(1-2\delta-\gamma)}\mathbb{E}[N_\epsilon],
\end{aligned}$$

which completes the proof. ∎

Combining Lemmas 3.5 and 3.12, we have the key bound

$$\begin{aligned}
\mathbb{E}\left[N_\epsilon\right] &\le \mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1}\Lambda_k\right] + \mathbb{E}\left[\sum_{k=0}^{N_\epsilon-1}(1-\Lambda_k)\right] \\
&\le \frac{2}{1-2\delta-\gamma}\frac{Z_\epsilon}{h(\bar\alpha)} + \frac{(1-\gamma)}{1-2\delta-\gamma}\log_\tau\left(\frac{\bar\alpha}{\alpha_0}\right) \\
&\quad + \frac{\gamma\delta}{(1-\delta)(1-2\delta-\gamma)}\mathbb{E}[N_\epsilon] + \frac{1}{2(1-\delta)}\mathbb{E}[N_\epsilon].
\end{aligned}$$

Collecting the terms with $\mathbb{E}\left[N_\epsilon\right]$ on the left-hand side and multiplying both sides by $1-2\delta-\gamma$ we obtain

$$\left[1-2\delta-\gamma-\frac{\gamma\delta}{1-\delta}-\frac{1-2\delta-\gamma}{2(1-\delta)}\right]\mathbb{E}\left[N_\epsilon\right] \le \frac{2Z_\epsilon}{h(\bar\alpha)} + (1-\gamma)\log_\tau\left(\frac{\bar\alpha}{\alpha_0}\right).$$

If the coefficient in front of $\mathbb{E}\left[N_\epsilon\right]$ is positive, that immediately gives us a bound on the expected stopping time $\mathbb{E}\left[N_\epsilon\right]$. This coefficient is

$$1-2\delta-\gamma-\frac{\gamma\delta}{1-\delta}-\frac{1-2\delta-\gamma}{2(1-\delta)} = \frac{4\delta^2-4\delta+1-\gamma}{2(1-\delta)} = \frac{(1-2\delta)^2-\gamma}{2(1-\delta)}.$$

The smaller of the two roots of $4\delta^2-4\delta+1-\gamma$ is $\frac{1}{2}-\frac{\sqrt{\gamma}}{2} \le \frac{1}{2}-\frac{\gamma}{2}$. Hence, we have the following final bound.

THEOREM 3.13. *Under the condition that $\delta < \frac{1}{2}-\frac{\sqrt{\gamma}}{2}$, the stopping time N_ϵ is bounded in expectation as follows:*

$$\mathbb{E}[N_\epsilon] \le \frac{2(1-\delta)}{(1-2\delta)^2-\gamma}\left[\frac{2Z_\epsilon}{h(\bar\alpha)} + (1-\gamma)\log_\tau\left(\frac{\bar\alpha}{\alpha_0}\right)\right]. \tag{3.10}$$

REMARK 3.14. *The result of Theorem* 3.13 *is a generalization of the result in* [6] *to the case where the function is computed with some noise. Specifically, when $\epsilon_f = 0$, and as a result $r(\epsilon_f) = 0$, then $\gamma = 0$ and* (3.10) *reduces to the bounds in*

[6]. *We should note that the condition* $\delta < \frac{1}{2}$ *corresponds to the condition* $p > \frac{1}{2}$ *in* [6]. *If, on the other hand,* $\delta = 0$, *then we recover the deterministic complexity bound. If* $r(\epsilon_f) > 0$, *the quantity* γ *can be chosen to be some fixed constant, for example,* $\frac{1}{2}$. *This implies the condition that* $\delta < \frac{1}{2}(1 - \frac{1}{\sqrt{2}})$ *and the bound* (3.10) *is adjusted accordingly. We see that larger values of* γ *imply tighter bounds on* δ; *however, as will be shown in the next section, they allow the algorithm to achieve better accuracy for the same noise level* ϵ_f. *Thus, the constant* γ *simply serves as a means to highlight the trade-off between imposing smaller bounds on* δ *and achieving a smaller radius of convergence.*

4. Convergence analysis of the modified line search. In this section, we derive expected complexity bounds for the modified line search Algorithm 2.1, where the step size parameter is chosen using Algorithm 2.2.

We begin by stating the first condition on the gradient estimates which we use in our analysis,

$$\|g_k - \nabla\phi(x_k)\| \leq \theta\|\nabla\phi(x_k)\| \quad \text{for all} \quad k = 0, 1, 2, \ldots \tag{4.1}$$

for some $\theta \in [0, 1)$. This condition is referred to as a *norm condition* and was introduced and studied in [5] in the context of trust-region methods with inaccurate gradients. Note that this condition implies that g_k is a descent direction for the function ϕ. When unbiased stochastic estimators of $\nabla\phi(x)$ are available, g_k can be computed by averaging these estimators. If the variance of these estimators is bounded by $\mathcal{O}(\|\nabla\phi(x_k)\|^2)$, then condition (4.1) can be satisfied, with probability $1 - \delta$, by using a sufficiently large number of the estimators (batch size) to compute g_k. We chose to consider condition (4.1) because we are motivated by the specific setting where estimates g_k are computed via finite differences, interpolation, or smoothing [1, 2, 7, 8, 16].

In a more general stochastic setting, unless one knows $\|\nabla\phi(x_k)\|$, condition (4.1) is hard or impossible to verify or guarantee. A simple way of making condition (4.1) realizable is to replace $\|\nabla\phi(x_k)\|$ with ϵ, where ϵ is the desired convergence accuracy. However, if the cost of obtaining g_k that satisfies $\|g_k - \nabla\phi(x_k)\| \leq \theta\epsilon$ increases as ϵ decreases, replacing $\|\nabla\phi(x_k)\|$ by its global lower bound ϵ can lead to inefficient algorithms.

In the literature, a significant number of attempts to circumvent the aforementioned difficulties in the case of general stochastic gradient estimates have been made; see, e.g., [4, 6, 18]. In [4] a practical approach to estimate $\|\nabla\phi(x_k)\|$ is proposed and used to ensure that some approximation of (4.1) holds. In [6] and [18], (4.1) is replaced with a condition that, for some $\kappa \geq 0$,

$$\|g_k - \nabla\phi(x_k)\| \leq \kappa\alpha_k\|g_k\| \quad \text{for all} \quad k = 0, 1, 2, \ldots \tag{4.2}$$

holds with probability $1 - \delta$, and it is discussed how this condition can be ensured. Under this condition, expected complexity bounds are derived for a line search method that has access to deterministic function values in [6] and stochastic function values (with additional assumptions) in [18]. While this condition does not require the variance to diminish with $\|\nabla\phi(x_k)\|$, it may be hard or impossible to ensure when α_k is very small, due to the noise. Thus, we propose the following modification of this condition:

$$\|g_k - \nabla\phi(x_k)\| \leq \max\{\zeta\epsilon_g, \kappa\alpha_k\|g_k\|\} \quad \text{for all} \quad k = 0, 1, 2, \ldots, \tag{4.3}$$

where $\zeta > 1$ and $\epsilon_g \geq 0$ (we precisely define ϵ_g in section 4.3). We extend the analysis in [6] and derive complexity bounds based on (4.3) for our setting (i.e., noisy function evaluations).

In the remainder of this section, we present a convergence analysis for the generic line search algorithm (Algorithms 2.1 and 2.2). The analysis is an extension of the analysis presented in [6] to the case where functions are computed with noise (Assumption 1.3). We first consider the *norm condition* (4.1), and prove complexity guarantees for the special case where $d_k = -g_k$ (section 4.1) and general descent (section 4.2). We then prove similar results for condition (4.3) (section 4.3). For brevity we omit the results for general descent under condition (4.3) as these results are very similar to those for (4.1).

4.1. Convergence under condition (4.1). We use the following notion of *sufficiently accurate gradients*.

DEFINITION 4.1. *A sequence of random gradients* $\{G_k\}$ *is* $(1-\delta)$*-probabilistically "sufficiently accurate" for Algorithm* 2.1 *if there exists a constant* $\theta \in [0, \frac{1-c_1}{2-c_1})$*, such that the indicator variables*

$$I_k = \mathbb{1}\{\|G_k - \nabla\phi(X_k)\| \leq \theta\|\nabla\phi(X_k)\|\}$$

satisfy the submartingale condition

$$\mathbb{P}(I_k = 1 | \mathcal{F}_{k-1}^{G,\mathcal{E}}) \geq 1 - \delta$$

for all realizations of $\mathcal{F}_{k-1}^{G,\mathcal{E}}$*, where* $\mathcal{F}_{k-1}^{G,\mathcal{E}} = \sigma(G_0, \ldots, G_{k-1}, \mathcal{E}_{k-1})$ *is the* σ*-algebra generated by* $G_0, \ldots, G_{k-1}$ *and* $\mathcal{E}_{k-1}$*. Moreover, we say that iteration* k *is a* true *iteration if the event* $I_k = 1$ *occurs; otherwise the iteration is called* false*.*

For the remainder of this section, we make the following additional assumption.

ASSUMPTION 4.2 (sufficiently accurate gradients). *The sequence of random gradients* $\{G_k\}$ *generated by Algorithm* 2.1 *is* $(1-\delta)$*-probabilistically "sufficiently accurate" with* $\delta < \frac{1}{2} - \frac{\sqrt{\gamma}}{2}$ *for some* $\gamma \in (0, 1)$*.*

Equipped with the above definitions, assumptions, and theorems, we now provide convergence guarantees for the generic line search algorithm (Algorithms 2.1 and 2.2) for convex, strongly convex, and nonconvex objective functions. We remind the reader of the definition of the stopping time N_ϵ given in Definition 3.2.

For each *true* iteration (i.e., $I_k = 1$), we have

$$\|g_k - \nabla\phi(x_k)\| \leq \theta\|\nabla\phi(x_k)\|,$$

which implies, using the triangle inequality, that

$$\|g_k\| \geq (1-\theta)\|\nabla\phi(x_k)\|. \tag{4.4}$$

We now show that Assumption 3.3 is satisfied. To this end, for the three classes of functions, we show that there exists an upper bound $\bar{\alpha}$ on the step length parameter, and functions $h(\alpha)$ and $r(\epsilon_f)$ such that the assumption is true. First, we derive an expression for the constant $\bar{\alpha}$.

LEMMA 4.3. *Let Assumptions* 1.1 *and* 1.3 *hold. For every realization of Algorithm* 2.1*, if iteration* k *is* true *(i.e.,* $I_k = 1$*), and if*

$$\alpha_k \leq \bar{\alpha} = \frac{2(1 - 2\theta - c_1(1-\theta))}{L(1-\theta)}, \tag{4.5}$$

then (2.1) *holds. In other words, when* (4.5) *holds, any* true *iteration is also a* successful *iteration. Moreover, for every* true *and* successful *iteration,*

$$\phi(x_{k+1}) \leq \phi(x_k) - c_1\alpha_k(1-\theta)^2\|\nabla\phi(x_k)\|^2 + 4\epsilon_f. \tag{4.6}$$

Proof. By Assumption 1.1, we have

$$\phi(x_k - \alpha_k g_k) \leq \phi(x_k) - \alpha_k {g_k}^T \nabla\phi(x_k) + \frac{\alpha_k^2 L}{2}\|g_k\|^2.$$

Applying the Cauchy–Schwarz inequality and (4.1) and (4.4), for every *true* iteration

$$\begin{aligned}
\phi(x_k - \alpha_k g_k) &\leq \phi(x_k) - \alpha_k {g_k}^T \nabla\phi(x_k) + \frac{\alpha_k^2 L}{2}\|g_k\|^2 \\
&= \phi(x_k) - \alpha_k {g_k}^T(\nabla\phi(x_k) - g_k) - \alpha_k\left[1 - \frac{\alpha_k L}{2}\right]\|g_k\|^2 \\
&\leq \phi(x_k) + \alpha_k\|g_k\|\|\nabla\phi(x_k) - g_k\| - \alpha_k\left[1 - \frac{\alpha_k L}{2}\right]\|g_k\|^2 \\
&\leq \phi(x_k) + \frac{\alpha_k\theta}{1-\theta}\|g_k\|^2 - \alpha_k\left[1 - \frac{\alpha_k L}{2}\right]\|g_k\|^2 \\
&= \phi(x_k) - \alpha_k\left[\frac{1-2\theta}{1-\theta} - \frac{\alpha_k L}{2}\right]\|g_k\|^2.
\end{aligned}$$

By Assumption 1.3, we have

$$f(x_k - \alpha_k g_k, \xi) \leq f(x_k, \xi) - \alpha_k\left[\frac{1-2\theta}{1-\theta} - \frac{\alpha_k L}{2}\right]\|g_k\|^2 + 2\epsilon_f.$$

From this we conclude that (2.1) holds whenever

$$f(x_k, \xi) - \alpha_k\left[\frac{1-2\theta}{1-\theta} - \frac{\alpha_k L}{2}\right]\|g_k\|^2 + 2\epsilon_f \leq f(x_k, \xi) - c_1\alpha_k\|g_k\|^2 + 2\epsilon_f,$$

which is equivalent to (4.5). Therefore, using Assumption 1.3 and (4.4), for every *true* and *successful* iteration we have

$$\phi(x_{k+1}) \leq \phi(x_k) - c_1\alpha_k(1-\theta)^2\|\nabla\phi(x_k)\|^2 + 4\epsilon_f,$$

which completes the proof. ∎

We should mention that when the error in the gradient approximation is zero, i.e., $\theta = 0$, we recover the step size parameter condition from the deterministic setting. Moreover, when there is no noise in the function, i.e., $\epsilon_f = 0$, we recover the sufficient decrease condition of the deterministic gradient descent algorithm with an Armijo backtracking line search.

Next, we state and prove a result for the case of *false* and *successful* iterations.

LEMMA 4.4. *Let Assumption* 1.3 *hold. For every* false *and* successful *iteration of Algorithm* 2.1*, we have*

$$\phi(x_{k+1}) \leq \phi(x_k) - c_1\alpha_k\|g_k\|^2 + 4\epsilon_f.$$

Proof. The proof of this lemma is straightforward. For every *successful* iteration we have

$$f(x_{k+1},\xi) \leq f(x_k,\xi) - c_1\alpha_k\|g_k\|^2 + 2\epsilon_f.$$

Thus, by Assumption 1.3,

$$\phi(x_{k+1}) \leq \phi(x_k) - c_1\alpha_k\|g_k\|^2 + 4\epsilon_f,$$

which completes the proof. ∎

The result of Lemma 4.4 shows the amount of decrease in *false* and *successful* iterations. Note that the error term $4\epsilon_f$ illustrates that on *false* iterations the function value may increase and that the increase is related to the noise in the function values.

4.1.1. Convex functions. In this section, we analyze the expected complexity of Algorithm 2.1 in the case when ϕ is a convex function.

ASSUMPTION 4.5 (convexity and boundedness of iterates). *The function ϕ is convex and there exists a constant $D > 0$ such that*

$$\|x - x^\star\| \leq D \quad \text{for all } x \in \mathcal{U}, \tag{4.7}$$

where $x^\star$ is some global minimizer of ϕ (and $\phi^\star = \phi(x^\star)$) and the set $\mathcal{U}$ contains all iteration realizations.

This assumption may seem strong since it requires all iterates of the algorithm to remain in a bounded region. When the objective function is not allowed to increase, this assumption is simply ensured by assuming bounded level sets of $\phi(x)$. In the case of noisy function values in principle, iterates can wander out of a bounded region with some small probability (as this would require a large sequence of *false successful* iterations). Thus, ideally, we need to modify the algorithm to prevent it from going outside of some predefined bounded region, which is known to contain $x^\star$. Such modification is simple and our analysis will still apply, but with some notational complications. Therefore, we choose not to impose this modification explicitly. Note that we only use this assumption in the convex case and drop it in the strongly convex and nonconvex cases, and thus the nonconvex case convergence rates apply to the convex case without (4.7).

We bound the number of iterations taken by Algorithm 2.1 until $\phi(X_k) - \phi^\star \leq \epsilon$ occurs. Let

$$\Delta_k^\phi = \phi(X_k) - \phi^\star \quad \text{and} \quad Z_k = \frac{1}{\Delta_k^\phi} - \frac{1}{\Delta_0^\phi}. \tag{4.8}$$

By this definition, N_ϵ is the number of iterations taken until $Z_k \geq \frac{1}{\epsilon} - \frac{1}{\Delta_0^\phi} = Z_\epsilon$. Note that, due to the noise in the function evaluations, ϵ cannot be chosen to be arbitrarily small. We make an assumption on ϵ that explicitly defines the neighborhood of convergence.

ASSUMPTION 4.6 (neighborhood of convergence, convex case).

$$\epsilon^2 > \max\left\{\frac{8\epsilon_f LD^2}{\gamma c_1(1-\theta)(1-2\theta-c_1(1-\theta))}, 16\epsilon_f^2\right\},$$

with the same $\gamma \in (0,1)$ *as used in Assumption* 4.2.

REMARK 4.7. *We will show that the above assumption implies Assumption* 3.3(iv) *with the same constant* γ. *Hence here we see the direct connection between* γ *and the lower bound on* ϵ. *As discussed previously,* γ *can be chosen to be* $\frac{1}{2}$, *for example.*

By Lemma 4.3, whenever $\mathcal{A}_k \le \bar{\alpha}$, then every *true* iteration is also *successful*. We now show that on *true* and *successful* iterations, Z_k increases by at least some function $h(\mathcal{A}_k) - r(\epsilon_f)$ for all $k < N_\epsilon$.

LEMMA 4.8. *Let Assumptions* 1.3, 4.5, *and* 4.6 *hold, and consider any realization of Algorithm* 2.1. *For every iteration that is* true *and* successful, *we have*

$$z_{k+1} \ge z_k + \frac{c_1\alpha_k(1-\theta)^2}{4D^2} - \frac{4\epsilon_f}{\epsilon^2}.$$

Proof. By Assumption 4.5, for all $x, y \in \mathbb{R}^n$, we have

$$\phi(x) - \phi(y) \ge \nabla\phi(y)^T(x-y).$$

Thus, if $x = x^\star$ and $y = x_k$, we have

$$-\Delta_k^\phi = \phi^\star - \phi(x_k) \ge \nabla\phi(x_k)^T(x^\star - x_k) \ge -D\|\nabla\phi(x_k)\|,$$

where we used the Cauchy–Schwarz inequality and (4.7). Thus, when k is a *true* iteration, by (4.4) we have

$$\|g_k\|^2 \ge (1-\theta)^2\|\nabla\phi(x_k)\|^2 \ge \frac{(1-\theta)^2\left(\Delta_k^\phi\right)^2}{D^2}.$$

If k is also a *successful* iteration, then

$$\Delta_k^\phi - \Delta_{k+1}^\phi = \phi(x_k) - \phi(x_{k+1}) \ge c_1\alpha_k\|g_k\|^2 - 4\epsilon_f \ge \frac{c_1\alpha_k(1-\theta)^2\left(\Delta_k^\phi\right)^2}{D^2} - 4\epsilon_f,$$

and thus

$$\Delta_k^\phi + 4\epsilon_f - \Delta_{k+1}^\phi \ge \frac{c_1\alpha_k(1-\theta)^2\left(\Delta_k^\phi\right)^2}{D^2}.$$

Dividing by $\left(\Delta_{k+1}^\phi\right)\left(\Delta_k^\phi + 4\epsilon_f\right)$,

$$\frac{1}{\Delta_{k+1}^\phi} - \frac{1}{\Delta_k^\phi + 4\epsilon_f} \ge \frac{c_1\alpha_k(1-\theta)^2\left(\Delta_k^\phi\right)^2}{D^2\left(\Delta_{k+1}^\phi\right)\left(\Delta_k^\phi + 4\epsilon_f\right)}. \tag{4.9}$$

The left-hand side of (4.9) can be bounded by

$$\begin{aligned}\frac{1}{\Delta_{k+1}^{\phi}}-\frac{1}{\Delta_k^{\phi}+4\epsilon_f} &= \frac{1}{\Delta_{k+1}^{\phi}}-\frac{1}{\Delta_k^{\phi}}+\frac{1}{\Delta_k^{\phi}}-\frac{1}{\Delta_k^{\phi}+4\epsilon_f}\\ &= \frac{1}{\Delta_{k+1}^{\phi}}-\frac{1}{\Delta_k^{\phi}}+\frac{4\epsilon_f}{\left(\Delta_k^{\phi}\right)\left(\Delta_k^{\phi}+4\epsilon_f\right)}\\ &\le \frac{1}{\Delta_{k+1}^{\phi}}-\frac{1}{\Delta_k^{\phi}}+\frac{4\epsilon_f}{\epsilon^2},\end{aligned}$$

where the last inequality holds since $\Delta_k^{\phi}+4\epsilon_f \ge \Delta_k^{\phi} \ge \epsilon$. The right-hand side of (4.9) can be bounded by

$$\begin{aligned}\frac{c_1\alpha_k(1-\theta)^2\left(\Delta_k^{\phi}\right)^2}{D^2\left(\Delta_{k+1}^{\phi}\right)\left(\Delta_k^{\phi}+4\epsilon_f\right)} &\ge \frac{c_1\alpha_k(1-\theta)^2\left(\Delta_k^{\phi}\right)^2}{D^2\left(\Delta_k^{\phi}+4\epsilon_f\right)^2}\\ &\ge \frac{c_1\alpha_k(1-\theta)^2}{4D^2},\end{aligned}$$

where the first inequality holds since $\Delta_{k+1}^{\phi} \le \Delta_k^{\phi}+4\epsilon_f$, and the second due to the fact that $\Delta_k^{\phi} \ge \epsilon > 4\epsilon_f$ (due to Assumption 4.6) and thus $\frac{\Delta_k^{\phi}}{\Delta_k^{\phi}+4\epsilon_f} \ge \frac{1}{2}$.

Therefore, we have

$$\left(\frac{1}{\Delta_{k+1}^{\phi}}-\frac{1}{\Delta_0^{\phi}}\right)-\left(\frac{1}{\Delta_k^{\phi}}-\frac{1}{\Delta_0^{\phi}}\right)=\frac{1}{\Delta_{k+1}^{\phi}}-\frac{1}{\Delta_k^{\phi}} \ge \frac{c_1\alpha_k(1-\theta)^2}{4D^2}-\frac{4\epsilon_f}{\epsilon^2},$$

which completes the proof. ∎

We now bound the amount of increase in *false* and *successful* iterations.

LEMMA 4.9. *Let Assumptions* 1.3, 4.5*, and* 4.6 *hold, and consider any realization of Algorithm* 2.1*. For every iteration that is* false *and* successful*, we have*

$$z_{k+1} \ge z_k - \frac{4\epsilon_f}{\epsilon^2}.$$

Proof. For every *false* and *successful* iteration, by Lemma 4.4 we have

$$\begin{aligned}\phi(x_{k+1}) &\le \phi(x_k) - c_1\alpha_k\|g_k\|^2 + 4\epsilon_f\\ &\le \phi(x_k) + 4\epsilon_f.\end{aligned}$$

The rest of the proof is essentially a simplified version of the proof of Lemma 4.8, where the right-hand side in (4.9) is simply replaced with 0. ∎

Let

$$h(\alpha) = \frac{c_1\alpha(1-\theta)^2}{4D^2} \qquad \text{and} \qquad r(\epsilon_f) = \frac{4\epsilon_f}{\epsilon^2}. \tag{4.10}$$

By Lemmas 4.3, 4.8, and 4.9 and Assumption 4.6, for any realization of Algorithm 2.1 (which specifies the sequence $\{\alpha_k, z_k\}$) and $k < N_\epsilon$, we have the following:

1. (Lemma 4.8) If k is a *true* and *successful* iteration, then

$$z_{k+1} \ge z_k + h(\alpha_k) - r(\epsilon_f) \quad \text{and} \quad \alpha_{k+1} = \tau^{-1}\alpha_k.$$

2. (Lemma 4.3) If $\alpha_k \leq \bar{\alpha}$ and iteration k is *true*, then it is also *successful*.
3. (Lemma 4.9) If k is a *false* and *successful* iteration, then

$$z_{k+1} \geq z_k - r(\epsilon_f).$$

4. (Assumption 4.6) $\frac{r(\epsilon_f)}{h(\bar{\alpha})} < \gamma$ for $\gamma \in (0,1)$.

Hence, Assumption 3.3 holds, with $\bar{\alpha} > 0$ defined in (4.5), and with $h(\mathcal{A}_k)$ and $r(\epsilon_f)$ defined in (4.10).

We now use Theorem 3.13 and the definitions of $\bar{\alpha}$, $h(\bar{\alpha})$, $r(\epsilon_f)$, and Z_ϵ to bound $\mathbb{E}[N_\epsilon]$.

THEOREM 4.10. *Let Assumptions* 1.1, 1.3, 4.2, *and* 4.5 *hold. Moreover, let Assumption* 4.6 *hold, i.e.,*

$$\epsilon^2 > \max\left\{\frac{8\epsilon_f L D^2}{\gamma c_1(1-\theta)(1-2\theta-c_1(1-\theta))}, 16\epsilon_f^2\right\},$$

with the same $\gamma \in (0,1)$ *as used in Assumption* 4.2. *Then the expected number of iterations that Algorithm* 2.1 *takes until* $\phi(X_k) - \phi^\star \leq \epsilon$ *occurs is bounded as follows:*

$$\mathbb{E}[N_\epsilon] \leq \frac{2(1-\delta)}{(1-2\delta)^2 - \gamma}\left[M\left(\frac{1}{\epsilon} - \frac{1}{\phi(x_0)-\phi^\star}\right) + (1-\gamma)\log_\tau\left(\frac{\bar{\alpha}}{\alpha_0}\right)\right],$$

where $M = \frac{4LD^2}{c_1(1-\theta)(1-2\theta-c_1(1-\theta))}$.

REMARK 4.11. *If* $\delta = \theta = \epsilon_f = 0$, *our algorithm reduces to a deterministic line search algorithm with exact function evaluations and gradients. When* $\epsilon_f = 0$, γ *can be chosen arbitrarily small, and the lower bound on* ϵ *is* 0. *Notice that the complexity bound has two components: the first component* $\frac{8D^2L}{c_1(1-c_1)\epsilon}$ *achieves its minimum value,* $\frac{32D^2L}{\epsilon}$, *for* $c_1 = 1/2$ *and is similar to the complexity bounds of the fixed step gradient descent method for convex functions, and the second term* $\log_\tau\left(\frac{2(1-c_1)}{\alpha_0 L}\right)$ *bounds the total number of unsuccessful iterations, on which* α_k *is reduced.*

4.1.2. Strongly convex functions. In this section, we analyze the expected complexity of Algorithm 2.1 in the case when ϕ is a strongly convex function.

ASSUMPTION 4.12 (strong convexity of $\boldsymbol{\phi}$). *There exists a positive constant* μ *such that*

$$\phi(x) \geq \phi(y) + \nabla\phi(y)^T(x-y) + \frac{\mu}{2}\|x-y\|^2 \quad \textit{for all } x, y \in \mathbb{R}^n.$$

Under Assumption 4.12, let $\phi^\star = \phi(x^\star)$, where $x^\star$ is the minimizer of ϕ.

Recall the definition of Δ_k^ϕ (4.8). In this setting, we bound the number of iterations taken by Algorithm 2.1 until $\Delta_k^\phi \leq \epsilon$ occurs. However, in this setting Z_k is defined as $Z_k = \log\left(\frac{1}{\Delta_k^\phi}\right)$ and the resulting complexity bound is logarithmic in $\frac{1}{\epsilon}$. Note that, similar to the convex case, due to the noise in the function evaluations, ϵ cannot be chosen to be arbitrarily small. We give a precise lower bound on ϵ, and thus explicitly derive a bound for the neighborhood of convergence.

ASSUMPTION 4.13 (neighborhood of convergence, strongly convex case).

$$\epsilon > \frac{4\epsilon_f}{\left(1 - \frac{2\mu c_1(1-\theta)(1-2\theta-c_1(1-\theta))}{L}\right)^{-\gamma} - 1},$$

with the same $\gamma \in (0,1)$ *as used in Assumption* 4.2.

REMARK 4.14. *The above assumption again implies Assumption* 3.3(iv) *with the same constant* γ*, which connects* ϵ_f *to the lower bound on* ϵ*. Again,* γ *can be chosen to be* $\frac{1}{2}$*, for simplicity.*

By Lemma 4.3, whenever $\mathcal{A}_k \leq \bar{\alpha}$, then every *true* iteration is also *successful*. We now show that on *true* and *successful* iterations, Z_k increases by at least some function $h(\mathcal{A}_k) - r(\epsilon_f)$ for all $k < N_\epsilon$.

LEMMA 4.15. *Let Assumptions* 1.3, 4.12*, and* 4.13 *hold, and consider any realization of Algorithm* 2.1*. For every iteration that is* true *and* successful*, we have*

$$z_{k+1} \geq z_k - \log\left(1 - \mu c_1 \alpha_k (1-\theta)^2\right) - \log\left(1 + \frac{4\epsilon_f}{\epsilon}\right).$$

Proof. Assumption 4.12 implies that ($x = x_k$ and $y = x^\star$)

$$\phi(x_k) - \phi^\star \leq \frac{1}{2\mu}\|\nabla\phi(x_k)\|^2;$$

see [15, Theorem 2.1.10]. Equivalently, using (4.4),

$$\|g_k\|^2 \geq (1-\theta)^2\|\nabla\phi(x_k)\|^2 \geq 2\mu(1-\theta)^2(\phi(x_k) - \phi^\star).$$

By (4.6), for every *true* and *successful* iteration we have

$$\begin{aligned} \phi(x_{k+1}) &\leq \phi(x_k) - c_1\alpha_k(1-\theta)^2\|\nabla\phi(x_k)\|^2 + 4\epsilon_f \\ &\leq \phi(x_k) - 2\mu c_1\alpha_k(1-\theta)^2(\phi(x_k) - \phi^\star) + 4\epsilon_f, \end{aligned} \tag{4.11}$$

and thus

$$\phi(x_{k+1}) - \phi^\star \leq \left(1 - 2\mu c_1\alpha_k(1-\theta)^2\right)(\phi(x_k) - \phi^\star) + 4\epsilon_f.$$

Since we have that $\phi(x_k) - \phi^\star \geq \epsilon$,

$$\begin{aligned} \phi(x_{k+1}) - \phi^\star &\leq \left(1 - 2\mu c_1\alpha_k(1-\theta)^2\right)(\phi(x_k) - \phi^\star) + 4\epsilon_f \\ &\leq \left(1 - 2\mu c_1\alpha_k(1-\theta)^2\right)(\phi(x_k) - \phi^\star) + \frac{4\epsilon_f}{\epsilon}(\phi(x_k) - \phi^\star) \\ &= \left(1 - 2\mu c_1\alpha_k(1-\theta)^2 + \frac{4\epsilon_f}{\epsilon}\right)(\phi(x_k) - \phi^\star). \end{aligned}$$

Thus, using the definition of Δ_k^ϕ, we have

$$\Delta_{k+1}^\phi \leq \left(1 - 2\mu c_1\alpha_k(1-\theta)^2 + \frac{4\epsilon_f}{\epsilon}\right)\Delta_k^\phi.$$

Since $\epsilon > 4\epsilon_f$ (due to Assumption 4.13), we have

$$\begin{aligned} \Delta_{k+1}^\phi &\leq \left(1 - 2\mu c_1\alpha_k(1-\theta)^2 + \frac{4\epsilon_f}{\epsilon}\right)\Delta_k^\phi \\ &\leq \left(1 - \mu c_1\alpha_k(1-\theta)^2 - \frac{4\epsilon_f}{\epsilon}\mu c_1\alpha_k(1-\theta)^2 + \frac{4\epsilon_f}{\epsilon}\right)\Delta_k^\phi \\ &= \left(1 - \mu c_1\alpha_k(1-\theta)^2\right)\left(1 + \frac{4\epsilon_f}{\epsilon}\right)\Delta_k^\phi. \end{aligned}$$

Notice that since $\left(1+\frac{4\epsilon_f}{\epsilon}\right) > 0$, $\Delta_k^\phi > 0$, and $\Delta_{k+1}^\phi \geq 0$, this implies that $1-\mu c_1\alpha_k(1-\theta)^2 \geq 0$. Now taking the inverse and then the log of both sides and adding $\log \Delta_0^\phi$, we have

$$\log\left(\frac{\Delta_0^\phi}{\Delta_{k+1}^\phi}\right) \geq \log\left(\frac{\Delta_0^\phi}{\Delta_k^\phi}\right) - \log\left(1-\mu c_1\alpha_k(1-\theta)^2\right) - \log\left(1+\frac{4\epsilon_f}{\epsilon}\right),$$

which completes the proof. ∎

We note here that $1-\mu c_1\alpha_k(1-\theta)^2 \geq 0$ holds for all $\alpha_k \leq \bar{\alpha}$ due to the constraint $\theta \in \left[0, \frac{1-c_1}{2-c_1}\right)$.

We now bound the amount of increase in *false* and *successful* iterations.

LEMMA 4.16. *Let Assumptions* 1.3, 4.12*, and* 4.13 *hold, and consider any realization of Algorithm* 2.1*. For every iteration that is* false *and* successful*, we have*

$$z_{k+1} \geq z_k - \log\left(1+\frac{4\epsilon_f}{\epsilon}\right).$$

Proof. For every *false* and *successful* iteration, by Lemma 4.4 we have

$$\phi(x_{k+1}) \leq \phi(x_k) - c_1\alpha_k\|g_k\|^2 + 4\epsilon_f.$$

The rest of the proof is essentially a simplification of the proof of Lemma 4.15 with the middle term of the right-hand side of (4.11) replaced by 0. ∎

Let

$$h(\alpha) = -\log(1-\mu c_1(1-\theta)^2\alpha) \quad \text{and} \quad r(\epsilon_f) = \log\left(1+\frac{4\epsilon_f}{\epsilon}\right). \tag{4.12}$$

By Lemmas 4.3, 4.15, and 4.16 and Assumption 4.13, for any realization of Algorithm 2.1 (which specifies the sequence $\{\alpha_k, z_k\}$) and $k < N_\epsilon$, we have the following:

1. (Lemma 4.15) If k is a *true* and *successful* iteration, then
$$z_{k+1} \geq z_k + h(\alpha_k) - r(\epsilon_f) \quad \text{and} \quad \alpha_{k+1} = \tau^{-1}\alpha_k.$$
2. (Lemma 4.3) If $\alpha_k \leq \bar{\alpha}$ and iteration k is *true*, then it is also *successful*.
3. (Lemma 4.16) If k is a *false* and *successful* iteration, then
$$z_{k+1} \geq z_k - \log\left(1+\frac{4\epsilon_f}{\epsilon}\right).$$
4. (Assumption 4.13) $\frac{r(\epsilon_f)}{h(\bar{\alpha})} < \gamma$ for some $\gamma \in (0,1)$.

Hence, Assumption 3.3 holds, with $\bar{\alpha} > 0$ defined in (4.5), and $h(\mathcal{A}_k)$ and $r(\epsilon_f)$ defined in (4.12).

We now use Theorem 3.13 and the definitions of $\bar{\alpha}$, $h(\bar{\alpha})$, $r(\epsilon_f)$, and Z_ϵ to bound $\mathbb{E}[N_\epsilon]$.

THEOREM 4.17. *Let Assumptions* 1.1, 1.3, 4.2, *and* 4.12 *hold. Moreover, let Assumption* 4.13 *hold, i.e.,*

$$\epsilon > \frac{4\epsilon_f}{\left(1 - \frac{2\mu c_1(1-\theta)(1-2\theta-c_1(1-\theta))}{L}\right)^{-\gamma} - 1},$$

with the same $\gamma \in (0,1)$ *as used in Assumption* 4.2. *Then the expected number of iterations that Algorithm* 2.1 *takes until* $\phi(X_k) - \phi^\star \le \epsilon$ *occurs is bounded as follows:*

$$\mathbb{E}[N_\epsilon] \le \frac{2(1-\delta)}{(1-2\delta)^2 - \gamma}\left[2\log_{1/M}\left(\frac{\phi(x_0)-\phi^\star}{\epsilon}\right) + (1-\gamma)\log_\tau\left(\frac{\bar\alpha}{\alpha_0}\right)\right],$$

where $M = 1 - \frac{2\mu c_1(1-\theta)(1-2\theta-c_1(1-\theta))}{L}$.

REMARK 4.18. *Again, if* $\delta = \theta = \epsilon_f = 0$, *our algorithm reduces to a deterministic line search algorithm with exact function evaluations and gradients. The complexity bound has two components:* $4\log_{1/M}\left(\frac{1}{\epsilon}\right)$, *where* $M = 1 - \frac{4\mu c_1(1-c_1)}{L}$ *achieves its minimum value,* $1 - \frac{\mu}{L}$, *for* $c_1 = 1/2$ *and is similar to complexity bounds of the fixed step gradient descent method for strongly convex functions, and the second term again is the bound on the total number of unsuccessful iterations.*

4.1.3. Nonconvex functions. In this section, we analyze the expected complexity of Algorithm 2.1 in the case when ϕ is a nonconvex function. Again, we first specify the neighborhood of convergence. In this setting $Z_k = \phi(X_0) - \phi(X_k)$.

ASSUMPTION 4.19 (neighborhood of convergence, nonconvex case).

$$\epsilon^2 > \frac{2\epsilon_f L}{\gamma c_1(1-\theta)(1-2\theta-c_1(1-\theta))},$$

with the same $\gamma \in (0,1)$ *as used in Assumption* 4.2.

REMARK 4.20. *The role of* γ *is the same as in the convex and strongly convex cases.*

Let

$$h(\alpha) = c_1\alpha(1-\theta)^2\|\nabla\phi(x_k)\|^2 \qquad \text{and} \qquad r(\epsilon_f) = 4\epsilon_f. \tag{4.13}$$

By Lemmas 4.3 and 4.4 and Assumption 4.19, for any realization of Algorithm 2.1 (which specifies the sequence $\{\alpha_k, z_k\}$) and $k < N_\epsilon$, we have the following:

1. (Lemma 4.3) If k is a *true* and *successful* iteration, then
$$z_{k+1} \ge z_k + h(\alpha_k) - r(\epsilon_f) \quad \text{and} \quad \alpha_{k+1} = \tau^{-1}\alpha_k$$
2. (Lemma 4.3) If $\alpha_k \le \bar\alpha$ and iteration k is *true*, then it is also *successful*.
3. (Lemma 4.4) If k is a *false* and *successful* iteration, then
$$z_{k+1} \ge z_k - 4\epsilon_f.$$
4. (Assumption 4.19) $\frac{r(\epsilon_f)}{h(\bar\alpha)} < \gamma$ for some $\gamma \in (0,1)$.

Hence, Assumption 3.3 holds, with $\bar\alpha > 0$ defined in (4.5), and $h(\mathcal{A}_k)$ and $r(\epsilon_f)$ defined in (4.13).

We now use Theorem 3.13 and the definitions of $\bar\alpha$, $h(\bar\alpha)$, $r(\epsilon_f)$, and Z_ϵ to bound $\mathbb{E}[N_\epsilon]$.

THEOREM 4.21. *Let Assumptions* 1.1, 1.2, 1.3, *and* 4.2 *hold. Moreover, let Assumption* 4.19 *hold, i.e.,*

$$\epsilon^2 > \frac{2\epsilon_f L}{\gamma c_1(1-\theta)(1-2\theta-c_1(1-\theta))},$$

with the same $\gamma \in (0,1)$ *as used in Assumption* 4.2. *Then the expected number of iterations that Algorithm* 2.1 *takes until* $\|\nabla\phi(X_k)\| \leq \epsilon$ *occurs is bounded as follows:*

$$\mathbb{E}[N_\epsilon] \leq \frac{2(1-\delta)}{(1-2\delta)^2-\gamma}\left[\frac{M}{\epsilon^2} + (1-\gamma)\log_\tau\left(\frac{\bar{\alpha}}{\alpha_0}\right)\right],$$

where $M = \frac{(\phi(x_0)-\hat{\phi})L}{c_1(1-\theta)(1-2\theta-c_1(1-\theta))}$.

REMARK 4.22. *Again, if* $\delta = \theta = \epsilon_f = 0$, *our algorithm reduces to a deterministic line search with the exact gradients. The complexity bound has two components:* $\frac{2M}{\epsilon^2}$, *where* $M = \frac{(\phi(x_0)-\hat{\phi})L}{c_1(1-c_1)}$ *achieves its minimum value,* $4(f(x_0)-\hat{f})L$, *for* $c_1 = 1/2$ *and is similar to complexity bounds of the fixed step gradient descent for nonconvex functions, and the second term, as before, is the bound on the total number of unsuccessful iterations.*

4.2. General descent. For simplicity, in the analysis of the previous sections we assumed that the search direction at every iteration was defined as $d_k = -g_k$. Here, we show how our analysis can be extended to account for more general search direction, e.g., the quasi-Newton search direction where $d_k = -H_k g_k$ [17], provided the search directions satisfy, together with (4.1), the following conditions:

- There exists a constant $\beta > 0$, such that

$$\frac{d_k^T g_k}{\|d_k\|\|g_k\|} \leq -\beta \qquad \text{for all } k. \tag{4.14}$$

- There exist constants $\kappa_1, \kappa_2 > 0$, such that

$$\kappa_1\|g_k\| \leq \|d_k\| \leq \kappa_2\|g_k\| \qquad \text{for all } k. \tag{4.15}$$

Of course, in this setting, the modified line search would be given by (2.1), and the convergence analysis would have dependence on β, κ_1, and κ_2.

All we need to do is derive an expression for $\bar{\alpha}$ for the general search direction case and prove analogues of Lemmas 4.3 and 4.4. First, we change the bound on θ in Definition 4.1. In particular we will require that $\theta \in \left[0, \frac{(1-c_1)\beta}{1+(1-c_1)\beta}\right)$. Now we can prove the following lemma.

LEMMA 4.23. *Let Assumption* 1.1 *hold. For every realization of Algorithm* 2.1, *if iteration* k *is* true *(i.e.,* $I_k = 1$*), and if*

$$\alpha_k \leq \bar{\alpha} = \frac{2}{L\kappa_2}\left[\frac{(1-c_1)(1-\theta)\beta-\theta}{1-\theta}\right], \tag{4.16}$$

then (2.1) *holds. In other words, when* (4.16) *holds, any* true *iteration is also a* successful *iteration. Moreover, for every* true *and* successful *iteration,*

$$\phi(x_{k+1}) \leq \phi(x_k) - c_1\alpha_k\beta\kappa_1(1-\theta)^2\|\nabla\phi(x_k)\|^2 + 4\epsilon_f.$$

Proof. The proof is very similar to that of Lemma 4.3. First, from Assumption 1.1, we have

$$\phi(x_{k+1}) \le \phi(x_k) + \alpha_k d_k^T \nabla\phi(x_k) + \frac{L}{2}\|\alpha_k d_k\|^2.$$

Applying the Cauchy–Schwarz inequality and (4.1) and (4.4), for every *true* iteration we have

$$\begin{aligned}
\phi(x_k + \alpha_k d_k) &\le \phi(x_k) + \alpha_k d_k^T \nabla\phi(x_k) + \frac{\alpha_k^2 L}{2}\|d_k\|^2 \\
&= \phi(x_k) + \alpha_k d_k^T (\nabla\phi(x_k) - g_k) + \alpha_k d_k^T g_k + \frac{\alpha_k^2 L}{2}\|d_k\|^2 \\
&\le \phi(x_k) + \alpha_k \|d_k\|\|\nabla\phi(x_k) - g_k\| + \alpha_k d_k^T g_k) + \frac{\alpha_k^2 L}{2}\|d_k\|^2 \\
&\le \phi(x_k) + \frac{\alpha_k \theta}{1-\theta}\|d_k\|\|g_k\| + \alpha_k d_k^T g_k + \frac{\alpha_k^2 L \kappa_2}{2}\|d_k\|\|g_k\| \\
&\le \phi(x_k) + \alpha_k d_k^T g_k + \alpha_k \left[\frac{\theta}{1-\theta} + \frac{\alpha_k L \kappa_2}{2}\right]\|d_k\|\|g_k\|.
\end{aligned}$$

Now, using Assumption 1.3, we have

$$f(x_k + \alpha_k d_k, \xi) \le f(x_k, \xi) + \alpha_k d_k^T g_k + \alpha_k \left[\frac{\theta}{1-\theta} + \frac{\alpha_k L \kappa_2}{2}\right]\|d_k\|\|g_k\| + 2\epsilon_f.$$

From this we conclude that (2.1) holds whenever

$$\begin{aligned}
&f(x_k, \xi) + \alpha_k d_k^T g_k + \alpha_k \left[\frac{\theta}{1-\theta} + \frac{\alpha_k L \kappa_2}{2}\right]\|d_k\|\|g_k\| + 2\epsilon_f \\
&\quad \le f(x_k, \xi) + c_1 \alpha_k d_k^T g_k + 2\epsilon_f,
\end{aligned}$$

or equivalently, since $\alpha_k > 0$,

$$\left[\frac{\theta}{1-\theta} + \frac{\alpha_k L \kappa_2}{2}\right]\|d_k\|\|g_k\| \le -(1-c_1) d_k^T g_k.$$

Using (4.14), the above expression holds whenever α_k satisfies (4.16). Therefore, using Assumption 1.3, (4.15), and (4.4), for every *true* and *successful* iteration we have

$$\phi(x_{k+1}) \le \phi(x_k) - c_1 \alpha_k \beta \kappa_1 (1-\theta)^2 \|\nabla\phi(x_k)\|^2 + 4\epsilon_f,$$

which completes the proof. ∎

Next, we state and prove a result for the case of *false* and *successful* iterations.

LEMMA 4.24. *For every* false *and* successful *iteration of Algorithm* 2.1 *we have*

$$\phi(x_{k+1}) \le \phi(x_k) - c_1 \beta \alpha_k \kappa_1 \|g_k\|^2 + 4\epsilon_f.$$

Proof. For every *successful* iteration we have

$$f(x_{k+1}, \xi) \le f(x_k, \xi) + c_1 \alpha_k d_k^T g_k + 2\epsilon_f.$$

Thus, by Assumption 1.3, (4.15), and (4.4)

$$\begin{aligned}
\phi(x_{k+1}) &\le \phi(x_k) + c_1 \alpha_k d_k^T g_k + 4\epsilon_f \\
&\le \phi(x_k) - c_1 \alpha_k \beta \|d_k\|\|g_k\| + 4\epsilon_f \\
&\le \phi(x_k) - c_1 \alpha_k \beta \kappa_1 \|g_k\|^2 + 4\epsilon_f,
\end{aligned}$$

which is a repetition of the last part of the proof of Lemma 4.23. ∎

The rest of the analysis (deriving expected complexity bounds) applies almost without change, taking into account the influence of the constants β, κ_1, and κ_2.

4.3. Convergence under condition (4.3). In this section we demonstrate how our analysis can be extended to a different setting in terms of gradient estimate computations. To avoid introducing new notation, we will keep the discussion at a high level, which will hopefully be clear to the reader. The precise derivations in this sections are straightforward extensions of the derivations above.

As we have pointed out before, the key condition (4.1) can be satisfied by various gradient approximation schemes discussed in [2]. However, all these schemes require $\mathcal{O}(n)$ function evaluations to obtain g_k that satisfies (4.1). This can be expensive in a high-dimensional setting. On the other hand, in many applications a stochastic estimate of $\nabla\phi(x)$ may be directly available, and thus g_k can be computed by a sample averaging scheme. Since we assume that the function values are computed with noise, we cannot assume that these stochastic estimates are unbiased. However, as in the case of the function noise, we can assume that this bias is bounded.

ASSUMPTION 4.25 (biased gradient estimates). *For each x, we have an ability to compute a random vector $h(x,\xi)$, which is a (possibly) biased estimate of $\nabla\phi(x_k)$, and the bias is bounded by a known constant ϵ_g, i.e., for all x*

$$\|\mathbb{E}[h(x,\xi)] - \nabla\phi(x)\| \leq \epsilon_g,$$

where the expectation is over random variable ξ.

Thus, for any $\zeta > 1$, by averaging a sufficiently large number of samples $h(x,\xi)$ we can compute a (random) g such that $\|g - \nabla\phi(x)\| \leq \zeta\epsilon_g$, with sufficiently high probability. On the other hand, without knowing $\|\nabla\phi(x_k)\|$ we cannot ensure (4.1). Here, we present the outline of the analysis of our modified line search method where (4.1) is replaced with condition

$$\|g_k - \nabla\phi(x_k)\| \leq \max\{\zeta\epsilon_g, \kappa\alpha_k\|g_k\|\}$$

for some $\zeta > 1$ and $\kappa \geq 0$. Essentially, we want to relax (4.1) as long as $\kappa\alpha_k\|g_k\|$ is not so small that $\|g_k - \nabla\phi(x_k)\| \leq \kappa\alpha_k\|g_k\|$ cannot be enforced with sufficiently high probability. When this happens, we want (4.1) to hold, which we can ensure by $\|g_k - \nabla\phi(x_k)\| \leq \zeta\epsilon_g$, as long as $\|\nabla\phi(x_k)\| > \frac{\zeta\epsilon_g}{\theta}$. Thus we need to add this lower bound on the gradient to our definition of the stopping time.

DEFINITION 4.26.

- *If ϕ is convex or strongly convex: N_ϵ is the number of iterations required until either $\phi(X_k) - \phi^\star \leq \epsilon$ or $\|\nabla\phi(x_k)\| \leq \frac{\zeta\epsilon_g}{\theta}$ occurs for the first time. Note that $\phi^\star = \phi(x^\star)$, where $x^\star$ is a global minimizer of ϕ.*
- *If ϕ is nonconvex: N_ϵ is the number of iterations required until $\|\nabla\phi(X_k)\| \leq \max\{\epsilon, \frac{\zeta\epsilon_g}{\theta}\}$ occurs for the first time.*

For brevity, in this section we do not derive all the results, or state all the intermediate lemmas, but rather state the key results, without proof. We first present the analogue of Definition 4.1 where (4.1) is replaced with (4.3).

DEFINITION 4.27. *A sequence of random gradients $\{G_k\}$ is $(1-\delta)$-probabilistically "sufficiently accurate" for Algorithm 2.1 if there exist constants $\zeta > 1$ and $\kappa \geq 0$, such that the indicator variables*

$$I_k = \mathbb{1}\{\|G_k - \nabla\phi(X_k)\| \leq \max\{\zeta\epsilon_g, \kappa\mathcal{A}_k\|G_k\|\}\}$$

satisfy the submartingale condition

$$\mathbb{P}(I_k = 1 | \mathcal{F}_{k-1}^{G,\mathcal{E}}) \geq 1 - \delta$$

for all realizations of $\mathcal{F}_{k-1}^{G,\mathcal{E}}$*, where* $\mathcal{F}_{k-1}^{G,\mathcal{E}} = \sigma(G_0, \ldots, G_{k-1}, \mathcal{E}_{k-1})$ *is the* σ*-algebra generated by* $G_0, \ldots, G_{k-1}$ *and* $\mathcal{E}_{k-1}$ *for all realizations. Moreover, we say that iteration* k *is a* true *iteration if the event* $I_k = 1$ *occurs; otherwise the iteration is called* false.

We assume (as was done in section 4.1) that Assumption 4.2 holds for Definition 4.27. In order to prove expected complexity bounds under (4.2), we make the following minor modification to Algorithm 2.2. When the step is successful, $\alpha_{k+1} = \min\{\tau^{-1}\alpha_k, \alpha_{\max}\}$, where $\alpha_{\max} > 0$.

LEMMA 4.28. *Let Assumptions* 1.1 *and* 1.3 *hold. For every realization of Algorithm* 2.1*, if iteration* k *is* true *(i.e.,* $I_k = 1$*), and if*

$$\alpha_k \leq \bar{\alpha} = \min\left\{ \frac{2(1 - 2\theta - c_1(1-\theta))}{L(1-\theta)}, \frac{2(1-c_1)}{L + 2\kappa} \right\}, \tag{4.17}$$

then (2.1) *holds. In other words, when* (4.17) *holds, any* true *iteration is also a* successful *iteration. Moreover, for every* true *and* successful *iteration,*

$$\phi(x_{k+1}) \leq \phi(x_k) - c_1\alpha_k \min\left\{ (1-\theta)^2, \frac{1}{(1+\kappa\alpha_{\max})^2} \right\} \|\nabla\phi(x_k)\|^2 + 4\epsilon_f. \tag{4.18}$$

Furthermore, for every false *and* successful *iteration of Algorithm* 2.1*, we have*

$$\phi(x_{k+1}) \leq \phi(x_k) - c_1\alpha_k \|g_k\|^2 + 4\epsilon_f.$$

We should note that if g_k is the true gradient, we recover the step size parameter condition from the deterministic setting.

We now present the complexity bounds for condition (4.3) for convex (Theorem 4.29), strongly convex (Theorem 4.30), and nonconvex (Theorem 4.32) functions.

THEOREM 4.29. *Let Assumptions* 1.1, 1.3, 4.25, 4.2*, and* 4.5 *hold. Moreover, let Assumption* 4.6 *hold, i.e.,*

$$\epsilon^2 > \max\left\{ \frac{8\epsilon_f D^2}{\gamma c_1 \min\left\{ \frac{(1-\theta)(1-2\theta-c_1(1-\theta))}{L}, \frac{1-c_1}{(L+2\kappa)(1+\kappa\alpha_{\max})^2} \right\}}, 16\epsilon_f^2 \right\},$$

with the same $\gamma \in (0,1)$ *as used in Assumption* 4.2*. Then the expected number of iterations that Algorithm* 2.1 *takes until* $\phi(X_k) - \phi^\star \leq \epsilon$ *or* $\|\nabla\phi(X_k)\| \leq \frac{\zeta\epsilon_g}{\theta}$ *occurs is bounded as follows:*

$$\mathbb{E}[N_\epsilon] \leq \frac{2(1-\delta)}{(1-2\delta)^2 - \gamma} \left[M\left(\frac{1}{\epsilon} - \frac{1}{\phi(x_0) - \phi^\star} \right) + (1-\gamma)\log_\tau\left(\frac{\bar{\alpha}}{\alpha_0} \right) \right],$$

where $M = \frac{4D^2}{c_1 \min\left\{ \frac{(1-\theta)(1-2\theta-c_1(1-\theta))}{L}, \frac{1-c_1}{(L+2\kappa)(1+\kappa\alpha_{\max})^2} \right\}}$.

THEOREM 4.30. *Let Assumptions* 1.1, 1.3, 4.25, 4.2, *and* 4.12 *hold. Moreover, let Assumption* 4.13 *hold, i.e.,*

$$\epsilon > \max\left\{ \frac{4\epsilon_f}{\left(1 - 2\mu c_1 \min\left\{ \frac{(1-\theta)(1-2\theta-c_1(1-\theta))}{L}, \frac{1-c_1}{(L+2\kappa)(1+\kappa\alpha_{\max})^2} \right\}\right)^{-\gamma} - 1}, \frac{\zeta^2\epsilon_g^2}{2\mu\theta^2} \right\},$$

with the same $\gamma \in (0,1)$ *as used in Assumption* 4.2. *Then the expected number of iterations that Algorithm* 2.1 *takes until* $\phi(X_k) - \phi^\star \leq \epsilon$ *or* $\|\nabla\phi(X_k)\| \leq \frac{\zeta\epsilon_g}{\theta}$ *occurs is bounded as follows:*

$$\mathbb{E}[N_\epsilon] \leq \frac{2(1-\delta)}{(1-2\delta)^2 - \gamma}\left[2\log_{1/M}\left(\frac{\phi(x_0) - \phi^\star}{\epsilon}\right) + (1-\gamma)\log_\tau\left(\frac{\bar{\alpha}}{\alpha_0}\right)\right],$$

where $M = 1 - 2\mu c_1 \min\left\{ \frac{(1-\theta)(1-2\theta-c_1(1-\theta))}{L}, \frac{1-c_1}{(L+2\kappa)(1+\kappa\alpha_{\max})^2} \right\}$.

REMARK 4.31. *In the last two theorems the bound on* $\mathbb{E}[N_\epsilon]$ *depends on* ϵ *but not on* ϵ_g. *This bound should be understood as the bound on expected complexity to reach* ϵ*-accuracy in terms of the function value. If* $\|\nabla\phi(X_k)\| \leq \frac{\zeta\epsilon_g}{\theta}$ *occurs before the* ϵ*-accuracy in the function value is reached, the bound clearly still holds. The next theorem derives the bound on the complexity of reaching* ϵ*-accuracy in terms of* $\|\nabla\phi(X_k)\|$, *which applies to convex and nonconvex functions, and has no direct implications on accuracy in terms of the function value.*

THEOREM 4.32. *Let Assumptions* 1.1, 1.2, 1.3, 4.25, *and* 4.2 *hold. Moreover, let Assumption* 4.19 *hold, i.e.,*

$$\epsilon^2 > \max\left\{ \frac{2\epsilon_f}{\gamma c_1 \min\left\{ \frac{(1-\theta)(1-2\theta-c_1(1-\theta))}{L}, \frac{1-c_1}{(L+2\kappa)(1+\kappa\alpha_{\max})^2} \right\}}, \frac{\zeta^2\epsilon_g^2}{\theta^2} \right\},$$

with the same $\gamma \in (0,1)$ *as used in Assumption* 4.2. *Then the expected number of iterations that Algorithm* 2.1 *takes until* $\|\nabla\phi(X_k)\| \leq \epsilon$ *occurs is bounded as follows:*

$$\mathbb{E}[N_\epsilon] \leq \frac{2(1-\delta)}{(1-2\delta)^2 - \gamma}\left[\frac{M}{\epsilon^2} + (1-\gamma)\log_\tau\left(\frac{\bar{\alpha}}{\alpha_0}\right)\right],$$

where $M = \frac{\phi(x_0) - \hat{\phi}}{c_1 \min\left\{ \frac{(1-\theta)(1-2\theta-c_1(1-\theta))}{L}, \frac{1-c_1}{(L+2\kappa)(1+\kappa\alpha_{\max})^2} \right\}}$.

REMARK 4.33. *If* $\delta = \theta = \kappa = \epsilon_f = \epsilon_g = 0$, *our algorithm reduces to a deterministic line search algorithm with exact function evaluations and gradients. The dependence on the target accuracy* ϵ *is the same as that of a deterministic line search algorithm.*

REMARK 4.34. *Independent of the condition used on the gradient accuracy (condition* (4.1) *or* (4.2)*), the dependence on* ϵ *(the target accuracy) and* δ *(the probability of a true iteration) is the same. Moreover, in the setting where* $\theta = \kappa = \epsilon_g = 0$, *the results are identical. Finally, determining which condition is stronger is not trivial as it depends on the iteration-specific quantities* $\|\nabla\phi(x_k)\|$, $\|g_k\|$, *and* α_k.

5. Final remarks. We presented the analysis of a modified line search method that can be applied to functions with bounded noise, and where the gradient approximations g_k are possibly random, e.g., Gaussian smoothed gradients [16, 20] or sphere smoothed gradients [8, 9]. However, as a special case, we recover results for gradient approximations that are not random ($\delta = 0$), e.g., finite difference approximations [1, 10] or linear interpolation gradient approximations [7].

Furthermore, we discuss the effect of the parameter γ, which plays a crucial role in the analysis presented. This parameter depends on the error in the function evaluations, and effectively controls the size of the neighborhood of convergence, i.e., the lower bound on the ϵ. When there is zero error in the function evaluations, i.e., $\epsilon(x) = 0$ for all $x \in \mathbb{R}^n$, γ can be chosen arbitrarily close to zero, in which case we recover the exact convergence results from [6].

Finally, while our analysis assumes that the step size parameter is chosen using an adaptive line search procedure (Algorithm 2.2), and thus varies at every iteration, it also holds for a constant step size parameter choice. Namely, if $\alpha_0 \leq \bar{\alpha}$ and $\tau = 1$, then $\alpha_k \leq \bar{\alpha}$ for all k, and all *true* iterations are also *successful* iterations. Thus, as a special case of the analysis presented in section 4, we recover results for a fixed step size parameter procedure. We should note that the second term in the complexity bounds is zero in the case where $\tau = 1$ and $\alpha_0 = \bar{\alpha}$.

We establish a bound on the expected number of iterations N_ϵ that the algorithm takes until it reaches the desired near-optimal neighborhood. This is in contrast with the analyses of many other stochastic algorithms (such as stochastic gradient), where a bound is established on the expected "proximity" to the optimum (e.g., the expected smallest size of the gradient) achieved sometime during a given number of iterations. However, in all these cases there are no guarantees that the algorithm will remain in the near-optimal neighborhood, once it reaches it. To analyze the behavior of a stochastic algorithm, near optimality is a nontrivial task and requires considering the nature of the function in and near such a neighborhood. For example, for nonconvex functions, where the algorithm may converge to a near-saddle point, it will very likely leave the neighborhood and never return to it. On the other hand, if the objective function is strongly convex in the near-optimal neighborhood, then the algorithm is very likely to either stay in this neighborhood or keep returning to it frequently. Formally analyzing this behavior is the subject of a separate study.

REFERENCES

[1] A. S. Berahas, R. H. Byrd, and J. Nocedal, *Derivative-free optimization of noisy functions via quasi-Newton methods*, SIAM J. Optim., 29 (2019), pp. 965–993, https://doi.org/10.1137/18M1177718.

[2] A. S. Berahas, L. Cao, K. Choromanski, and K. Scheinberg, *A Theoretical and Empirical Comparison of Gradient Approximations in Derivative-Free Optimization*, preprint, https://arxiv.org/abs/1905.01332, 2019.

[3] J. Blanchet, C. Cartis, M. Menickelly, and K. Scheinberg, *Convergence rate analysis of a stochastic trust region method via supermartingale*, INFORMS J. Optim., 1 (2019), pp. 92–110.

[4] R. H. Byrd, G. M. Chin, J. Nocedal, and Y. Wu, *Sample size selection in optimization methods for machine learning*, Math. Program., 134 (2012), pp. 127–155.

[5] R. G. Carter, *On the global convergence of trust region algorithms using inexact gradient information*, SIAM J. Numer. Anal., 28 (1991), pp. 251–265, https://doi.org/10.1137/0728014.

[6] C. Cartis and K. Scheinberg, *Global convergence rate analysis of unconstrained optimization methods based on probabilistic models*, Math. Program., 169 (2018), pp. 337–375.

[7] A. R. Conn, K. Scheinberg, and L. N. Vicente, *Introduction to Derivative-Free Optimization*, MPS-SIAM Ser. Optim. 8, SIAM, Philadelphia, 2009, https://doi.org/10.1137/1.9780898718768.

[8] M. Fazel, R. Ge, S. M. Kakade, and M. Mesbahi, *Global convergence of policy gradient methods for the linear quadratic regulator*, in Proceedings of the 35th International Conference on Machine Learning, PMLR, 2018, pp. 1467–1476.
[9] A. D. Flaxman, A. T. Kalai, and H. B. McMahan, *Online convex optimization in the bandit setting: Gradient descent without a gradient*, in Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, Philadelphia, 2005, pp. 385–394.
[10] C. T. Kelley, *Implicit Filtering*, Software Environ. Tools 23, SIAM, Philadelphia, 2011, https://doi.org/10.1137/1.9781611971903.
[11] J. Larson, M. Menickelly, and S. M. Wild, *Derivative-free optimization methods*, Acta Numer., 28 (2019), pp. 287–404.
[12] A. Maggiar, A. Wächter, I. S. Dolinskaya, and J. Staum, *A derivative-free trust-region algorithm for the optimization of functions smoothed via Gaussian convolution using adaptive multiple importance sampling*, SIAM J. Optim., 28 (2018), pp. 1478–1507, https://doi.org/10.1137/15M1031679.
[13] J. J. Moré and S. M. Wild, *Benchmarking derivative-free optimization algorithms*, SIAM J. Optim., 20 (2009), pp. 172–191, https://doi.org/10.1137/080724083.
[14] J. J. Moré and S. M. Wild, *Estimating computational noise*, SIAM J. Sci. Comput., 33 (2011), pp. 1292–1314, https://doi.org/10.1137/100786125.
[15] Y. Nesterov, *Introductory Lectures on Convex Optimization*, Kluwer Academic, Boston, MA, 2004.
[16] Y. Nesterov and V. Spokoiny, *Random gradient-free minimization of convex functions*, Found. Comput. Math., 17 (2017), pp. 527–566.
[17] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., Springer Ser. Oper. Res., Springer, New York, 2006.
[18] C. Paquette and K. Scheinberg, *A stochastic line search method with expected complexity analysis*, SIAM J. Optim., 30 (2020), pp. 349–376, https://doi.org/10.1137/18M1216250.
[19] R. Pasupathy, P. Glynn, S. Ghosh, and F. S. Hashemi, *On sampling rates in simulation-based recursions*, SIAM J. Optim., 28 (2018), pp. 45–73, https://doi.org/10.1137/140951679.
[20] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, *Evolution Strategies as a Scalable Alternative to Reinforcement Learning*, preprint, https://arxiv.org/abs/1703.03864, 2017.