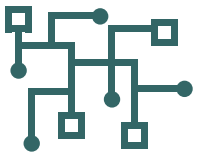

ISE

Industrial and Systems Engineering

A Gradient Sampling Algorithm for Noisy Nonsmooth Nonconvex Optimization

Albert S. Berahas

Department of Industrial and Operations Engineering, University of Michigan

Frank E. Curtis and Lara Zebiane

Department of Industrial and Systems Engineering, Lehigh University

COR@L Technical Report 26T-006-R1

LEHIGH UNIVERSITY

COR@L COMPUTATIONAL OPTIMIZATION RESEARCH AT LEHIGH

A Gradient Sampling Algorithm for Noisy Nonsmooth Nonconvex Optimization

Albert S. Berahas[*1], Frank E. Curtis[†2], and Lara Zebiane[‡2]

[1]Department of Industrial and Operations Engineering, University of Michigan
[2]Department of Industrial and Systems Engineering, Lehigh University

Original Publication: April 1, 2026
Last Revised: August 12, 2026

Abstract

An algorithm is proposed, analyzed, and tested for minimizing locally Lipschitz objective functions that may be nonconvex and/or nonsmooth. The algorithm, which is built upon the gradient-sampling methodology, is designed specifically for cases when objective and generalized gradient values might be subject to bounded, uncontrollable errors. A novel assumption pertaining to such errors in nonsmooth settings is introduced that decouples the noise that may be present due to typical errors in derivative estimates with noise that may be due to misidentification of segments of differentiability of the objective. Similarly to state-of-the-art guarantees for noisy smooth optimization, it is proved for the algorithm that, with probability one, either the sequence of objective values will decrease without bound or the algorithm will generate an iterate at which a measure of stationarity is below a threshold that depends proportionally on the error bounds for the objective function and generalized gradient values. The results of numerical experiments are presented, which show that the algorithm can indeed perform approximate optimization robustly despite errors in objective and generalized gradient values.

1 Introduction

Modern applications in a diverse set of areas, including machine learning, partial-differential-equation (PDE-)constrained optimization, and operations research, require solving continuous optimization problems where the objectives are merely presumed to be locally Lipschitz, meaning that they may be nonsmooth and/or nonconvex. In such settings, classical derivative-based algorithms such as gradient descent can fail since they rely on the objective being differentiable and the gradient function being continuous throughout the domain. General locally Lipschitz optimization, on the other hand, requires specialized algorithms. One of the most successful frameworks for solving such challenging problems is the gradient sampling (GS) methodology. Originally introduced and analyzed in foundational work by Burke, Lewis, and Overton [6], and later refined by Kiwiel [31], this methodology involves evaluating gradients at points that are sampled randomly in a neighborhood around each iterate to construct a search direction that reliably approximates a direction of descent. Subsequent research has significantly expanded this framework to include adaptive sampling strategies, quasi-Newton acceleration, and sequential quadratic programming methods for solving constrained problems; see, e.g., the articles [9, 10, 11, 12, 13, 15], as well as the survey paper [5].

[*]E-mail: albertberahas@gmail.com
[†]E-mail: frank.e.curtis@lehigh.edu
[‡]E-mail: lara.zebiane@lehigh.edu

Despite these efforts in algorithm design and analysis, all previously proposed GS methods share the assumption that exact objective function and generalized gradient values are available. However, for many modern settings, an algorithm only has access to inexact values. (Throughout the paper, we use *inexact* values, *noisy* values, values subject to *error*, and related such terms interchangeably when discussing situations in which exact function and/or derivative values are unavailable to an algorithm.) Such inexactness is often unavoidable, arising from numerical approximation, simulation-based modeling, or stochastic estimation procedures. In the smooth optimization regime, the challenges posed by such inexactness are well documented; researchers have developed various algorithms for solving both unconstrained and constrained problems with rigorous convergence and complexity guarantees that account for different types of noise models when the problem function(s) are smooth. In the context of a GS method for more general locally Lipschitz settings, however, noisy optimization has not been explored. (See §1.2 for more discussion of the related literature on different types of noise models and inexact generalized-gradient-based methods for solving nonsmooth optimization problems.) It is not surprising that, in these more general settings, errors are particularly disruptive as they corrupt the sampled objective gradient information, which leads to unreliability in the search direction computation and compromises the algorithm's convergence guarantees. This gap in the literature motivates the design of a GS framework that is designed specifically to handle errors in objective function and generalized gradient values while maintaining rigorous convergence guarantees that are consistent with the noiseless setting.

1.1 Contributions

The primary contributions of this work are the design and analysis of a GS-based algorithm for solving nonconvex and nonsmooth optimization problems that remains robust in the presence of errors in both the objective function and generalized gradient values, particularly when the errors in these quantities are only assumed to be bounded, and so may be biased and uncontrollable by the algorithm. Unlike previous GS methods that assume exact evaluations—and consistent with previously proposed gradient-based methods for noisy smooth optimization—our approach incorporates a threshold within the backtracking line search to prevent the failure of the search in the presence of noise. We provide a theoretical analysis proving that, with probability one, the algorithm either generates objective function values that decrease without bound or it generates an iterate satisfying a stationary condition whose accuracy is tied directly to the magnitudes of the evaluation errors. *Critical to our work is a novel assumption about the noise in the generalized gradient information that decouples the noise that may be due to shifts in the domain of the optimization variables with the noise that may be due to typical errors in the derivative evaluations; see Assumption 2.2 on page 6 and our subsequent discussions of it.* Overall, our proposed method extends classical gradient sampling theory by accounting for noise that is unavoidable in numerous settings of interest. To the best of our knowledge, the proposed method is the first gradient-sampling method that explicitly incorporates bounded evaluation errors in both objective and generalized gradient information while preserving convergence guarantees.

1.2 Literature Review

In this subsection, we recall some articles in the nonsmooth optimization literature that pertain to generalized-gradient-based methods for solving problems when objective function and/or generalized gradient values may be subject to inexactness/noise/errors. We close this subsection by providing some arguments about why it is particularly attractive to consider the extension of the gradient-sampling methodology to settings where values are subject to noise.

The literature on noisy optimization can be classified in a variety of ways. One major distinction is whether the noise in the function and/or derivative information is deterministic or stochastic. Roughly speaking, deterministic noise refers to situations in which two calls to an evaluation oracle with the same optimization-variable input always produce the same value, whereas stochastic noise refers to situations in which any call to an oracle—even with the same optimization-variable input—might produce a different value. The former type of noise may arise, for example, from computational noise [1, 38] generated through a numerical simulation, say for solving a discretized partial differential equation. On the other hand, the

latter type of noise may arise, e.g., in stochastic-gradient-type methods for solving machine learning and other data-driven problems [4].

There is a long history of work on stochastic/noisy subgradient-type methods for solving both convex and nonconvex unconstrained nonsmooth minimization problems; see, e.g., [2, 3, 18, 22, 33, 35, 36, 37, 41, 42, 44]. See also [17] for a more general setting of stochastic model-based optimization with complexity guarantees. The method in [19], which builds off of the work in [47], can be viewed as being related to a GS approach like the one that we propose, although it is quite distinct in spirit; the method is a normalized/randomized gradient method with complexity guarantees. Let us emphasize that, while some of these cited methods have theoretical guarantees for certain nonconvex problems, the guarantees require somewhat restrictive assumptions, such as weak convexity. By contrast, GS methods are applicable to a larger class of locally Lipschitz minimization problems.

More closely related to our general setting is the literature on proximal-bundle methods, many of which, like GS methods, have theoretical guarantees for locally Lipschitz minimization under generally loose assumptions. In terms of proximal-bundle methods that assume that only inexact information is available either in the objective function and/or generalized gradient values, we refer the reader to [20, 26, 27, 28, 30, 34, 43]. Among these, let us highlight [26], which proposes and analyzes a proximal-bundle method for solving nonconvex and nonsmooth minimization problems when errors in the objective and generalized-gradient values may be subject to nonvanishing noise, as we presume in our context.

We emphasize that there are benefits to extending the gradient sampling methodology to noisy settings. For example, GS methods are arguably simpler to implement than proximal-bundle methods, especially when it comes to solving nonconvex problems. Also, compared to such proximal-bundle methods, GS methods involve fewer parameters that require careful tuning in order to obtain good performance in practice. (For example, proximal-bundle methods for solving nonconvex problems require the use of techniques known as downshifting or tilting, which introduce parameters that need to be tuned carefully for good performance.) It is thus worthwhile to explore GS methods for noisy settings as approaches that are easier to implement and require less parameter tuning. We contend that these features can be seen for our proposed algorithm. In particular, despite having to deal with nonsmoothness, nonconvexity, and errors in objective and generalized-gradient values, our algorithm involves relatively few parameters, and even though our theoretical guarantees require that the parameters are chosen within certain bounds, our numerical experiments show that the behavior of our algorithm is reliable in practice, even when the parameter choices are not made precisely.

1.3 Notation

We use $\mathbb{N} := \{1, 2, \dots\}$ to denote the set of positive integers. We use $\mathbb{R}$ to denote the set of real numbers, $\mathbb{R}_{\geq r}$ (respectively, $\mathbb{R}_{>r}$) to denote the set of real numbers greater than or equal to (respectively, greater than) $r \in \mathbb{R}$, $\mathbb{R}^n$ to denote the set of real n-vectors, and $\mathbb{R}^{m\times n}$ to denote the set of real m-by-n matrices.

Given a nonempty set $\mathcal{X} \subseteq \mathbb{R}^n$, we denote the distance function from points in the domain $\mathbb{R}^n$ to the set $\mathcal{X}$ by $\mathrm{dist}_{\mathcal{X}} : \mathbb{R}^n \to \mathbb{R}$, which is defined by

$$\mathrm{dist}_{\mathcal{X}}(v) = \inf_{x\in\mathcal{X}} \|x - v\|_2 \quad \text{for all} \quad v \in \mathbb{R}^n.$$

If $\mathcal{X} \subseteq \mathbb{R}^n$ is nonempty, closed, and convex, then by the Projection Theorem [14, Theorem 6.5] there exists a unique projection of any $x \in \mathbb{R}^n$ onto $\mathcal{X}$, which we denote by $\mathrm{proj}_{\mathcal{X}}(x)$. Given $\epsilon \in \mathbb{R}_{\geq 0}$ and $x \in \mathbb{R}^n$, we denote the closed Euclidean (i.e., ℓ_2-norm) ball in $\mathbb{R}^n$ centered at x with radius ϵ as

$$\mathbb{B}^n_{\leq\epsilon}(x) := \{\overline{x} \in \mathbb{R}^n : \|\overline{x} - x\|_2 \leq \epsilon\}.$$

Given two nonempty sets $\mathcal{X} \subseteq \mathbb{R}^n$ and $\bar{\mathcal{X}} \subseteq \mathbb{R}^n$, we denote their Minkowski sum as

$$\mathcal{X} + \bar{\mathcal{X}} = \{x + \overline{x} \in \mathbb{R}^n : x \in \mathcal{X} \text{ and } \overline{x} \in \bar{\mathcal{X}}\}.$$

Given $\tilde{f} : \mathbb{R}^n \to \mathbb{R}$ and a scalar $r \in \mathbb{R}$, we denote the r-sublevel set of $\tilde{f}$ as

$$\mathcal{L}_r(\tilde{f}) := \{x \in \mathbb{R}^n : \tilde{f}(x) \leq r\}.$$

Suppose that a function $f : \mathbb{R}^n \to \mathbb{R}$ is locally Lipschitz in the sense that, at each point $x \in \mathbb{R}^n$, there exist $\epsilon_{f,x} \in \mathbb{R}_{>0}$ and $L_{f,x} \in \mathbb{R}_{>0}$ such that

$$|f(\bar{x}) - f(\hat{x})| \leq L_{f,x}\|\bar{x} - \hat{x}\|_2 \quad \text{for all} \quad (\bar{x}, \hat{x}) \in \mathbb{B}^n_{\leq \epsilon_{f,x}}(x) \times \mathbb{B}^n_{\leq \epsilon_{f,x}}(x). \tag{1}$$

By Rademacher's theorem [23, 39], this implies that f is continuous over $\mathbb{R}^n$ and differentiable almost everywhere in $\mathbb{R}^n$, i.e., the set in $\mathbb{R}^n$ over which it is not differentiable has a Lebesgue measure of zero [8, 14]. Following [8], we denote the Clarke generalized directional derivative function for f as $f^\circ : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ with

$$f^\circ(x, d) = \limsup_{\substack{\bar{x} \to x \\ \alpha \searrow 0}} \frac{f(\bar{x} + \alpha d) - f(\bar{x})}{\alpha} \quad \text{for all} \quad (x, d) \in \mathbb{R}^n \times \mathbb{R}^n.$$

Subsequently, we denote the Clarke generalized gradient mapping for f—a set-valued mapping—by $\eth f : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$, which is defined for all $x \in \mathbb{R}^n$ by

$$\begin{aligned} \eth f(x) &:= \{g \in \mathbb{R}^n : f^\circ(x, d) \geq g^T d \text{ for all } d \in \mathbb{R}^n\} \\ &= \text{conv}(\{g \in \mathbb{R}^n : \{\nabla f(x_k)\} \to g \text{ for some } \{x_k\} \to x \text{ with } \{x_k\} \subset \mathcal{D}_{\nabla f}\}). \end{aligned}$$

Here, $\mathcal{D}_{\nabla f} \subseteq \mathbb{R}^n$ is the full-measure set in $\mathbb{R}^n$ over which f is differentiable. For the equivalence indicated by the latter equation above, see, e.g., [8, 14, 40].

1.4 Organization

In §2, we state our optimization problem of interest, the assumptions that we make about the noise (i.e., errors) that may be present in the objective function and generalized gradient values, and our proposed algorithm. In §3, we present our theoretical convergence analysis of our proposed algorithm, which shows that, unless the sequence of objective values decreases without bound, the algorithm eventually generates an iterate at which a measure of stationarity with respect to the optimization problem is below a threshold that depends on the bounds on the errors in the objective function and generalized gradient values. We present the results of numerical experiments in §4 and concluding remarks in §5.

2 Problem and Algorithm Statements

Our proposed algorithm is designed to solve (approximately) the problem of minimizing an objective function $f : \mathbb{R}^n \to \mathbb{R}$, as in

$$\min_{x \in \mathbb{R}^n} \; f(x), \tag{2}$$

where f, along with a corresponding approximation function $\tilde{f}$ that is employed by the algorithm, at least satisfy the following assumption (recall (1)).

Assumption 2.1. *The objective $f : \mathbb{R}^n \to \mathbb{R}$ is locally Lipschitz. In addition, there exist $\epsilon_f \in \mathbb{R}_{\geq 0}$ and an approximation function $\tilde{f} : \mathbb{R}^n \to \mathbb{R}$ such that, at any $x \in \mathbb{R}^n$,*

$$|\tilde{f}(x) - f(x)| \leq \epsilon_f.$$

Moreover, given the starting point $x_1 \in \mathbb{R}^n$ employed by the algorithm, there exist $\bar{\epsilon} \in \mathbb{R}_{>0}$, $L_{f,\mathcal{X}} \in \mathbb{R}_{>0}$ (known by the algorithm), and an open set $\mathcal{X} \subseteq \mathbb{R}^n$ such that

$$\begin{aligned} &\mathcal{L}_{\tilde{f}(x_1)}(\tilde{f}) + \mathbb{B}^n_{\leq \bar{\epsilon}}(0) \subseteq \mathcal{X} \\ &\text{and} \quad |f(\bar{x}) - f(\hat{x})| \leq L_{f,\mathcal{X}}\|\bar{x} - \hat{x}\|_2 \quad \text{for all} \quad (\bar{x}, \hat{x}) \in \mathcal{X} \times \mathcal{X}. \end{aligned}$$

Under Assumption 2.1, the true objective function f may be nonconvex and/or nonsmooth over $\mathcal{X}$. The existence of ϵ_f and $\tilde{f}$ such that $|\tilde{f}(x) - f(x)| \leq \epsilon_f$ for all $x \in \mathbb{R}^n$ is a common assumption in the context of noisy smooth optimization; see, e.g., [1, 38]. Let us emphasize that, under Assumption 2.1, any two evaluations of the approximation function $\tilde{f}$ at the same point $x \in \mathbb{R}^n$ always yield the same value, namely, $\tilde{f}(x)$. The strongest part of Assumption 2.1 is the assumption that the algorithm has access to a Lipschitz constant for f over an open set $\mathcal{X}$ containing an enlargement of the sublevel set $\mathcal{L}_{\tilde{f}(x_1)}(\tilde{f})$. We emphasize that the Lipschitz constant required here is for the true objective f while the sublevel set over which it is defined is for the approximation function $\tilde{f}$. This is since the algorithm works with $\tilde{f}$ whereas our theoretical analysis requires knowledge of the Lipschitz constant for f. We do not assume that $\tilde{f}$ is Lipschitz continuous; indeed, under Assumption 2.1, it may even be discontinuous. Knowledge of such a Lipschitz constant $L_{f,\mathcal{X}}$ is not required, e.g., by the algorithms for noisy smooth optimization in [1, 38], and it is also not needed for GS methods when objective function and gradient values can be obtained without error. However, it is needed for the convergence guarantees for our proposed method, even though we argue in §4 that it can be ignored in any practical implementation. As it happens, in many situations, even with noisy function evaluations, it is reasonable to assume that one has access to such a Lipschitz constant (although not necessarily the smallest such Lipschitz constant). We further justify this part of Assumption 2.1 later in this section, and discuss along with our conclusions in §5 some approaches that one can, if desired, employ to compute/estimate such a Lipschitz constant for the objective f in practical applications of our proposed algorithm.

As is generally the case for nonsmooth optimization, we assume that at any point $x \in \mathbb{R}^n$ it would be intractable to compute the entire set of generalized gradients $\eth f(x)$, although it would be tractable to compute (or at least approximate) an element of this set. Along these lines, for our proposed algorithm, we assume that a generalized gradient approximation function is available to the algorithm that, for each x, returns a single vector as a generalized gradient approximation. To state our assumption about this function, we introduce two objects. First, let us introduce the ϵ-generalized gradient mapping $\eth_\epsilon f : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ as defined by Goldstein [24], which is defined for all $\epsilon \in \mathbb{R}_{\geq 0}$ and for any $x \in \mathbb{R}^n$ by

$$\eth_\epsilon f(x) := \mathrm{cl}\left(\mathrm{conv}\left(\mathcal{U}_{f,\epsilon}(x)\right)\right), \quad \text{where } \mathcal{U}_{f,\epsilon}(x) := \bigcup_{\bar{x} \in \mathbb{B}^n_{\leq \epsilon}(x)} \eth f(\bar{x}).$$

At the same time, given a function $\tilde{g} : \mathbb{R}^n \to \mathbb{R}^n$ (which will represent our generalized gradient approximation function), let us introduce for any $\epsilon \in \mathbb{R}_{>0}$ the mapping $\tilde{\mathcal{G}}_\epsilon(x) : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ as being defined for all $x \in \mathbb{R}^n$ by

$$\tilde{\mathcal{G}}_\epsilon(x) := \mathrm{cl}(\mathrm{conv}(\{\tilde{g}(\bar{x}) : \bar{x} \in \mathbb{B}^n_{\leq \epsilon}(x)\})).$$

We now make the following assumption about the generalized gradient approximation function that is available to the algorithm. (The factor of 1/3 in part (b) of the assumption could be generalized to any factor in (0,1). We merely use 1/3 as a natural choice when it comes to deriving a lower bound for the step size in the algorithm's line search; see upcoming Lemma 3.3.)

Assumption 2.2. *There exist $\epsilon_x \in \mathbb{R}_{\geq 0}$, $\epsilon_g \in \mathbb{R}_{\geq 0}$, and an approximation function $\tilde{g} : \mathbb{R}^n \to \mathbb{R}^n$ such that the following hold.*

(a) $\tilde{g}(x) \in \eth_{\epsilon_x} f(x) + \mathbb{B}^n_{\leq \epsilon_g}(0)$ for all $x \in \mathbb{R}^n$.

(b) $\eth_{\epsilon/3} f(x) \subseteq \tilde{\mathcal{G}}_\epsilon(x) + \mathbb{B}^n_{\leq \epsilon_g}(0)$ for all $x \in \mathbb{R}^n$ and $\epsilon \in \mathbb{R}_{>0}$ with $\epsilon \geq \epsilon_x$.

Let us emphasize that, under Assumption 2.2, any two evaluations of $\tilde{g}$ at the same point x always yield the same value $\tilde{g}(x)$. Otherwise, Assumption 2.2 requires some explanation as it is a unique contribution to the literature. The main idea of the assumption, which allows our algorithm and our analysis of it to reduce to the noiseless case when $(\epsilon_f, \epsilon_x, \epsilon_g) \to (0, 0, 0)$ is to decouple the noise in the generalized gradients that may be present due to certain shifts in the domain versus noise that may be present due to more typical derivative evaluation errors.

Assumption 2.2 is best understood through a few examples.

Example 2.1. *Consider the graphs pertaining to* $f : \mathbb{R} \to \mathbb{R}$ *with* $f(x) = |x|$ *on the upper-left of Figure 1 along with the corresponding noisy example on the upper-right of the figure. In this example, the noise in the objective values is consistent throughout the domain, as is commonly expected in the context of noisy optimization. As for the noise in the generalized gradient values, this particular example turns out to be relatively straightforward since the jump in the graph of* $\tilde{g}$ *occurs at* $x = 0$*, i.e., the unique point at which the true set of generalized gradients is not a singleton. Considering any* $\epsilon_x \in \mathbb{R}_{\geq 0}$*, one finds that* $\tilde{g}(0) \approx 1 \in [-1, 1] + \mathbb{B}^n_{\leq \epsilon_g}(0) = \eth_{\epsilon_x} f(x) + \mathbb{B}^n_{\leq \epsilon_g}(0)$ *for small* $\epsilon_g \in \mathbb{R}_{>0}$*, and for any* $x \neq 0$ *one finds that* $\tilde{g}(x) \in \{\nabla f(x)\} + \mathbb{B}^n_{\leq \epsilon_g}(0) \subseteq \eth_{\epsilon_x} f(x) + \mathbb{B}^n_{\leq \epsilon_g}(0)$ *for small* $\epsilon_g \in \mathbb{R}_{>0}$*. Thus, the inclusion in part (a) of Assumption 2.2 holds for any* $\epsilon_x \in \mathbb{R}_{\geq 0}$*, small* $\epsilon_g \in \mathbb{R}_{>0}$*, and all* $x \in \mathbb{R}$*. At the same time, now for any (arbitrarily small)* $\epsilon_x \in \mathbb{R}_{>0}$ *and small* $\epsilon_g \in \mathbb{R}_{>0}$*, one finds for any* $\epsilon \geq \epsilon_x$ *that* $\eth_{\epsilon/3} f(0) = [-1, 1] \subseteq \tilde{\mathcal{G}}_\epsilon(0) + \mathbb{B}^n_{\leq \epsilon_g}(0)$ *since* $\tilde{\mathcal{G}}_\epsilon(0)$ *captures values of* $\tilde{g}$ *both to the left and to the right of the origin. All that remains is to verify that there exist* $\epsilon_x \in \mathbb{R}_{\geq 0}$ *and* $\epsilon_g \in \mathbb{R}_{>0}$ *such that the inclusion in part (b) of the assumption holds for all* $x \neq 0$ *and* $\epsilon \geq \epsilon_x > 0$*. Consider any (arbitrarily small)* $\epsilon_x \in \mathbb{R}_{>0}$*. If such* (x, ϵ) *yields an interval* $[x - \epsilon/3, x + \epsilon/3]$ *that does not contain the origin, then* $\eth_{\epsilon/3} f(x) = \{\nabla f(x)\}$*, which is contained in* $\{\tilde{g}(x)\} + \mathbb{B}^n_{\leq \epsilon_g}(0)$ *for small* $\epsilon_g \in \mathbb{R}_{>0}$*, and so contained in* $\tilde{\mathcal{G}}_\epsilon(x) + \mathbb{B}^n_{\leq \epsilon_g}(0)$ *for small* $\epsilon_g \in \mathbb{R}_{>0}$*. On the other hand, if* $[x - \epsilon/3, x + \epsilon/3]$ *contains the origin, then the larger interval* $[x - \epsilon, x + \epsilon]$ *contains the origin in its interior, in which case one finds that* $\eth_{\epsilon/3} f(x) = [-1, 1]$ *is a subset of* $\tilde{\mathcal{G}}_\epsilon(x) + \mathbb{B}^n_{\leq \epsilon_g}(0)$ *for small* $\epsilon_g \in \mathbb{R}_{>0}$*. Putting everything together, one finds that Assumption 2.2 holds with any (arbitrarily small)* $\epsilon_x \in \mathbb{R}_{>0}$ *and small* $\epsilon_g \in \mathbb{R}_{>0}$*.*

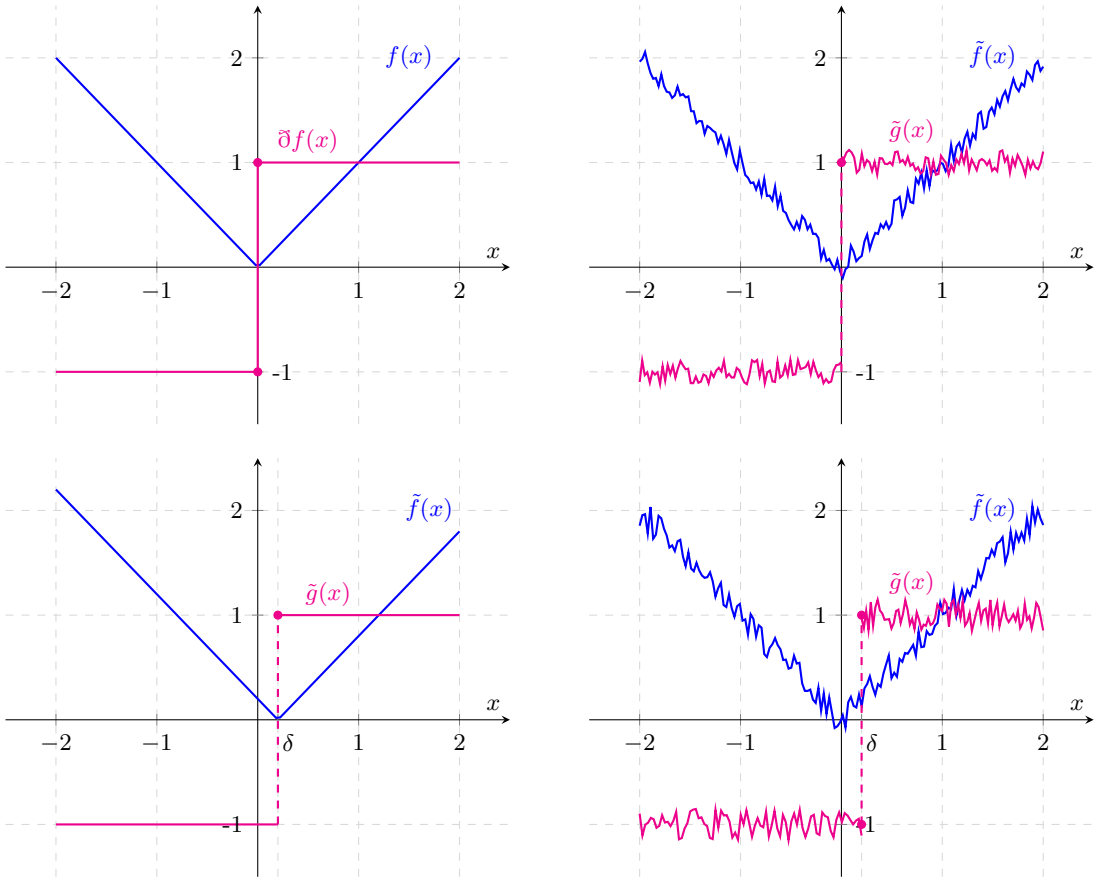

Figure 1: On the upper-left, the absolute value function and its corresponding generalized-gradient mapping. On the upper-right, the same mappings subject to bounded errors where the point of nondifferentiability of f is preserved exactly with respect to the jump in the graph of $\tilde{g}$. Here, and in the other examples, as is typical in noisy optimization, the mapping $\tilde{g}$ does *not* correspond to the derivative function of $\tilde{f}$; rather, the approximation function $\tilde{g}$ approximates elements of the generalized-gradient mapping directly. For this particular example on the upper-right, Assumption 2.2 holds with arbitrary $\epsilon_x > 0$ and positive $\epsilon_g \approx 0$. On the lower-left, a shifted absolute value function $\tilde{f}(x) = |x - \delta|$ for some $\delta \in \mathbb{R}_{>0}$ and its generalized-gradient mapping $\tilde{g}$. Assumption 2.2 holds for this example with $\epsilon_x > 3\delta/2$ and $\epsilon_g = 0$. On the lower-right, the same mappings subject to bounded errors, where errors in the generalized-gradient mapping are due to misidentification of the point of nondifferentiability as well as typical errors in derivative values. Assumption 2.2 holds with $\epsilon_x > 3\delta/2$ and $\epsilon_g \approx 0$.

The example illustrated in the upper-right of Figure 1 represents an idealized case in which the jump in the approximation function $\tilde{g}$ occurs exactly at the point in the domain at which the true generalized-gradient mapping does not yield a singleton. Consider next another idealized situation in which the noise in both the objective and generalized gradient values is due solely to a shift in the formulation of the approximate objective function, and beyond the consequences of this shift the generalized gradient values have no additional error.

Example 2.2. *Consider the example illustrated in the bottom-left of Figure 1. In this example, the noise in the objective is again consistent throughout the domain, as can be seen in the shifted graph. As for $\tilde{g}$, however, there is a large difference, e.g., between $\{\tilde{g}(x)\} = \{-1\}$ and $\eth f(x) = \{\nabla f(x)\} = \{1\}$ for all $x \in (0, \delta)$. That said, one can deduce that Assumption 2.2 indeed holds for appropriately chosen $\epsilon_x \in \mathbb{R}_{\geq 0}$ and $\epsilon_g = 0$. Let $\epsilon_g = 0$ be fixed for the remainder of the discussion of this example. With respect to part (a) of the assumption,*

one finds for any $\epsilon_x \geq \delta$ that $x < \delta$ gives $\tilde{g}(x) = -1 \in \eth f(\bar{x}) \subseteq \eth_{\epsilon_x} f(x)$ for some $\bar{x} \leq 0$ with $|x - \bar{x}| \leq \delta \leq \epsilon_x$, whereas $x \geq \delta$ gives $\tilde{g}(x) = 1 = \nabla f(x) \in \eth_{\epsilon_x} f(x)$. Hence, the inclusion in part (a) of the assumption holds for all $x \in \mathbb{R}$. Now to show that part (b) also holds, let us choose any $\epsilon_x > 3\delta/2$ (say, arbitrarily close to $3\delta/2$) and any $\epsilon \geq \epsilon_x$. If $x < -\epsilon/3$ or $x > \delta + \epsilon/3$, then it follows that $\eth_{\epsilon/3} f(x) = \{\nabla f(x)\} = \{\tilde{g}(x)\} \subseteq \tilde{\mathcal{G}}_\epsilon(x)$. On the other hand, for any $x \in [-\epsilon/3, \delta + \epsilon/3]$ one finds that $\delta < 2\epsilon_x/3 \leq 2\epsilon/3$, from which it follows that the distance from any such x to both the origin and the point δ is strictly less than ϵ. Consequently, one finds that

$$\tilde{\mathcal{G}}_\epsilon(x) = [-1, 1] \supseteq \eth_{\epsilon/3} f(x) \text{ for all } x \in [-\epsilon/3, \delta + \epsilon/3].$$

Overall, Assumption 2.2 holds with any $\epsilon_x > 3\delta/2$ and $\epsilon_g = 0$. This example demonstrates a critical feature of Assumption 2.2. That is, if errors in the generalized-gradient mapping occur essentially due to misidentification of the points of nondifferentiability, then such errors can be captured by $\epsilon_x > 0$ while the value for ϵ_g can remain small—to account only for errors in derivative values similarly as in the context of noisy smooth optimization, where a typical assumption is that the approximate gradient mapping $\tilde{g}$ yields $\|\tilde{g}(x) - \nabla f(x)\|_2 \leq \epsilon_g$ for all $x \in \mathbb{R}^n$ [1, 38].

To complete this discussion, let us provide one last example.

Example 2.3. *Consider the more realistic example illustrated on the bottom-right of Figure 1. Here, one can understand that noise in the generalized-gradient mapping is present due to both misidentification of the point of nondifferentiability and typical noise in the gradient estimates at points of differentiability. Combining the insights from the previous two examples, one finds that Assumption 2.2 holds with $\epsilon_x > 3\delta/2$ and small $\epsilon_g \in \mathbb{R}_{>0}$. It is worthwhile to emphasize that the assumption can also be seen to hold if the approximation function $\tilde{g}$ experienced multiple jumps (down to approximately -1 and up to approximately 1) in an interval containing the origin, as long as $\epsilon_x > 0$ is large enough with respect to the width of the interval containing the jumps. Overall, one finds that Assumption 2.2 represents a natural generalization of the noise assumptions typically employed in noisy smooth optimization [1, 38], where in the smooth setting one has $\epsilon_x = 0$.*

Let us now state our proposed algorithm, followed by some additional discussion to justify its design and our previously stated assumptions. Each iteration of our proposed algorithm operates in the following manner. Consider an arbitrary iteration index $k \in \mathbb{N}$, and let $x_k \in \mathbb{R}^n$ and $\epsilon_k \in \mathbb{R}_{>0}$ denote the kth iterate and sampling radius that have been generated by the algorithm, respectively. The iteration begins by the algorithm sampling a set of $m \geq n+1$ points $\{x_{k,1}, \ldots, x_{k,m}\}$ from a uniform distribution in a ball of radius ϵ_k that is centered at x_k. (This lower bound for m is consistent with GS methods for the noiseless setting; we discuss the role that it plays in our theoretical analysis in §3.) This set of points, along with the current iterate x_k, is used to construct the matrix

$$\tilde{G}_k := \begin{bmatrix} \tilde{g}(x_k) & \tilde{g}(x_{k,1}) & \cdots & \tilde{g}(x_{k,m}) \end{bmatrix}. \tag{3}$$

Then, as in a standard gradient-sampling method, a search direction is obtained by solving the primal-dual pair of quadratic optimization problems (QPs)

$$\left\{ \begin{array}{c} \min\limits_{(\tilde{z}, \tilde{d}) \in \mathbb{R} \times \mathbb{R}^n} \tilde{z} + \frac{1}{2}\|\tilde{d}\|_2^2 \\ \text{s.t. } \tilde{G}_k^T \tilde{d} \leq \tilde{z} \mathbb{1} \end{array} \right\} \quad \text{and} \quad \left\{ \begin{array}{c} \max\limits_{\tilde{y} \in \mathbb{R}^{m+1}} -\frac{1}{2}\|\tilde{G}_k \tilde{y}\|_2^2 \\ \text{s.t. } \mathbb{1}^T \tilde{y} = 1,\ \tilde{y} \geq 0 \end{array} \right\}. \tag{4}$$

The final main component of an iteration of our proposed algorithm is a backtracking line search that is conducted using the approximate function values that are available. Specifically, starting from a unit step size, the line search backtracks to attempt to obtain a step size $\alpha_k \in (0, 1]$ such that, for some user-defined parameter $\eta \in (0, \frac{1}{2})$ and threshold $\epsilon_{\text{ls}} \in \mathbb{R}_{>0}$, the following conditions are satisfied:

$$\tilde{f}(x_k + \alpha_k \tilde{d}_k) < \tilde{f}(x_k) - \eta \alpha_k \|\tilde{d}_k\|_2^2 + \epsilon_{\text{ls}} \quad \text{and} \quad \tilde{f}(x_k + \alpha_k \tilde{d}_k) \leq \tilde{f}(x_1). \tag{5}$$

(The latter condition merely ensures that the iterates remain within the sublevel set $\mathcal{L}_{\tilde{f}(x_1)}(\tilde{f})$, which partly justifies Assumption 2.1.) However, if the line search backtracks to a step size that is too small relative to

the known Lipschitz constant for f over $\mathcal{X}$ from Assumption 2.1, then it resets the step size to zero and proceeds to the next iteration, where a new sample set is generated. As shown in our theoretical analysis, this procedure ensures that, if x_k is far from stationarity in a certain sense, then any positive step size results in decrease in the approximation function $\tilde{f}$.

The complete algorithm is stated as Algorithm 1. As discussed further along with our numerical experiments in §4, despite the fact that the algorithm involves multiple input parameters, in practice the performance of the method is insensitive to changes in nearly all of them. The only one that has a significant effect on performance is the line-search parameter ϵ_{ls}, the behavior of which is the feature that we investigate primarily in our experiments. Besides the perturbation in the sufficient decrease condition in (5), there are two main differences between Algorithm 1 and a standard GS method for the noiseless setting. We comment on these two differences in the first two bullets below, then discuss well-posedness of the requirement for the initial sampling radius ϵ_1 in a third bullet.

Algorithm 1 Noise-Tolerant Gradient Sampling Algorithm

Require: $x_1 \in \mathbb{R}^n$ and $m \in \mathbb{N}$ with $m \geq n+1$

Require: $\gamma \in (0,1)$, $\eta \in (0, \frac{1}{2})$, $\theta \in (0,1)$, and $\phi \in \mathbb{R}_{>2}$

Require: $\nu \in \mathbb{R}_{>0}$, $\epsilon_f \in \mathbb{R}_{\geq 0}$, $\tilde{f} : \mathbb{R}^n \to \mathbb{R}$, $\bar{\epsilon} \in \mathbb{R}_{>0}$, $\epsilon_{\text{ls}} \in \mathbb{R}_{>2\epsilon_f}$, $\epsilon_x \in \mathbb{R}_{\geq 0}$, $\epsilon_g \in \mathbb{R}_{>0}$, $\tilde{g} : \mathbb{R}^n \to \mathbb{R}^n$, $L_{f,\mathcal{X}} \in \mathbb{R}_{>0}$, and $\epsilon_1 \in \mathbb{R}_{>0}$ such that Assumptions 2.1 and 2.2 hold,

$$\begin{aligned} &\epsilon_x < 3(L_{f,\mathcal{X}} + \epsilon_g), \\ &\epsilon_{\text{ls}} < \tfrac{9}{2}\eta\gamma\nu^2(L_{f,\mathcal{X}} + \epsilon_g)^2, \\ &\epsilon_1 \in (\max\{\zeta, \epsilon_x\}, 3(L_{f,\mathcal{X}} + \epsilon_g)) \\ \text{and } \ &\bar{\epsilon} \geq \epsilon_1 + \epsilon_x, \end{aligned} \tag{6}$$

where

$$\zeta := \left(\frac{6\epsilon_{\text{ls}}(L_{f,\mathcal{X}} + \epsilon_g)}{\eta\gamma\nu^2}\right)^{1/3}$$

for $k = 1, 2, 3, \ldots$ **do**
  sample $\{x_{k,1}, \ldots, x_{k,m}\}$ from a uniform distribution over $\mathbb{B}^n_{\leq \epsilon_k}(x_k)$
  set $\tilde{G}_k$ by (3)
  compute $(\tilde{z}_k, \tilde{d}_k, \tilde{y}_k)$ as the primal-dual solution of (4)
  set $\tilde{g}_k \leftarrow \tilde{G}_k \tilde{y}_k = -\tilde{d}_k$
  if $\|\tilde{g}_k\|_2 \leq \max\{\nu\epsilon_k, \phi\epsilon_g\}$ **then**
    if $\theta\epsilon_k < \max\{\zeta, \epsilon_x\}$ **then**
      return x_k
    end if
    set $\epsilon_{k+1} \leftarrow \theta\epsilon_k$
    set $\alpha_k \leftarrow 0$
  else
    set $\epsilon_{k+1} \leftarrow \epsilon_k$
    for $j = 0, 1, 2, \ldots$ **do**
      if $\gamma^j < \dfrac{\gamma\epsilon_k}{3(L_{f,\mathcal{X}} + \epsilon_g)}$ **then**
        set $\alpha_k \leftarrow 0$ and **break**
      else if (5) holds with $\alpha_k \equiv \gamma^j$ **then**
        set $\alpha_k \leftarrow \gamma^j$ and **break**
      end if
    end for
  end if
  set $x_{k+1} \leftarrow x_k + \alpha_k \tilde{d}_k$
end for

- A standard GS method that employs exact objective function and (generalized) gradient values involves an additional set of steps to ensure that $x_k \in \mathcal{D}_{\nabla f}$ for all $k \in \mathbb{N}$. We do not include such a procedure in Algorithm 1 since, in our setting, the algorithm cannot obtain accurate generalized gradient values, let alone determine whether a point is included in $\mathcal{D}_{\nabla f}$. (In fact, even in the noiseless setting, checking for inclusion in $\mathcal{D}_{\nabla f}$ may be impractical, which is why implementations of GS methods often skip these steps [6].) We are able to prove our theoretical guarantees without the algorithm having such a procedure since, due to noisy function and generalized gradient evaluations, our guarantees are naturally weaker than those that can be obtained with exact values.
- The algorithm's need for $L_{f,\mathcal{X}}$ as defined in Assumption 2.1 is due essentially to its stochastic nature. In the setting of noisy smooth optimization, it can be shown that if an iterate x_k is far from stationarity in the sense that $\|\nabla f(x_k)\|_2 \gg 0$, then, even with a perturbed sufficient decrease condition in the line

search (i.e., the addition of a constant such as ϵ_{ls} to the right-hand side), a step will yield decrease in the true objective function f. However, in the context of Algorithm 1, it is possible for the algorithm to yield insufficient decrease due to a bad sample set, i.e., not specifically due to the noise in the objective function and generalized gradient values. Algorithm 1 overcomes this issue through its knowledge of the Lipschitz constant $L_{f,\mathcal{X}}$, which in turn informs the algorithm of a threshold for the step size that would be obtained if the current iterate is sufficiently far from stationarity and the sample set is good in some sense. (All of these notions are characterized rigorously through our theoretical analysis in the next section.) Overall, while requiring knowledge of $L_{f,\mathcal{X}}$ is not ideal, we are not aware of any other GS approach that has been proposed for noisy nonconvex nonsmooth optimization with theoretical convergence guarantees, let alone one that does not require knowledge of such a Lipschitz constant. It should also be said that if in practice the algorithm only has access to a conservatively large Lipschitz constant, then the primary practical effect is that the threshold in Line 15 is smaller. In such a setting, Assumption 2.1 would hold, and the only bad effect is that the algorithm may perform additional approximate function evaluations during the line searches. It should also be said again that, in our numerical experiments, we do not make use of such a Lipschitz constant, yet for our experiments the algorithm experiences good performance nonetheless.

- The parameter requirements in (6) primarily perform the role of ensuring that the initial sampling radius ϵ_1 is set up appropriately for our theoretical guarantees. (This should not be surprising since, for example, it is natural to expect that the sampling radius should be initialized large enough with respect to the error bounds for the objective and generalized gradient values.) For a gradient-sampling method in the setting where function and generalized gradient values are computed without errors, there is no restriction on the initial sampling radius ϵ_1. In our setting, however, the interval for the initial sampling radius is needed for two purposes. The upper bound of the interval is needed to ensure that if the current iterate is far from stationarity and the sample set is good in a certain sense, then there is a useful lower bound for the step size that will be computed by the line search. Note that this upper bound for ϵ_1 appears in the line search procedure of the algorithm. As for the lower bound of the interval for ϵ_1, specifically that based on ζ (since $\epsilon_1 > \epsilon_x$ is quite natural), we include it to ensure that if the sampling radius is below a critical threshold, then it must have reduced the sampling radius at least once, which in turn means that the algorithm at least produced an iterate that can be recognized as approximately stationary. The details for this can be found in upcoming Theorem 3.1. It should be noted that the inequalities and inclusion in (6) are always satisfiable. First, the requirement that $\epsilon_x < 3(L_{f,\mathcal{X}} + \epsilon_g)$, which is one part of ensuring that the interval for ϵ_1 can be made nonempty, is a relatively loose requirement for practical purposes, especially in the typical context when $\epsilon_x \approx 0$, $\epsilon_g \approx 0$, and $L_{f,\mathcal{X}} \not\approx 0$; in any case, it can be ensured by selecting a larger Lipschitz constant, if necessary. Second, similar comments can be made about the requirement that $\epsilon_{\text{ls}} < \frac{9}{2}\eta\gamma\nu^2(L_{f,\mathcal{X}} + \epsilon_g)^2$, although in this case the condition can also be ensured by choosing larger ν. Finally, the lower bound for $\bar{\epsilon}$ is present to ensure that the set $\mathcal{X}$ in Assumption 2.1 is large enough to capture the region that appears in our theoretical results; note that it plays no direct role in the algorithm itself.

3 Analysis

For a gradient-sampling method in the noiseless setting, one can show that an algorithm similar to Algorithm 1 (with a few additional features) generates, with probability one, a sequence of iterates over which either the objective function values diverge to $-\infty$ or for which any limit point is stationary for f [5, 6, 14, 31]. Such a guarantee is not reasonable to expect in a noisy setting, since with noise in objective function and generalized-gradient evaluations, one should not expect that the algorithm would be able to generate iterates that converge all the way to stationarity for f. That being said, one can expect that, at any point that is far from stationarity, the algorithm will generate a search direction that leads to sufficient decrease in $\tilde{f}$, which in turn may correspond to sufficient decrease in f, at least when the norm of the step is large relative to the error bound ϵ_f. Consequently, our aim in this section is to show that, from any initial point, Algorithm 1 will at least generate a sequence of iterates such that, for some $k \in \mathbb{N}$, a measure of stationarity with respect

to x_k is small relative to the noise in the objective function and generalized-gradient values.

Our ultimate convergence guarantee is stated in Theorem 3.1 at the end of this section. Let us preview this guarantee before commencing our analysis. Informally, our convergence guarantee in Theorem 3.1 shows that, under reasonable assumptions and with probability one, Algorithm 1 generates $\{x_k\}$ over which either

(a) $\{f(x_k)\} \to -\infty$, or

(b) in some iteration $k \in \mathbb{N}$, one has that $\|g\|_2 = \mathcal{O}(\epsilon_g + \max\{\epsilon_{\text{ls}}^{1/3}, \epsilon_x\})$ for some $g \in \eth_{2\epsilon_k} f(x_k)$ with $\epsilon_k = \mathcal{O}(\max\{\epsilon_{\text{ls}}^{1/3}, \epsilon_x\})$; i.e., there exist positive real numbers C_ϵ and C_g such that, for some $k \in \mathbb{N}$, one finds $\epsilon_k \leq C_\epsilon \max\{\epsilon_{\text{ls}}^{1/3}, \epsilon_x\}$ and $\|g\|_2 \leq C_g(\epsilon_g + \max\{\epsilon_{\text{ls}}^{1/3}, \epsilon_x\})$ for some $g \in \eth_{2\epsilon_k} f(x_k)$.

Such a conclusion is all that one can expect in our noisy setting. It means that, with probability one, the algorithm will reduce the sampling radius to a level that is proportional to the noise bound ϵ_x for the generalized gradients and the perturbation factor in the line search condition (5), which in turn only needs to be proportional to the noise bound for the objective function values. Moreover, at this sampling radius, the algorithm will almost surely reach an iteration $k \in \mathbb{N}$ at which the minimum-norm element of $\eth_{2\epsilon_k} f(x_k)$ will have norm that is proportional to the noise bounds. We note upfront that the $\epsilon_{\text{ls}}^{1/3}$ term may be surprising, since from the setting of noisy smooth optimization one might expect this term to be $\epsilon_{\text{ls}}^{1/2}$. That is, assuming ϵ_{ls} is proportional to ϵ_f, one might expect the final generalized gradient error to be $\mathcal{O}(\epsilon_g + \max\{\epsilon_f^{1/2}, \epsilon_x\})$. We discuss the reason for this different exponent after the proof of our main theorem at the end of this section.

Let us now commence our analysis of Algorithm 1 to reach these conclusions. Our first lemma, a related version of which is a hallmark of GS methods (see [31, Lemma 3.1]), is adapted here for our analysis of the noisy setting.

Lemma 3.1. *Suppose that $\mathcal{G} \subset \mathbb{R}^n$ and $\tilde{\mathcal{G}} \subseteq \mathbb{R}^n$ are nonempty, convex, and compact sets such that the following conditions hold:*

(a) $\text{dist}_{\tilde{\mathcal{G}}}(0) > 2\epsilon_g$*;*

(b) $\text{dist}_{\tilde{\mathcal{G}}}(g) \leq \epsilon_g$ *for all* $g \in \mathcal{G}$*.*

Then, for any $\eta \in (0, \frac{1}{2})$, there exists $\beta \in \mathbb{R}_{>0}$ such that if $\tilde{u} \in \tilde{\mathcal{G}}$ satisfies $\|\tilde{u}\|_2 \leq \text{dist}_{\tilde{\mathcal{G}}}(0) + \beta$, then $\tilde{u}^T v > \eta \|\tilde{u}\|_2^2$ for all $v \in \mathcal{G}$.

Proof. Suppose that the implication is false. That is, suppose that for some $\eta \in (0, \frac{1}{2})$ and all $\beta \in \mathbb{R}_{>0}$, there exists $(\tilde{u}, v) \in \tilde{\mathcal{G}} \times \mathcal{G}$ such that $\|\tilde{u}\|_2 \leq \text{dist}_{\tilde{\mathcal{G}}}(0) + \beta$ and $\tilde{u}^T v \leq \eta \|\tilde{u}\|_2^2$. Consider such $\eta \in (0, \frac{1}{2})$. This implies that there exist infinite sequences $\{\tilde{u}_j\}$ and $\{v_j\}$ such that for all $j \in \mathbb{N}$ one has $\tilde{u}_j \in \tilde{\mathcal{G}}$, $v_j \in \mathcal{G}$, $\|\tilde{u}_j\|_2 \leq \text{dist}_{\tilde{\mathcal{G}}}(0) + 1/j$, and $\tilde{u}_j^T v_j \leq \eta \|\tilde{u}_j\|_2^2$. Since $\mathcal{G}$ and $\tilde{\mathcal{G}}$ are compact, the sequence $\{(\tilde{u}_j, v_j)\}$ has a convergent subsequence. Passing to a subsequence if necessary, one can conclude that there exists a pair of limit points $(\bar{u}, \bar{v}) \in \tilde{\mathcal{G}} \times \mathcal{G}$ such that

$$\bar{u}^T \bar{v} \leq \eta \|\bar{u}\|_2^2. \tag{7}$$

On the other hand, by the definition of the sequence $\{\tilde{u}_j\}$, it follows that $\bar{u} = \text{proj}_{\tilde{\mathcal{G}}}(0)$, which under condition (a) of the lemma has $\|\bar{u}\|_2 = \text{dist}_{\tilde{\mathcal{G}}}(0) > 2\epsilon_g$. Now let $\hat{v} = \text{proj}_{\tilde{\mathcal{G}}}(\bar{v})$. By the Projection Theorem [14, Theorem 6.5] and convexity of $\tilde{\mathcal{G}}$, one has $(0 - \bar{u})^T(\hat{v} - \bar{u}) \leq 0$, i.e., $\bar{u}^T \hat{v} \geq \|\bar{u}\|_2^2$. Since $\bar{v} \in \mathcal{G}$, condition (b) of the lemma gives $\|\bar{v} - \hat{v}\|_2 = \text{dist}_{\tilde{\mathcal{G}}}(\bar{v}) \leq \epsilon_g$. Thus, one finds that

$$\begin{aligned}\bar{u}^T \bar{v} &= \bar{u}^T(\hat{v} + \bar{v} - \hat{v}) \\ &\geq \bar{u}^T \hat{v} - \|\bar{u}\|_2 \|\bar{v} - \hat{v}\|_2 \geq \|\bar{u}\|_2^2 - \|\bar{u}\|_2 \epsilon_g = \|\bar{u}\|_2(\|\bar{u}\|_2 - \epsilon_g) \geq \tfrac{1}{2}\|\bar{u}\|_2^2,\end{aligned}$$

where the last inequality follows since $\|\bar{u}\| > 2\epsilon_g$, so $\frac{1}{2}\|\bar{u}\| > \epsilon_g$, which means that $\|\bar{u}\|_2 - \epsilon_g > \|\bar{u}\|_2 - \frac{1}{2}\|\bar{u}\|_2 = \frac{1}{2}\|\bar{u}\|_2$. However, this contradicts (7) since $\eta < \frac{1}{2}$. □ □

Let us now establish some useful properties that follow by construction of the algorithm. We emphasize that these properties hold regardless of the randomness induced by the sampling it performs when computing search directions.

Lemma 3.2. *Suppose that Assumptions 2.1 and 2.2 hold. Then, for all $k \in \mathbb{N}$:*

(a) $\epsilon_k \in [\max\{\zeta, \epsilon_x\}, \epsilon_1]$ *and* $\epsilon_{k+1} \leq \epsilon_k$*;*

(b) $\tilde{f}(x_k) \leq \tilde{f}(x_1)$ *and* $\mathbb{B}^n_{\leq \epsilon_x + \epsilon_k}(x_k) \subseteq \mathcal{X}$*;*

(c) $\tilde{g}_k \in \eth_{\epsilon_x+\epsilon_k} f(x_k) + \mathbb{B}^n_{\leq \epsilon_g}(0)$ *and* $\|\tilde{g}_k\|_2 \leq L_{f,\mathcal{X}} + \epsilon_g$*;*

(d) $\gamma^0 = 1 > \frac{\gamma\epsilon_k}{3(L_{f,\mathcal{X}}+\epsilon_g)}$*, so if the line search is reached it tests* (5) *at least once, then terminates with either* $\alpha_k = 0$ *or* $\alpha_k \geq \frac{\gamma\epsilon_k}{3(L_{f,\mathcal{X}}+\epsilon_g)}$*;*

(e) if $\alpha_k > 0$*, then* $\tilde{f}(x_{k+1}) < \tilde{f}(x_k) - \epsilon_{\text{ls}}$*, so overall* $\{\tilde{f}(x_k)\}$ *is monotonically nonincreasing and each iteration with* $\alpha_k > 0$ *decreases* $\tilde{f}$ *by more than* ϵ_{ls}*.*

Proof. Each part of the result can be proved as follows.

(a) The result follows by induction. The algorithm requires $\epsilon_1 > \max\{\zeta, \epsilon_x\}$. Then, $\epsilon_{k+1} = \epsilon_k$ or $\epsilon_{k+1} < \epsilon_k$, where the latter holds only if $\theta\epsilon_k \geq \max\{\zeta, \epsilon_x\}$ (otherwise, the algorithm terminated), in which case $\epsilon_{k+1} = \theta\epsilon_k \geq \max\{\zeta, \epsilon_x\}$.

(b) The inequality $\tilde{f}(x_k) \leq \tilde{f}(x_1)$ follows trivially by induction since $x_{k+1} \neq x_k$ only if (5) holds, which requires $\tilde{f}(x_{k+1}) \leq \tilde{f}(x_1)$ explicitly. Hence, for all $k \in \mathbb{N}$, one finds that $x_k \in \mathcal{L}_{\tilde{f}(x_1)}(\tilde{f})$, and along with Assumption 2.1 one finds that $\mathbb{B}^n_{\leq \epsilon_x+\epsilon_k}(x_k) \subseteq \mathcal{L}_{\tilde{f}(x_1)}(\tilde{f}) + \mathbb{B}^n_{\leq \bar{\epsilon}}(0) \subseteq \mathcal{X}$ since $\epsilon_x + \epsilon_k \leq \epsilon_x + \epsilon_1 \leq \bar{\epsilon}$.

(c) Each column of $\tilde{G}_k$ can be expressed as $\tilde{g}(x)$ for some $x \in \mathbb{B}^n_{\leq \epsilon_k}(x_k)$, and by Assumption 2.2(a) it follows that each such column satisfies $\tilde{g}(x) = g(x) + e(x)$ where $g(x) \in \eth_{\epsilon_x} f(x) \subseteq \eth_{\epsilon_x+\epsilon_k} f(x_k)$ and $\|e(x)\|_2 \leq \epsilon_g$. Moreover, since $\tilde{g}_k$ is a convex combination of the columns of $\tilde{G}_k$ and $\eth_{\epsilon_x+\epsilon_k} f(x_k)$ and $\mathbb{B}^n_{\leq \epsilon_g}(0)$ are convex, it follows that $\tilde{g}_k \in \eth_{\epsilon_x+\epsilon_k} f(x_k) + \mathbb{B}^n_{\leq \epsilon_g}(0)$, as claimed. Furthermore, by part (b) of this lemma and Assumption 2.1, one has that f is Lipschitz with constant $L_{f,\mathcal{X}}$ over the open set $\mathcal{X} \supseteq \mathbb{B}^n_{\leq \epsilon_x+\epsilon_k}(x_k)$, so every element of $\eth_{\epsilon_x+\epsilon_k} f(x_k)$ has norm at most $L_{f,\mathcal{X}}$, so $\|\tilde{g}_k\|_2 \leq L_{f,\mathcal{X}} + \epsilon_g$, as claimed.

(d) By part (a) and the requirement that $\epsilon_1 < 3(L_{f,\mathcal{X}} + \epsilon_g)$, one has $\frac{\gamma\epsilon_k}{3(L_{f,\mathcal{X}}+\epsilon_g)} \leq \frac{\gamma\epsilon_1}{3(L_{f,\mathcal{X}}+\epsilon_g)} < \gamma < 1 = \gamma^0$ for all $k \in \mathbb{N}$, so if the line search is reached, then it tests (5) at least once, as claimed. The remainder of the claim follows trivially by construction of the line search and the fact that $\gamma \in (0, 1)$.

(e) If $\alpha_k > 0$, then it must be true that the line search was reached, which by construction of the algorithm and part (d) means that $\|\tilde{d}\|_2 = \|\tilde{g}_k\|_2 > \max\{\nu\epsilon_k, \phi\epsilon_g\} \geq \nu\epsilon_k$ and $\alpha_k \geq \frac{\gamma\epsilon_k}{3(L_{f,\mathcal{X}}+\epsilon_g)}$, so by part (a) and the definition of ζ in the algorithm, it follows that

$$\eta\alpha_k\|\tilde{d}_k\|_2^2 \geq \frac{\eta\gamma\nu^2\epsilon_k^3}{3(L_{f,\mathcal{X}} + \epsilon_g)} \geq \frac{\eta\gamma\nu^2\zeta^3}{3(L_{f,\mathcal{X}} + \epsilon_g)} \geq 2\epsilon_{\text{ls}}. \tag{8}$$

Hence, with (5), one obtains $\tilde{f}(x_{k+1}) < \tilde{f}(x_k) - \eta\alpha_k\|\tilde{d}_k\|_2^2 + \epsilon_{\text{ls}} \leq \tilde{f}(x_k) - \epsilon_{\text{ls}}$.

Since each part has been proved individually, the proof is complete. □ □

Our next aim is to prove a lower bound for the step size obtained through the line search under certain critical conditions. Toward this end, let us now define for any $(x, \epsilon, \beta) \in \mathbb{R}^n \times \mathbb{R}_{>0} \times \mathbb{R}_{>0}$ the set

$$\mathcal{T}(x, \epsilon, \beta) = \Bigg\{ S \in \prod_{i\in[m]} \mathbb{B}^n_{\leq \epsilon}(x) : \\ \text{dist}_{\text{conv}(\{\tilde{g}(x)\}_{x\in S})}(0) \leq \text{dist}_{\tilde{\mathcal{G}}_\epsilon(x)}(0) + \beta \Bigg\}. \tag{9}$$

(Here, we overload notation and use $\{\tilde{g}(x)\}_{x\in S}$ to indicate the set of values of the approximation function $\tilde{g}$ evaluated at the elements in the tuple S.) Note that by its sampling procedure Algorithm 1 ensures for all $k \in \mathbb{N}$ that

$$S_k := (x_{k,1}, \ldots, x_{k,m}) \in \prod_{i\in[m]} \mathbb{B}^n_{\leq \epsilon_k}(x_k).$$

Therefore, if $S_k \in \mathcal{T}(x_k, \epsilon_k, \beta)$, then since $\tilde{g}_k$ is the minimum-norm element of the convex hull of the columns of $\tilde{G}_k$, it follows that

$$\|\tilde{g}_k\|_2 \leq \operatorname{dist}_{\tilde{\mathcal{G}}_{\epsilon_k}(x_k)}(0) + \beta. \tag{10}$$

Intuitively, the set in (9) identifies the sample sets that are good enough to guarantee that the minimum-norm element of $\tilde{\mathcal{G}}_\epsilon(x)$ is approximated well enough by the minimum-norm element of $\operatorname{conv}(\{\tilde{g}(x)\}_{x\in S})$ corresponding to the sample set S. This property is important for recognizing when a point x is approximately stationary. On the other hand, if x is not close to stationarity, then this property leads to a sufficiently large step size and, consequently, a sufficient decrease in the line search. This latter property is the subject of our next lemma.

Lemma 3.3. *Suppose that Assumptions 2.1 and 2.2 hold. Consider any $k \in \mathbb{N}$ such that*

$$\operatorname{dist}_{\tilde{\mathcal{G}}_{\epsilon_k}(x_k)}(0) > 2\epsilon_g. \tag{11}$$

Let $\eta \in (0, \frac{1}{2})$ be as in the algorithm and let $\beta_k \in \mathbb{R}_{>0}$ satisfy the conclusion of β in Lemma 3.1 with $\mathcal{G} = \eth_{\epsilon_k/3} f(x_k)$ and $\tilde{\mathcal{G}} = \tilde{\mathcal{G}}_{\epsilon_k}(x_k)$. Then, if line 14 is reached,

$$S_k \in \mathcal{T}(x_k, \epsilon_k, \beta_k) \implies \alpha_k \geq \frac{\gamma\epsilon_k}{3(L_{f,\mathcal{X}} + \epsilon_g)} \in (0, 1).$$

Proof. Under the stated conditions, suppose $S_k \in \mathcal{T}(x_k, \epsilon_k, \beta_k)$. By the definitions of $\tilde{\mathcal{G}}_{\epsilon_k}(x_k)$ and $\mathcal{T}(x_k, \epsilon_k, \beta_k)$ and (10), the algorithm computes $\tilde{g}_k$ with $\tilde{g}_k \in \tilde{\mathcal{G}}_{\epsilon_k}(x_k)$ and $\|\tilde{g}_k\|_2 \leq \operatorname{dist}_{\tilde{\mathcal{G}}_{\epsilon_k}(x_k)}(0) + \beta_k$. Hence, by Lemma 3.1, it follows that

$$\tilde{g}_k^T g > \eta \|\tilde{g}_k\|_2^2 \quad \text{for all} \quad g \in \eth_{\epsilon_k/3} f(x_k). \tag{12}$$

To derive a contradiction, suppose that $\alpha_k < \frac{\gamma\epsilon_k}{3(L_{f,\mathcal{X}}+\epsilon_g)}$, which by Lemma 3.2(d) means that $\alpha_k \leftarrow 0$. Let $\hat{\alpha}_k \in [\frac{\gamma\epsilon_k}{3(L_{f,\mathcal{X}}+\epsilon_g)}, \frac{\epsilon_k}{3(L_{f,\mathcal{X}}+\epsilon_g)}] \subset (0, 1)$ be the last positive step size considered by the line search before it set $\alpha_k \leftarrow 0$. Generally speaking, failure of the line search at $\hat{\alpha}_k$ means by (5) that either

$$\tilde{f}(x_k + \hat{\alpha}_k \tilde{d}_k) \geq \tilde{f}(x_k) - \eta\hat{\alpha}_k \|\tilde{d}_k\|_2^2 + \epsilon_{\text{ls}} \quad \text{or} \quad \tilde{f}(x_k + \hat{\alpha}_k \tilde{d}_k) > \tilde{f}(x_1). \tag{13}$$

Suppose that the latter of these two conditions in (13) holds, but the former does not. Then, from Lemma 3.2(b), one has that

$$\tilde{f}(x_k + \hat{\alpha}_k \tilde{d}_k) > \tilde{f}(x_1) \geq \tilde{f}(x_k).$$

Combining this with the fact that the former condition in (13) does not hold gives

$$\eta\hat{\alpha}_k \|\tilde{d}_k\|_2^2 < \epsilon_{\text{ls}}.$$

On the other hand, the fact that the line search was reached means $\|\tilde{d}_k\|_2 = \|\tilde{g}_k\|_2 > \nu\epsilon_k$, which as in (8) shows that $\eta\hat{\alpha}_k\|\tilde{d}_k\|_2^2 \geq 2\epsilon_{\text{ls}}$, a contradiction to the displayed inequality above. Thus, it is not possible for the latter condition in (13) to hold while the former condition in (13) does not, so we may proceed under the supposition that for $\hat{\alpha}_k$ the former condition in (13) holds, i.e.,

$$-\eta\hat{\alpha}_k \|\tilde{g}_k\|_2^2 + \epsilon_{\text{ls}} \leq \tilde{f}(x_k + \hat{\alpha}_k \tilde{d}_k) - \tilde{f}(x_k).$$

By the noise bound in Assumption 2.1, one consequently has that

$$-\eta\hat{\alpha}_k \|\tilde{g}_k\|_2^2 + \epsilon_{\text{ls}} \leq f(x_k + \hat{\alpha}_k \tilde{d}_k) - f(x_k) + 2\epsilon_f. \tag{14}$$

At the same time, Lebourg's Mean Value Theorem [14, Theorem 5.40] yields the existence of $\hat{x}_k$ on the interval $[x_k, x_k + \hat{\alpha}_k \tilde{d}_k]$ and $\hat{g}_k \in \eth f(\hat{x}_k)$ such that

$$f(x_k + \hat{\alpha}_k \tilde{d}_k) - f(x_k) = \hat{\alpha}_k \hat{g}_k^T \tilde{d}_k = -\hat{\alpha}_k \hat{g}_k^T \tilde{g}_k,$$

which with (14) gives

$$-\eta \hat{\alpha}_k \|\tilde{g}_k\|_2^2 + \epsilon_{\text{ls}} \leq -\hat{\alpha}_k \hat{g}_k^T \tilde{g}_k + 2\epsilon_f.$$

Rearranging the terms, and using the fact that $\epsilon_{\text{ls}} > 2\epsilon_f$, one obtains that

$$\hat{g}_k^T \tilde{g}_k \leq \eta \|\tilde{g}_k\|_2^2 + \frac{1}{\hat{\alpha}_k}(2\epsilon_f - \epsilon_{\text{ls}}) \leq \eta \|\tilde{g}_k\|_2^2,$$

which with (12) implies $\hat{g}_k \notin \eth_{\epsilon_k/3} f(x_k)$. On the other hand, Lemma 3.2(c) gives

$$\|\hat{x}_k - x_k\|_2 \leq \hat{\alpha}_k \|\tilde{g}_k\|_2 \leq \frac{\epsilon_k}{3(L_{f,\mathcal{X}} + \epsilon_g)}(L_{f,\mathcal{X}} + \epsilon_g) = \frac{\epsilon_k}{3},$$

which means that $\hat{g}_k \in \eth f(\hat{x}_k) \subseteq \eth_{\epsilon_k/3} f(x_k)$, a contradiction to the conclusion of the previous sentence. Since a contradiction has been reached, it must be true that $\alpha_k \geq \frac{\gamma \epsilon_k}{3(L_{f,\mathcal{X}} + \epsilon_g)}$, as claimed. Finally, $\frac{\gamma \epsilon_k}{3(L_{f,\mathcal{X}} + \epsilon_g)} \in (0, 1)$ from Lemma 3.2(d). □ □

Lemma 3.3 shows that, when the current iterate is not sufficiently close to stationarity, a good sample set yields a step size that is sufficiently large. In the context of a GS method that employs exact objective and generalized gradient values, it can be shown under loose assumptions that in the neighborhood of any point there exists a nonempty open set of such good sample sets, which in turn can be used to show that if the algorithm iterates converge to a point, it will at least generate an infinite subsequence of good sample sets. This conclusion relies primarily on the sample size being large enough (specifically $m \geq n + 1$) and an assumption that the objective function is continuously differentiable almost everywhere in $\mathbb{R}^n$; see, e.g., [14, Lemma 12.6]. Unfortunately, it is not possible to prove such a result in our setting since our noisy approximation function $\tilde{g}$ is not continuously differentiable almost everywhere in $\mathbb{R}^n$; indeed, it may even be discontinuous. Thus, to proceed in our analysis, we must impose an additional assumption about the approximation function $\tilde{g}$ that is employed by the algorithm. For the assumption that we employ to be reasonable in the noisy setting, i.e., for it to be consistent with the noiseless setting, we maintain in Algorithm 1 the requirement that $m \geq n + 1$, even though our assumption does not state this requirement explicitly.

We formalize our next assumption in the following manner. For each $k \in \mathbb{N}$, let $\mathcal{S}_k$ represent the random tuple of sample points generated in iteration k. This allows us to construct $(\Omega, \mathcal{F}, \mathbb{P})$ as a probability space that is generated by the sample tuples. Now letting $\{\mathcal{F}_k\}$ be the filtration defined by $\mathcal{F}_1$ (which is defined trivially with respect to the initial conditions of the algorithm) and otherwise $\mathcal{F}_k := \sigma(\mathcal{S}_1, \ldots, \mathcal{S}_{k-1})$ (i.e., the σ-algebra defined by all sample tuples prior to iteration $k \geq 1$), one has that x_k and ϵ_k are $\mathcal{F}_k$-measurable while $\mathcal{S}_k$ is $\mathcal{F}_{k+1}$-measurable. Specifically, conditioned on $\mathcal{F}_k$, the tuple $\mathcal{S}_k$ is uniformly distributed over $\prod_{i \in [m]} \mathbb{B}^n_{\leq \epsilon_k}(x_k)$. We now make the following reasonable assumption.

Assumption 3.1. *There exist measurable $\beta : \mathbb{R}^n \times \mathbb{R}_{>0} \to \mathbb{R}_{>0}$ and $p : \mathbb{R}^n \times \mathbb{R}_{>0} \to (0, 1]$ such that with $\tau_\phi := (\phi + 2)\epsilon_g/2$ and $\beta_\phi := (\phi - 2)\epsilon_g/2$ the following hold.*

- *(a) For any $(x, \epsilon) \in \mathbb{R}^n \times \mathbb{R}_{>0}$ with $\epsilon \geq \epsilon_x$ and $\text{dist}_{\tilde{\mathcal{G}}_\epsilon(x)}(0) > \tau_\phi$, the value $\beta(x, \epsilon) \in (0, \beta_\phi]$ satisfies the conclusion of Lemma 3.1 with $\mathcal{G} = \eth_{\epsilon/3} f(x)$, $\tilde{\mathcal{G}} = \tilde{\mathcal{G}}_\epsilon(x)$, and $\eta \in (0, \frac{1}{2})$ as in the algorithm.*
- *(b) For any $(x, \epsilon) \in \mathbb{R}^n \times \mathbb{R}_{>0}$ with $\epsilon \geq \epsilon_x$ and $\text{dist}_{\tilde{\mathcal{G}}_\epsilon(x)}(0) \leq \tau_\phi$, one has $\beta(x, \epsilon) = \beta_\phi$.*
- *(c) For all $k \in \mathbb{N}$, one has almost surely that*

$$\mathbb{P}(\mathcal{S}_k \in \mathcal{T}(x_k, \epsilon_k, \beta(x_k, \epsilon_k)) | \mathcal{F}_k) \geq p(x_k, \epsilon_k) > 0.$$

Under this assumption, we now present the following lemma, which essentially shows that if the sequence of noisy objective function values is bounded below, then, with probability one, the algorithm will terminate.

Lemma 3.4. *Suppose that Assumptions 2.1, 2.2, and 3.1 hold. For any $\hat{k} \in \mathbb{N}$, define*

$$\mathcal{E}_{\hat{k}} := \Big\{ \textit{Algorithm 1 does not terminate,} \\ \epsilon_k = \epsilon_{\hat{k}} \textit{ for all } k \geq \hat{k}, \textit{ and } \inf_{k \in \mathbb{N}} \tilde{f}(x_k) > -\infty \Big\}.$$

Then, $\mathbb{P}(\mathcal{E}_{\hat{k}}) = 0$. Thus, if $\inf_{k \in \mathbb{N}} \tilde{f}(x_k) > -\infty$, the algorithm terminates almost surely.

Proof. Consider arbitrary $\hat{k} \in \mathbb{N}$. To derive a contradiction, suppose that $\mathbb{P}(\mathcal{E}_{\hat{k}}) > 0$. On the event $\mathcal{E}_{\hat{k}}$, the sampling radius $\epsilon_{\hat{k}}$ is an $\mathcal{F}_{\hat{k}}$-measurable random variable satisfying $\epsilon_{\hat{k}} \geq \max\{\zeta, \epsilon_x\}$ by Lemma 3.2(a). Also, on the event $\mathcal{E}_{\hat{k}}$, for all $k \geq \hat{k}$ the sampling radius is not reduced and the algorithm does not terminate, from which it follows that the line search is reached with

$$\|\tilde{g}_k\|_2 > \max\{\nu \epsilon_{\hat{k}}, \phi \epsilon_g\} \quad \text{for all} \quad k \geq \hat{k}. \tag{15}$$

By Lemma 3.2(e), each iteration with $\alpha_k > 0$ decreases $\tilde{f}$ by more than ϵ_{ls}, whereas $\alpha_k = 0$ yields $\tilde{f}$ unchanged. Therefore, since $\tilde{f}$ is bounded below on the event $\mathcal{E}_{\hat{k}}$, it follows that there can only be a finite number of iterations with $k \geq \hat{k}$ and $\alpha_k > 0$. Hence, on the event $\mathcal{E}_{\hat{k}}$, there exists a random index $K \geq \hat{k}$ such that

$$\alpha_k = 0 \quad \text{and} \quad x_k = x_K \quad \text{for all} \quad k \geq K. \tag{16}$$

Now let us define, for each $k \in \mathbb{N}$, the event

$$\mathcal{A}_k := \{\mathcal{S}_k \in \mathcal{T}(x_k, \epsilon_k, \beta(x_k, \epsilon_k))\} \in \mathcal{F}_{k+1}.$$

By Assumption 3.1, one has almost surely that $\mathbb{P}(\mathcal{A}_k | \mathcal{F}_k) \geq p(x_k, \epsilon_k)$ with respect to the measurable function p. At the same time, on the event $\mathcal{E}_{\hat{k}}$, one has that $(x_k, \epsilon_k) = (x_K, \epsilon_{\hat{k}})$ for all $k \geq K$. Combined, this yields on the event $\mathcal{E}_{\hat{k}}$ that

$$\sum_{k \in \mathbb{N}} \mathbb{P}(\mathcal{A}_k | \mathcal{F}_k) \geq \sum_{k \geq K} p(x_K, \epsilon_{\hat{k}}) = \infty.$$

Thus, using the second Borel-Cantelli lemma [21, Theorem 4.3.4], it follows that almost surely on $\mathcal{E}_{\hat{k}}$ one has that $\mathcal{A}_k$ occurs for infinitely many $k \geq \hat{k}$. Consider now any realization on $\mathcal{E}_{\hat{k}}$ for which $\mathcal{A}_k$ occurs infinitely often and consider any $k \geq K$ with $\mathcal{S}_k \in \mathcal{T}(x_k, \epsilon_k, \beta(x_k, \epsilon_k))$. There are two cases to consider.

(a) If $\text{dist}_{\tilde{\mathcal{G}}_{\epsilon_{\hat{k}}}(x_K)}(0) \leq \tau_\phi$, then by (10) and Assumption 3.1(a)–(b) (since $\beta(x_K, \epsilon_{\hat{k}}) \leq \beta_\phi$ in either case) one has that $\|\tilde{g}_k\|_2 \leq \tau_\phi + \beta_\phi = \phi \epsilon_g$, contradicting (15).

(b) If $\text{dist}_{\tilde{\mathcal{G}}_{\epsilon_{\hat{k}}}(x_K)}(0) > \tau_\phi > 2\epsilon_g$, then by Assumption 3.1(a) the value $\beta(x_K, \epsilon_{\hat{k}})$ satisfies the conclusion of Lemma 3.1, and since the line search is reached one has from Lemma 3.3 that $\alpha_k \geq \frac{\gamma \epsilon_k}{3(L_{f,\mathcal{X}} + \epsilon_g)} > 0$, which contradicts (16).

Since a contradiction has been reached almost surely, in fact $\mathbb{P}(\mathcal{E}_{\hat{k}}) = 0$, as claimed.

With respect to the final claim of the lemma, suppose that the sequence $\{\tilde{f}(x_k)\}$ is bounded below and the algorithm does not terminate. Since $\epsilon_k \in [\max\{\zeta, \epsilon_x\}, \epsilon_1]$ for all $k \in \mathbb{N}$ by Lemma 3.2(a) and any reduction of the sampling radius causes it to be reduced by a fixed factor $\theta \in (0, 1)$, only a finite number of reductions can occur. Consequently, in any case, the sequence $\{\epsilon_k\}$ is eventually constant, which means that any realization of the algorithm lies in

$$\bigcup_{\hat{k} \in \mathbb{N}} \mathcal{E}_{\hat{k}}.$$

However, as proved already, this is a countable union of events each of which has probability zero of occurrence, meaning that the event that $\{\tilde{f}(x_k)\}$ is bounded below and the algorithm does not terminate has probability zero. □ □

Before presenting our main theorem, we prove the following lemma that provides conditions under which the distance from the origin to a certain Goldstein generalized gradient set is bounded by the norm of the search direction computed by the algorithm, up to the noise in the generalized gradient values.

Lemma 3.5. *Suppose that Assumptions 2.1 and 2.2 hold. Then, for any $k \in \mathbb{N}$,*

$$\operatorname{dist}_{\eth_{2\epsilon_k} f(x_k)}(0) \leq \|\tilde{g}_k\|_2 + \epsilon_g.$$

Proof. By Lemma 3.2(c), there exists $g \in \eth_{\epsilon_x+\epsilon_k} f(x_k)$ such that $\|g - \tilde{g}_k\|_2 \leq \epsilon_g$. On the other hand, by Lemma 3.2(a), one has $\epsilon_x \leq \max\{\zeta, \epsilon_x\} \leq \epsilon_k$, meaning that $\eth_{\epsilon_x+\epsilon_k} f(x_k) \subseteq \eth_{2\epsilon_k} f(x_k)$. Consequently, $\operatorname{dist}_{\eth_{2\epsilon_k} f(x_k)}(0) \leq \|g\|_2 \leq \|\tilde{g}_k\|_2 + \epsilon_g$. □ □

We now present our main theorem about the behavior of Algorithm 1. As one can expect from state-of-the-art guarantees for GS methods in the noiseless setting, as well as from state-of-the-art guarantees for the setting of noisy smooth optimization, the theorem shows that either the iterate sequence corresponds to true objective values that decrease without bound or the algorithm will terminate and return an iterate that is approximately stationary for f, where the termination tolerance is only limited by the objective and generalized gradient noise bounds.

Theorem 3.1. *Suppose that Assumptions 2.1, 2.2, and 3.1 hold. Then, Algorithm 1 generates sequences $\{x_k\}$ and $\{\epsilon_k\}$ such that, with probability one, either*

(a) the algorithm does not terminate and $\{f(x_k)\} \to -\infty$, or

(b) the algorithm terminates and returns a final iterate $x_k \in \mathbb{R}^n$ for some $k \in \mathbb{N}$ with

$$\epsilon_k \in [\bar{\zeta}, \theta^{-1}\bar{\zeta}) \quad \text{and} \quad \|\tilde{g}_k\|_2 \leq \theta^{-1} \max\{\nu\bar{\zeta}, \phi\epsilon_g\} \quad \text{with} \quad \bar{\zeta} := \max\{\zeta, \epsilon_x\},$$

from which it follows that, in the final iteration $k = k_*$, one has

$$\epsilon_{k_*} = \mathcal{O}(\max\{\epsilon_{\text{ls}}^{1/3}, \epsilon_x\}) \quad \text{and} \quad \operatorname{dist}_{\eth_{2\epsilon_{k_*}} f(x_{k_*})}(0) = \mathcal{O}(\epsilon_g + \max\{\epsilon_{\text{ls}}^{1/3}, \epsilon_x\}),$$

where these bounds tend to zero as $(\epsilon_f, \epsilon_{\text{ls}}, \epsilon_x, \epsilon_g) \to (0, 0, 0, 0)$.

Proof. By Lemma 3.2(e), the sequence $\{\tilde{f}(x_k)\}$ is monotonically nonincreasing. Thus, if the algorithm does not terminate, then either $\{\tilde{f}(x_k)\} \searrow -\infty$—meaning that (a) occurs under Assumption 2.1—or the sequence $\{\tilde{f}(x_k)\}$ is bounded below. However, using the same logic as in the proof of the final claim of Lemma 3.4, one can conclude that, with probability one, either the algorithm terminates or (a) occurs.

Now suppose that the algorithm terminates in iteration $k_* \in \mathbb{N}$. By construction, this means $\|\tilde{g}_{k_*}\|_2 \leq \max\{\nu\epsilon_{k_*}, \phi\epsilon_g\}$ and $\theta\epsilon_{k_*} < \bar{\zeta}$, whereas Lemma 3.2(a) shows $\epsilon_{k_*} \geq \bar{\zeta}$. Consequently, $\epsilon_{k_*} \in [\bar{\zeta}, \theta^{-1}\bar{\zeta})$, as claimed, and since $\theta \in (0, 1)$ one has

$$\|\tilde{g}_{k_*}\|_2 \leq \max\{\nu\theta^{-1}\bar{\zeta}, \phi\epsilon_g\} \leq \theta^{-1} \max\{\nu\bar{\zeta}, \phi\epsilon_g\}.$$

The final conclusion thus follows along with Lemma 3.5. □ □

As previously mentioned, one might be surprised by the $\epsilon_{\text{ls}}^{1/3}$ term in Theorem 3.1, since from the context of noisy smooth optimization one might expect this term to arise as $\epsilon_{\text{ls}}^{1/2}$. The reason for this different dependence on ϵ_{ls} comes from Lemma 3.4, the result of which depends on the lower bound for the step size proved in Lemma 3.3. Since the lower bound depends on ϵ_k, rather than on a constant term, one finds in (8) an upper bound for ϵ_{ls} that involves ϵ_k^3, rather than ϵ_k^2. Looking further, one finds that it is reasonable for the lower bound for the step size to depend on ϵ_k due to the proof technique employed for Lemma 3.3, which is a standard technique for proving this kind of result for a GS method.

We close this section by noting that, in practice, if the theoretical termination tolerance of the algorithm is not known, then as in the context of noisy smooth optimization, it is possible for the algorithm to produce an iterate satisfying the conditions of Theorem 3.1, then continue to produce iterates that do not satisfy such approximate stationarity conditions. However, in such cases, since the generalized gradient values have grown in norm again, the algorithm should be expected to produce directions that reduce the true objective again, meaning that the algorithm should be expected to return to approximate stationary points repeatedly.

4 Numerical Results

In this section, we demonstrate the performance of our Noise-Tolerant Gradient Sampling Algorithm (i.e., Algorithm 1) using a few distinct experimental setups. First, using a variant of the classical Rosenbrock function, we demonstrate with a challenging nonconvex and nonsmooth function that the behavior of the algorithm remains reliable across a spectrum of noise levels. Second, for a more realistic test case, we employ the algorithm to solve a PDE-constrained optimization problem where the noise arises due to approximate PDE and adjoint solves. Third, we employ the algorithm for a task to train a neural network for binary classification. With respect to these final experiments, we show that if the algorithm is employed in a manner such that our presumed noise bounds hold, then the algorithm performs reliably and demonstrates a clear trade-off between computational effort and classification accuracy. We also show, in a more realistic mini-batch setting, how the behavior of the algorithm depends primarily on the selection of the perturbation factor employed in the line search.

We emphasize at the outset of our experiments that, while Algorithm 1 involves multiple input parameters, the practical performance of the method is relatively insensitive to all but one of them. The performance is insensitive to γ and η; changes to these values only modestly affect the number of function evaluations required in the line searches. Similarly, the performance is insensitive to θ and ν; changes to these values only modestly affect the number of times that the sampling radius is reduced until the algorithm terminates. The performance is almost entirely insensitive to ϵ_x; theoretically, this value only affects the lower bound for the initial sampling radius (which in any case can decrease at a relatively fast rate) and termination test, but otherwise does not influence the iterate sequence. Similarly, the performance is relatively insensitive to ϵ_g; theoretically, this value only affects the condition for decreasing the sampling radius, but otherwise does not influence the iterate sequence. As previously discussed, the Lipschitz constant $L_{f,\mathcal{X}}$ also does not affect the performance greatly, and for our implementation (see §4.1) we do not attempt to estimate this value or employ it. Overall, the only parameter that significantly affects the performance of the algorithm is ϵ_{ls}, which theoretically depends on ϵ_f. Our experiments are primarily intended to explore the effect on performance of this parameter choice.

4.1 Implementation Details

All of our experiments were conducted using NonOpt through its Python interface.[1] NonOpt is an open-source C++ software package for solving nonsmooth and/or nonconvex minimization problems [16]. The software has implementations of multiple algorithms; for our experiments here, we employed its gradient-sampling framework through its `GradientCombination` choice for its `direction_computation` option. We also used the solver's `Backtracking` choice for its `line_search` option, rather than employ its default Weak Wolfe line search, to reflect the line search stated in Algorithm 1.

Otherwise, for consistency across all experiments, we primarily used the default parameter settings of NonOpt. (Any exceptions to these parameter choices are stated along with our discussion of each experiment in the following subsections.) This means that, rather than employ a straightforward GS-type approach as is stated in Algorithm 1, the implementation employs quasi-Newton Hessian approximations and adaptive sampling. We expect that our theoretical guarantees can be extended to variants of Algorithm 1 that use these strategies following [12, 13]. Using the default parameters in NonOpt, this means that for our experiments the parameters in Algorithm 1 were set as $m = 10$ (where adaptive sampling is invoked when $m < n + 1$), $\gamma = 0.5$, $\eta = 10^{-10}$, $\theta = 0.1$, and $\nu = 1$. Finally, rather than include the finite termination test in line 7 of Algorithm 1, we employed the default termination criteria in NonOpt, meaning that our implementation effectively chose $\epsilon_x = 0$ and ignored the finite termination test in line 7.

The values of ϵ_f, ϵ_g, and ϵ_{ls} that we employed varied across our experiments, as seen in the following subsections. As for $L_{f,\mathcal{X}}$, we did not attempt to estimate this constant in our experiments. This reflects practical scenarios, where one does not necessarily have access to such a Lipschitz constant. See §5 for further discussion. The fact that we did not attempt to estimate this constant had two consequences. First,

[1] https://github.com/frankecurtis/NonOpt

it means that our line search did not reset the step size to $\alpha_k = 0$ when the line search backtracked too much. Instead, as is default in NonOpt, the line search simply terminated whenever $\alpha_k < 10^{-20}$. Second, it means that our implementation did not attempt to choose ϵ_1 within the theoretical bound stated in Algorithm 1. Instead, we employed $\epsilon_1 = 10$ in all experiments.

4.2 Nonsmooth Rosenbrock Function

For our first experimental setup, we considered a nonsmooth variant of the classical Rosenbrock function as proposed by Nesterov; see [25]. Specifically, we considered the nonsmooth and nonconvex function

$$f(x, y) = (1 - x)^2 + 100|y - 2x^2 + 1|. \tag{17}$$

The global minimum of this function is $f(x_*, y_*) = 0$, attained at the unique minimizer $(1, 1)$. We introduced artificial noise such that the observed objective is $\tilde{f}(x, y) = f(x, y) + \xi_f$, where at each (x, y) the realization of ξ_f is drawn from a uniform distribution over the interval $[-\epsilon_f, \epsilon_f]$. Similarly, for the observed gradients we employed $\tilde{g}(x, y) = \nabla f(x, y) + \xi_g$, where at each (x, y) the realization of ξ_g is drawn from a uniform distribution over a Euclidean ball to ensure that $\|\xi_g\|_2 \leq \epsilon_g$. (We confirmed that the algorithm never requested a generalized gradient value at a point of nondifferentiability for f.) In our tests, we set $\epsilon_g = \sqrt{\epsilon_f}$ based on the common assumption that the gradient error is proportional to the square root of the error in the function values, as in finite-difference derivative approximations.

Figure 2 provides an illustration of the nonsmooth Rosenbrock function (17) around its minimizer. The left plot shows the true objective function f, whereas the right plot shows $\tilde{f}$ with $\epsilon_f = 1$. One sees that at this large noise level, the jagged surface possesses artificial stationary points. This visualization shows that an algorithm that presumes that exact function values are available can easily fail, say, if it gets trapped in a region with an artificial local minimizer. The perturbed line search in an algorithm such as Algorithm 1, on the other hand, can aid in escaping such regions, although one should still expect the behavior of the algorithm to be affected by noisy function and derivative values.

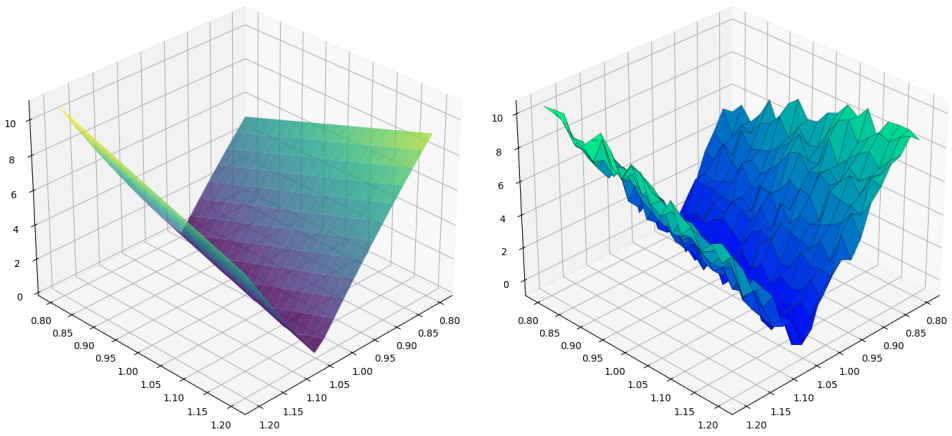

Figure 2: Surfaces of the nonsmooth Rosenbrock function (17). Left: the true surface near the global minimizer. Right: the surface corrupted by uniformly distributed noise with $\epsilon_f = 1$.

To evaluate the performance of our implemented algorithm, we conducted tests across four noise levels: $\epsilon_f \in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$. All runs of the algorithm started from the initial point $(-1.2, 1)$. For each noise level, we selected the line-search perturbation factor to have $\epsilon_{\text{ls}} > 2\epsilon_f$; specifically, we chose $\epsilon_{\text{ls}} = 2.1\epsilon_f$ in each case, ensuring that the condition in Assumption 2.1 was satisfied.

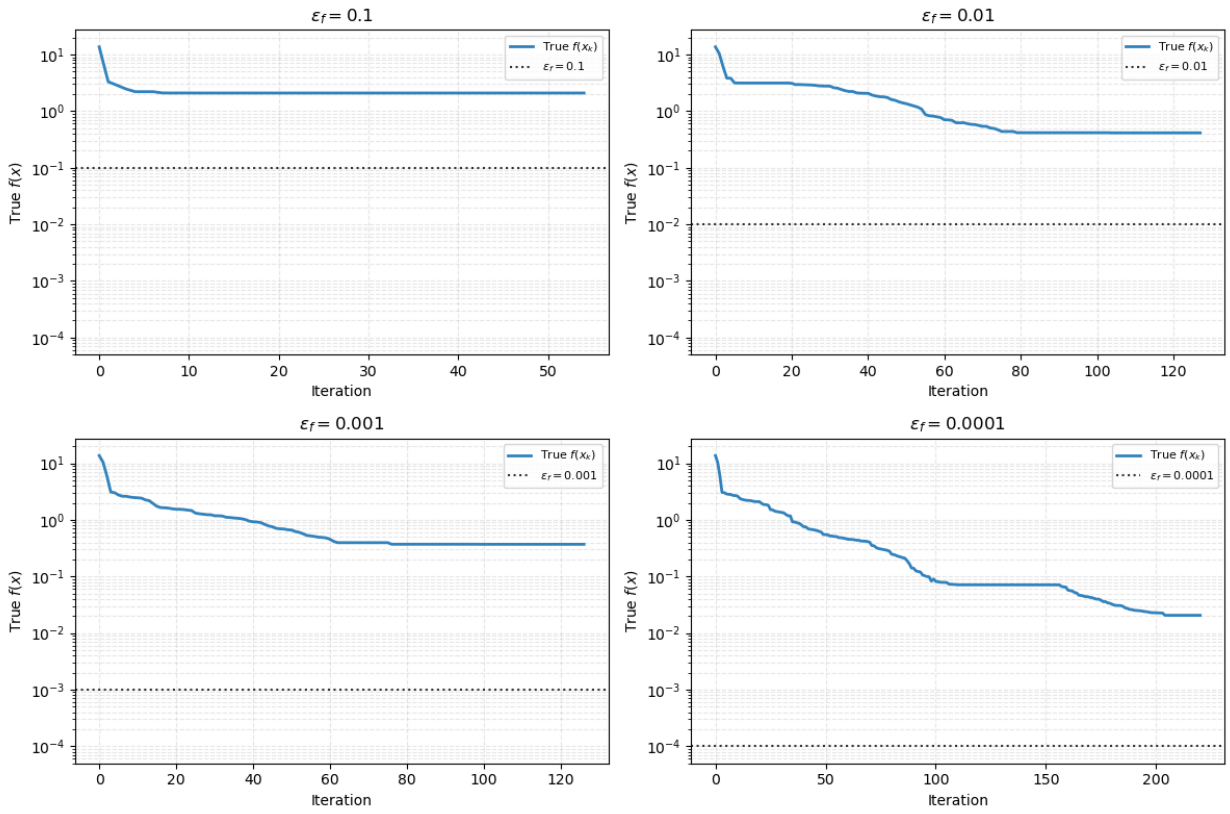

Figure 3: Nonsmooth Rosenbrock: True objective function values over k obtained by NonOpt across the four noise levels $\epsilon_f \in \{0.1, 0.01, 0.001, 0.0001\}$.

Figure 3 shows the behavior of $\{f(x_k)\}$ for each of the runs for the different noise levels. Despite the fact that the algorithm only had access to the noisy functions $(\tilde{f}, \tilde{g})$, the results show that the algorithm is able to make reliable progress in minimizing the true objective function across all of the noise levels. Indeed, even with a high noise level, the method maintains stable descent and significantly reduces the true objective value. As the noise level decreases, the convergence behavior becomes progressively smoother and increasingly resembles the behavior that one would expect in the noiseless setting, achieving lower final objective values. These results show that the algorithm is robust to noise across multiple scales, provided that the line-search tolerance is properly calibrated to the noise level.

To further illustrate the behavior of the algorithm for this challenging objective, we plot in Figure 4 the iterate trajectories for the same runs. For larger noise, the trajectory is relatively short as the algorithm stagnates due to the noise. It should be said, however, that the algorithm at least generates iterates that fall within the steep curved valley that contains the global minimizer, even though with high noise it is not able to navigate far around the valley toward the minimizer. As the noise decreases, however, the trajectories demonstrate better progress, approaching the behavior that one would expect in the noiseless setting.

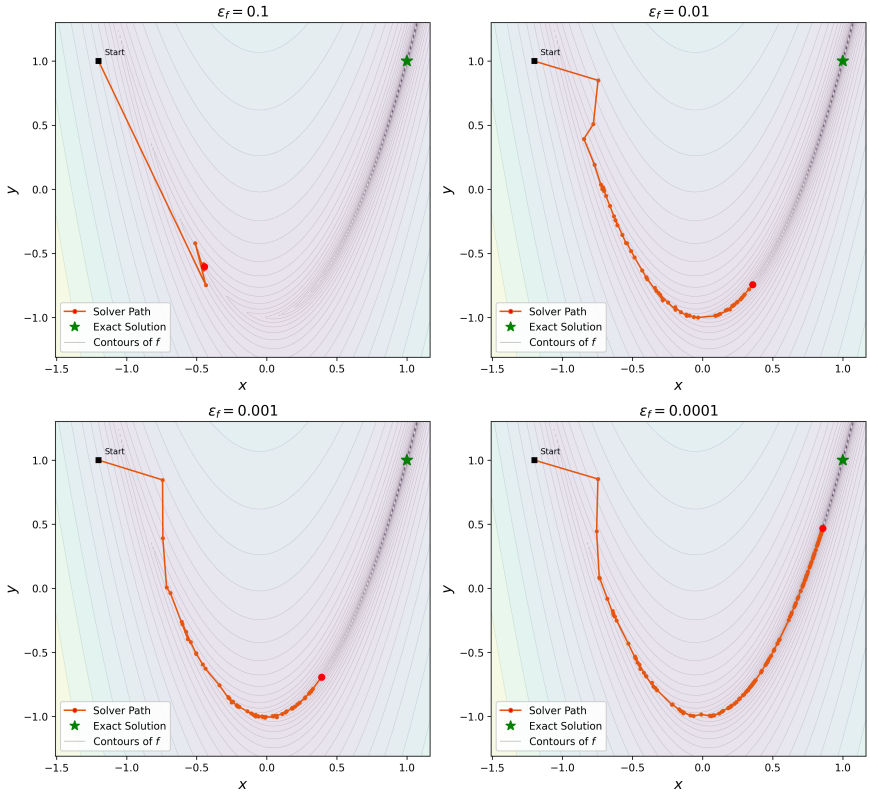

Figure 4: Nonsmooth Rosenbrock: Iterate trajectories obtained by NonOpt across the four noise levels $\epsilon_f \in \{0.1, 0.01, 0.001, 0.0001\}$.

Overall, this experimental setup demonstrates that the method offers stable behavior and is robust to noise, with improved performance as ϵ_f decreases.

4.3 PDE-Constrained Optimization with Computational Noise

To evaluate the performance of our proposed algorithm in a higher-dimensional setting, and one in which the noise arises naturally through computational error rather than artificially, we consider a setting of PDE-constrained optimization. For this example, Assumption 2.1 holds with an error bound ϵ_f that is known

a priori, and in a manner that is controllable by the user since it is determined by the tolerance imposed on a PDE solver, as explained below. (Recall that Algorithm 1 is applicable in settings in which the noise is uncontrollable. We consider here a setting of controllable noise since it allows for controlled experimentation.) We also demonstrate empirically that the errors in the generalized gradient values are of both of the types that are decoupled in Assumption 2.2, namely, typical derivative errors of size ϵ_g arising from an inexact adjoint solve, and distinct errors arising because an inexactly computed state can misidentify the correct segment in a maximum, i.e., the phenomenon that the value ϵ_x is designed to capture.

The following problem was constructed to have an objective function that is nonconvex and nonsmooth, and with controllable deterministic errors in the objective and generalized gradient evaluations. It uses ingredients that are common in the PDE-constrained optimization literature, namely, a semilinear elliptic state equation with a corresponding adjoint representation of derivatives (see, e.g., [46]), and a ℓ_1-norm penalty on the control as in the sparsity-promoting term in [45] that has also been explored for semilinear elliptic problems in [7]. It also views inexact simulation as a deterministic source of bounded error, as described in [32]. The remaining features of the problem—namely, the Chebyshev (L^∞) misfit, the minimum over two competing targets, and the concave peak-reward term—are introduced to obtain nonsmoothness and nonconvexity.

Consider the two-dimensional domain $\Omega \in [0,1]^2$. Given a control $x \in \mathbb{R}^n$, let the state of the system u be the solution of the semilinear elliptic PDE given by

$$-\Delta u + cu^3 = \sigma \sum_{j=1}^{n} x_j \chi_j \;\text{ in }\; \Omega \;\text{ while }\; u = 0 \;\text{ on }\; \partial\Omega, \tag{18}$$

where $(c, \sigma) \in \mathbb{R}_{>0} \times \mathbb{R}_{>0}$ are given parameters, χ_j is the indicator function of the jth cell of a uniform 8×8 partition of Ω, and x (the optimization variable) represents a piecewise-constant array of actuators. Discretizing (18) with a standard five-point Laplacian A on a 32×32 interior grid yields the discrete state equation

$$R(u, x) := Au + cu^3 - \sigma Bx = 0 \tag{19}$$

where $A \in \mathbb{R}^{1024\times 1024}$, $B \in \mathbb{R}^{1024\times n}$, and $u \in \mathbb{R}^{1024}$ with $n = 64$. Since A is positive definite and cu^3 is monotone in u, the mapping $R(\cdot, x)$ yields unique $u = u(x)$ for each x. Moreover, as a function of x, the state function u is smooth with Jacobian $J(u) = A + 3c\,\mathrm{diag}(u^2) \succ 0$. Given two prescribed target profiles $u_d^{(1)}$ and $u_d^{(2)}$ and an observation index set I, the objective function that we consider is

$$f(x) = \min_{\ell\in\{1,2\}} \max_{i\in I} \left| u_i(x) - u_{d,i}^{(\ell)} \right| - \mu \max_{i\in I} u_i(x) + \lambda \|x\|_1, \tag{20}$$

where $\mu \in \mathbb{R}_{>0}$ and $\lambda \in \mathbb{R}_{>0}$ are given parameters. That is, the objective seeks to drive the state to whichever of two admissible profiles is easier to match in a worst-case sense while rewarding a large peak state value and penalizing non-sparse actuation. The objective is nonsmooth due to the inner max, the outer min, and the ℓ_1 term, and it is nonconvex since a minimum of (non-constant) convex functions is nonconvex and since u is a nonlinear function of x. For our formulation, the two targets are placed in opposite corners of Ω so that the two branches of the minimum have well-separated basins. Considering (19) with respect to u shows that $\|u(x)\|_\infty = \mathcal{O}(\|x\|^{1/3})$, so f is coercive and bounded below for any $\lambda \in \mathbb{R}_{>0}$. For our experiments, we set the constants $c = 10$, $\sigma = 20$, $\lambda = 10^{-2}$, and $\mu = 1/4$ with the index set I taken to be all interior grid points in Ω.

It is worth emphasizing that this problem admits a computable Lipschitz constant for the objective in (20), as required by Assumption 2.1. Indeed, every element of $\eth f(x)$ has the form $\sigma B^T J(u)^{-1} q + \lambda \mathrm{sign}(x)$ with $\|q\|_2 \le \sqrt{1+\mu^2}$, and since $J(u) \succeq A \succ 0$ one has $\|J(u)^{-1}\|_2 \le \|A^{-1}\|_2$ for all u, so

$$\|g\|_2 \le \sigma \|B\|_2 \|A^{-1}\|_2 \sqrt{1+\mu^2} + \lambda\sqrt{n} \;\text{ for all }\; g \in \eth f(x) \;\text{ for all }\; x \in \mathbb{R}^n. \tag{21}$$

Hence, f is Lipschitz and (21) provides $L_{f,\mathcal{X}}$ with $\mathcal{X} = \mathbb{R}^n$; for the parameter values here it is ≈ 4.26. This illustrates the fact (discussed further in §5) that in some applications the strongest part of Assumption 2.1 is in fact verifiable.

For our experiments, the approximation functions $\tilde{f}$ and $\tilde{g}$ are obtained by solving (19) and the associated adjoint equation inexactly. Let us describe the computational procedures for computing these values, starting with $\tilde{f}$. The state equation is solved by a damped fixed-point (Picard) iteration, terminated for any given x as soon as $\tilde{u}$ is obtained satisfying $\|R(\tilde{u}, x)\|_\infty \le \tau$ for a prescribed residual tolerance τ. Subtracting the state equations at $\tilde{u}$ and $u = u(x)$ gives $R(\tilde{u}, x) = M(\tilde{u} - u)$ with $M = A + c\,\mathrm{diag}(\tilde{u}^2 + \tilde{u}u + u^2)$. Since $\tilde{u}^2 + \tilde{u}u + u^2 \ge 0$ pointwise, M is A plus a nonnegative diagonal. Both are M-matrices, so $0 \le M^{-1} \le A^{-1}$ entrywise, from which it follows that $\|\tilde{u} - u\|_\infty \le \|A^{-1}\|_\infty \|R(\tilde{u}, x)\|_\infty$. The u-dependent part of (20) is the minimum of maxima of 1-Lipschitz functions minus μ times a maximum, so it is $(1+\mu)$-Lipschitz with respect to $\|\cdot\|_\infty$; consequently,

$$|\tilde{f}(x) - f(x)| \le (1+\mu)\|A^{-1}\|_\infty \tau \quad \text{for all} \quad x \in \mathbb{R}^n, \tag{22}$$

where $\|A^{-1}\|_\infty = \max(A^{-1}\mathbb{1})$ is computed once. Thus, Assumption 2.1 holds where ϵ_f is given implicitly by the user through the choice of τ. Let us also emphasize that $\tilde{f}$ is implemented to be deterministic—in the sense that repeated evaluations at the same x return the same value—although it is discontinuous since the number of solver iterations may change as x varies. This is permitted by Assumption 2.1.

Given any x, a generalized gradient approximation is computed by the adjoint approach. Specifically, for any x, letting ℓ^* attain the minimum in (20), i^* be the corresponding maximum, s be the sign of the active residual, and j^* be the index attaining the peak, one can solve $J(\tilde{u})p = se_{i^*} - \mu e_{j^*}$ (approximately) to obtain $\tilde{g} = \sigma B^T p + \lambda \mathrm{sign}(x)$. For our experiments, this linear system is solved inexactly by the conjugate gradient method (CG), where the termination tolerance for CG yields a bound ϵ_g on the resulting $\tilde{g}$. Here we emphasize that the active indices (ℓ^*, i^*, j^*) are determined from the *inexact* state $\tilde{u}$, so near a point of nondifferentiability of f they may differ from those that the true state would select. When this occurs, $\tilde{g}$ essentially approximates a generalized gradient at a nearby point rather than at x itself, the discrepancy being on the order of ϵ_x no matter the magnitude of ϵ_g. Overall, the situation is related to that illustrated on the bottom-right of Figure 1 for which Assumption 2.2 applies.

As in §4.1, we used NonOpt's `GradientCombination` direction computation and `Backtracking` line search with $\epsilon_1 = 10$, and otherwise used the software's default parameters. All runs started from $x_1 = \mathbb{1}$ at which $f(x_1) = 1.2895$. As in §4.2, we considered the four noise levels $\epsilon_f \in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$ and in each case supplied the solver with $\epsilon_{\mathrm{ls}} = 2.1\epsilon_f$. For each ϵ_f, the residual tolerance τ was set from (22) so that Assumption 2.1 holds with that value of ϵ_f, and the CG termination tolerance for the adjoint solve was set so that the corresponding bound satisfied $\epsilon_g = \epsilon_f$. (We did not use $\epsilon_g = \sqrt{\epsilon_f}$ as in §4.2, which reflects finite-difference gradient estimation since here the generalized gradient is obtained from an adjoint solve whose accuracy is controlled directly and independently of the state solve.) Every run was given a budget of 500 iterations so that the noise levels can be compared at equal iteration counts. For the presentation of our results, we employed a reference objective value as an approximation of a true minimum. In particular, we ran the solver with $\epsilon_f = 10^{-6}$ for an iteration limit of 1500 and obtained the final objective value of $f_{\mathrm{ref}} = 0.07475$, which we use below to compare against the final objective values obtained using noisy objective and generalized gradient values.

Table 1 summarizes the runs and Figure 5 shows the true objective values along each run. The algorithm makes reliable progress at every noise level, reducing the true objective from 1.2895 by more than an order of magnitude in all four cases. The final optimality gap $f(x_{500}) - f_{\mathrm{ref}}$ was approximately $1.13\epsilon_f$, $0.93\epsilon_f$, $3.72\epsilon_f$, and $5.05\epsilon_f$ for the four noise levels, respectively, showing that the attainable accuracy is proportional to ϵ_f over four orders of magnitude. This demonstrates the behavior predicted by Theorem 3.1. The measured errors also confirm that the bound (22) is valid and not unreasonably loose, i.e., $\max_k |\tilde{f}(x_k) - f(x_k)|$ was between one and two orders of magnitude below ϵ_f in each run.

Figure 6 illustrates Assumption 2.2 and the unique manner in which it distinguishes errors between ϵ_x and ϵ_g. For all k, we plot $\|\tilde{g}(x_k) - g(x_k)\|_2$, distinguishing the iterates at which the inexact state selected the same active indices as the exact state (blue dots) from those at which it did not (red dots). The two situations yield completely different behavior. In the iterations with correctly identified active indices, the error respected merely the adjoint tolerance ϵ_g, with median values of 1.6×10^{-2}, 7.3×10^{-4}, 4.7×10^{-5}, and 5.4×10^{-6} for the four noise levels. In other words, the error values tracked ϵ_g as the PDE-solve tolerance

is tightened. At the remaining iterates, on the other hand, the median error was 1.38×10^{-1}, 1.45×10^{-1}, 1.47×10^{-1}, and 1.52×10^{-1} across the four noise levels, which means that they were essentially *independent* of $\epsilon_g = \epsilon_f$. The proportion of such iterates falls as the PDE-solve tolerance is tightened, from 29% at $\epsilon_f = 10^{-1}$ to 2% at $\epsilon_f = 10^{-4}$, but the corresponding errors do not decrease.

This observation is worth emphasizing, since it shows that the decoupling in Assumption 2.2 is not merely a technical convenience. Had we attempted to bound the total generalized gradient error by ϵ_g alone, we would have required $\epsilon_g \approx 0.15$ at every noise level—independently of how accurately the PDE is solved in each iteration—which would have rendered the conclusion of Theorem 3.1 vacuous, since the stationarity threshold depends additively on ϵ_g. Assumption 2.2 instead attributes these large discrepancies to ϵ_x, allowing ϵ_g to shrink with the solver tolerance, and the guarantee degrades only through the sampling radius. The present example thus provides a natural setting in which $\epsilon_x \in \mathbb{R}_{>0}$ is unavoidable and both roles of the error bounds in Assumption 2.2 are relevant.

Table 1: PDE-constrained optimization problem: Performance across four noise levels with $\epsilon_{\text{ls}} = 2.1\epsilon_f$. Here, τ is the PDE residual tolerance implied by (22), "solves" is the total number of sparse linear systems solved (approximately), $f(x_{500}) - f_{\text{ref}}$ is the true objective gap at the final iterate ($f(x_1) = 1.2895$ and $f_{\text{ref}} = 0.07475$), and "mis." is the percentage of iterates at which the inexact state selected different active indices than the exact state.

ϵ_f	τ	solves	$f(x_{500}) - f_{\text{ref}}$	$\max_k \lvert\tilde{f}(x_k) - f(x_k)\rvert$	mis.
10^{-1}	$1.1 \times 10^{+0}$	62,985	0.11295	1.1×10^{-2}	29%
10^{-2}	1.1×10^{-1}	122,973	0.00935	5.5×10^{-4}	23%
10^{-3}	1.1×10^{-2}	132,647	0.00375	5.1×10^{-5}	8%
10^{-4}	1.1×10^{-3}	141,086	0.00055	2.4×10^{-6}	2%

4.4 Neural Network Training for Binary Classification

Finally, merely for the sake of experimentation of our proposed algorithm in a setting that is typical instead of stochastic noise, we considered a problem to train a neural network for binary classification using a dataset pertaining to the diagnosis of heart disease [29]. The dataset contains 1,026 data points, each with 13 features, along with a binary label indicating the presence or absence of heart disease. All features are normalized to have zero mean and unit variance. The model that we employed was a feed-forward neural network with a single hidden layer of size 10 and ReLU activation. The network outputs a single logit per sample, which is converted to a label probability via a sigmoid function. We employed the binary cross-entropy loss function. Due to the use of ReLU activation in the hidden layer, the resulting objective function is nonsmooth, in addition to being nonconvex.

For this challenging problem, we considered two experimental setups. The first is an idealized setting in which values for the noise bounds (ϵ_f, ϵ_g) are known explicitly. The second represents a more realistic scenario when the objective and generalized-gradient values are computed through mini-batching with a fixed mini-batch size and the noise bounds are not presumed to be known. In this latter case, we demonstrate algorithm performance in terms of its dependence on ϵ_{ls}.

4.4.1 Adaptive Sampling

In this experiment, our main purpose is to demonstrate the trade-off between computational effort and final classification accuracy for different noise levels, assuming in each run that the noise levels are known. In order to run these experiments, in each iteration when the objective function or a generalized-gradient value is requested by the solver, the number of data points that are employed is increased adaptively until conditions on par with those in Assumptions 2.1 and 2.2 are found to hold. This represents an unrealistic scenario in which the true objective and gradient functions are evaluated explicitly. Thus, this experiment is for demonstration purposes only; a more realistic setup is considered in the following subsection.

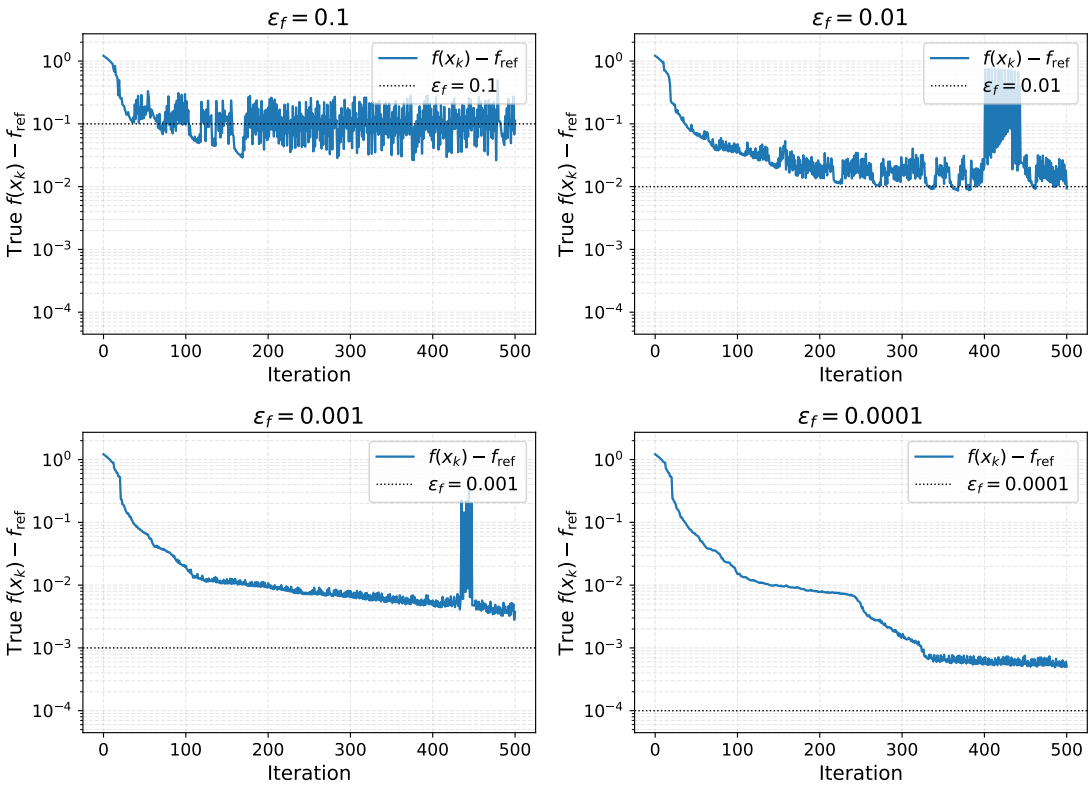

Figure 5: PDE-constrained optimization problem: true objective value, shown as the optimality gap $f(x_k) - f_{\text{ref}}$, over k obtained by NonOpt across the four noise levels $\epsilon_f \in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$ with $\epsilon_{\text{ls}} = 2.1\epsilon_f$. The dotted line in each panel marks ϵ_f. The true objective values were computed for plotting only and were never available to the algorithm.

More precisely, at each iteration, both the objective and gradient are estimated using a subset of data, starting from a small mini-batch. The mini-batch size is increased progressively until the desired accuracy conditions are satisfied, namely,

$$|\tilde{f}(x_k) - f(x_k)| \leq \epsilon_f \quad \text{and} \quad \|\tilde{g}(x_k) - g(x_k)\|_2 \leq \epsilon_g := \sqrt{\epsilon_f},$$

where $f(x_k)$ and $g(x_k)$ represent the objective function and gradient values that are computed using the entire set of data points and $\tilde{f}(x_k)$ and $\tilde{g}(x_k)$ are approximations obtained through mini-batch sampling.

To study the aforementioned trade-off between computational effort and classification accuracy, we run the algorithm for a range of ϵ_f values, recording both the total number of samples used throughout the optimization process and the final classification accuracy evaluated on the full dataset. For these experiments, we set `stationary_tolerance` of NonOpt to 10^{-8}. The results are reported in Figure 7. We observe a clear trade-off between computational effort and solution quality. Smaller values of ϵ_f lead to more accurate objective and gradient estimates, resulting a higher final accuracy, but at the expense of significantly increased computational cost in terms of the total number of samples employed. Conversely, larger values of ϵ_f reduce the number of samples required, but lead to poorer solver performance and lower classification accuracy. It should be said that there exists an intermediate level (e.g., $\epsilon_f = 0.05$), where the algorithm achieves high accuracy while using relatively fewer samples than the more exact setting. Overall, these results demonstrate that the algorithm offers a trade-off between accuracy and computational effort in modern settings such as neural network training.

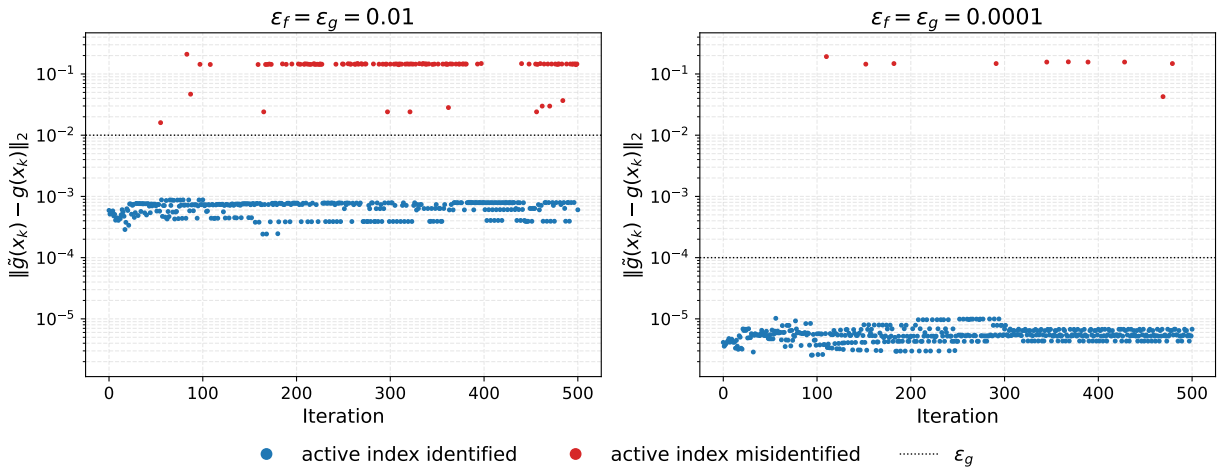

Figure 6: PDE-constrained optimization problem: generalized gradient error $\|\tilde{g}(x_k) - g(x_k)\|_2$ at each iterate for $\epsilon_f = 10^{-2}$ (left) and $\epsilon_f = 10^{-4}$ (right). Blue points are iterates at which the inexact state selected the same active indices as the exact state; red points are those at which it did not. The dotted line marks the adjoint-solve tolerance ϵ_g. The blue population tracks ϵ_g as the PDE solves are tightened, whereas the red population remains at $\mathcal{O}(10^{-1})$ regardless of ϵ_f, illustrating the distinction between the ϵ_g and ϵ_x contributions in Assumption 2.2.

4.4.2 Fixed Mini-Batch Sampling

For a more realistic setting in which our algorithm may be applied in the context of machine learning, we next investigate the behavior of the algorithm for neural network training in a fixed mini-batch-size setting. In our experiments here, the mini-batch size was fixed to 128 throughout the optimization process.

The same neural network architecture, dataset, and loss function as in the previous experiment were used. The only source of stochasticity arises from mini-batch sampling in each iteration. Here, we study the effect of ϵ_{ls}, which controls the sufficient decrease condition in the backtracking procedure. This parameter effectively determines how strictly the algorithm enforces descent in the presence of noisy function and gradient estimates. In each iteration, we recorded:

(i) the true objective value f evaluated at x_k,

(ii) the norm of the noisy/stochastic gradient evaluated at x_k,

(iii) the objective function error $\epsilon_{f,k} = |\tilde{f}(x_k) - f(x_k)|$, and

(iv) the gradient function error $\epsilon_{g,k} = |\tilde{g}(x_k) - g(x_k)|$.

We emphasize that f and g were only recorded in order to plot the results, and these quantities were not employed by the algorithm in any way. We ran the algorithm for $\epsilon_{\text{ls}} \in \{10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100\}$, using 10^{-4} for the `stationarity_tolerance` in NonOpt. The results are shown in Figure 8, where $f_{\text{true}} \equiv f$.

The results in Figure 8 illustrate the significant impact of the line-search parameter ϵ_{ls} on the behavior of the algorithm when fixed mini-batch sampling is employed. (The faint background traces represent the raw, noisy values of the objective and gradient metrics at each iteration. To clearly illustrate the underlying convergence trends despite the stochastic fluctuations, the dotted lines represent a moving average calculated with a window size of 8 iterations.) For small values of ϵ_{ls} (i.e., 10^{-3} and 10^{-2}), the line-search condition is overly strict, causing the algorithm to make little progress due to frequent step rejections in the presence of noise. This results in nearly flat objective trajectories and relatively poor final accuracies between 48% and

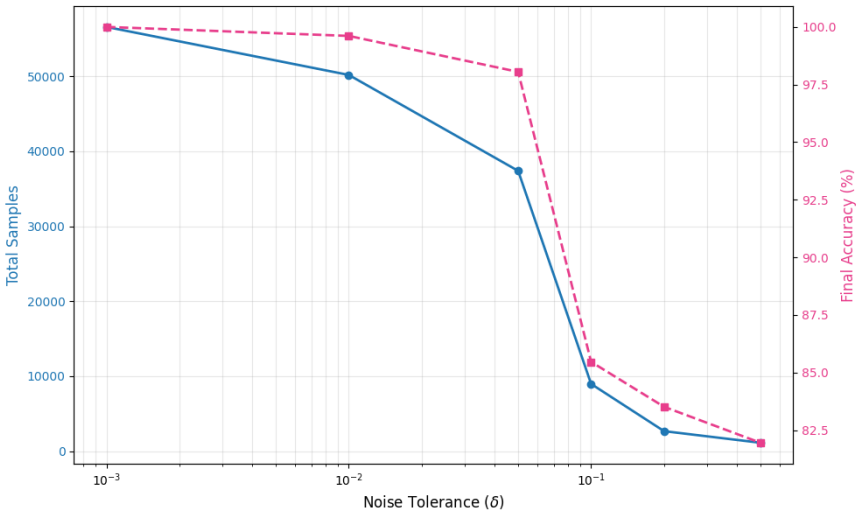

Figure 7: Trade-off between total samples used throughout the optimization process (i.e., computational cost) and final classification accuracy as a function of the noise level.

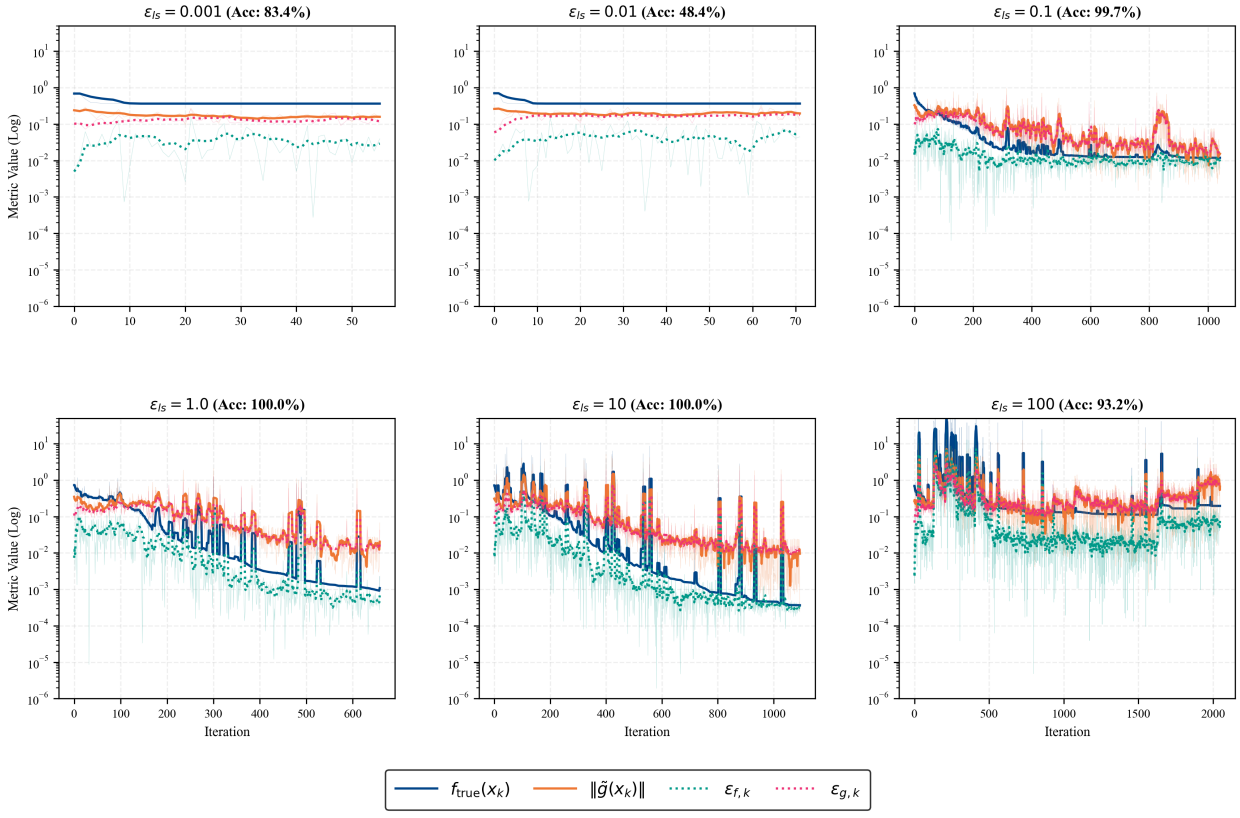

Figure 8: Various metrics over iterations for different ϵ_{ls} values. In the title of each plot, the final accuracy ("Acc.") over the training dataset is recorded.

83%. The discrepancy between these accuracies is due primarily to the randomness inherent in the algorithm terminating relatively early when minimizing the loss function, as is common in such settings.

In contrast, moderate values of ϵ_{ls} (i.e., 0.1, 1, and 10) yield stable behavior and better performance, achieving near-perfect to perfect classification accuracy from 99.7% to 100%. In these regimes, the condition $\epsilon_{\text{ls}} > 2\epsilon_f$ is often satisfied, as evidenced by the plotted values of $\epsilon_{f,k}$, which remain sufficiently small relative to ϵ_{ls}. However, when ϵ_{ls} becomes too large (i.e., 100), the method becomes unstable, producing erratic objective and gradient behavior, larger estimation errors, and a drop in accuracy to about 93%, indicating that the algorithm allows steps that do not make reliable improvement in the true objective function.

Overall, these results highlight a fundamental trade-off: too small ϵ_{ls} values lead to overly conservative updates and stagnation, while too large values cause instability, with intermediate values that satisfy the condition $\epsilon_{\text{ls}} > 2\epsilon_f$ offering the best compromise for robust and efficient optimization under stochastic noise.

5 Conclusion

We have proposed, analyzed, and tested an algorithm for solving nonconvex and nonsmooth optimization problems when both objective function and generalized-gradient evaluations may be corrupted by bounded, uncontrollable errors. To the best of our knowledge, ours is the first gradient-sampling-based method for solving problems in such a setting. We have proved reasonable convergence guarantees for our proposed algorithm, and demonstrated through multiple experiments that the algorithm is reliable and robust in the presence of noise, even when certain values required for the theoretical guarantees are not available in practice.

Our numerical experiments did not attempt to approximate a Lipschitz constant for the objective function despite the fact that a Lipschitz constant over a certain sublevel set is required for our theoretical guarantees. On one hand, setting a minimum step size to a very small value, such as 10^{-20} as is used in our experiments, would be consistent with our theoretical guarantees if the Lipschitz constant is not extremely large (i.e., if it is not large relative to 10^{20}). On the other hand, in practice, it is possible that one might obtain better performance if the Lipschitz constant were known, or at least estimated during the optimization process, and in turn employed by the algorithm. One way of obtaining a rough approximation of the Lipschitz constant is to maintain a maximum value of $|\tilde{f}(x_k) - \tilde{f}(x_{k+1})|/\|x_k - x_{k+1}\|_2$ over $k \in \mathbb{N}$. Such an approximation is reasonable as long as $\|x_k - x_{k+1}\|_2$ is large relative to ϵ_f. Thus, one might ignore such values when $\|x_k - x_{k+1}\|_2$ is smaller than a user-defined threshold. Overall, we do not expect that such an estimation procedure would lead to huge benefits for the algorithm's performance in practice. Rather, in our experience, the most important parameter is ϵ_{ls}, for which estimation techniques are well known.

References

[1] Albert S Berahas, Richard H Byrd, and Jorge Nocedal. Derivative-free optimization of noisy functions via quasi-newton methods. *SIAM Journal on Optimization*, 29(2):965–993, 2019.

[2] Pascal Bianchi, Walid Hachem, and Sholom Schechtman. Convergence of constant step stochastic gradient descent for non-smooth non-convex functions. *Set-Valued and Variational Analysis*, 30(3):1117–1147, 2022.

[3] Jérôme Bolte, Tam Le, and Edouard Pauwels. Subgradient sampling for nonsmooth nonconvex minimization. *SIAM Journal on Optimization*, 33(4):2542–2569, 2023.

[4] Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311, 2018.

[5] James V. Burke, Frank E. Curtis, Adrian S. Lewis, Michael L. Overton, and Lucas E. A. Simoes. Gradient sampling methods for nonsmooth optimization. In *Numerical Nonsmooth Optimization*, chapter 6, pages 201–225. Springer, 2020.

[6] James V. Burke, Adrian S. Lewis, and Michael L. Overton. A robust gradient sampling algorithm for nonsmooth, nonconvex optimization. *SIAM Journal on Optimization*, 15(3):751–779, 2005.

[7] Eduardo Casas, Roland Herzog, and Gerd Wachsmuth. Optimality conditions and error analysis of semilinear elliptic control problems with l^1 cost functional. *SIAM Journal on Optimization*, 22(3):795–820, 2012.

[8] F. H. Clarke. *Optimization and Nonsmooth Analysis*. Canadian Mathematical Society Series of Monographs and Advanced Texts. John Wiley & Sons, New York, NY, USA, 1983.

[9] Frank E. Curtis and Minhan Li. Gradient sampling methods with inexact subproblem solutions and gradient aggregation. *INFORMS Journal on Optimization*, 4(4):347–445, 2022.

[10] Frank E. Curtis, Tim Mitchell, and Michael L. Overton. A BFGS-SQP method for nonsmooth, nonconvex, constrained optimization and its evaluation using relative minimization profiles. *Optimization Methods and Software*, 32(1):148–181, 2017.

[11] Frank E. Curtis and Michael L. Overton. A sequential quadratic programming algorithm for nonconvex, nonsmooth constrained optimization. *SIAM Journal on Optimization*, 22(2):474–500, 2012.

[12] Frank E. Curtis and Xiaocun Que. An adaptive gradient sampling algorithm for nonsmooth optimization. *Optimization Methods and Software*, 28(6):1302–1324, 2013.

[13] Frank E. Curtis and Xiaocun Que. A quasi-Newton algorithm for nonconvex, nonsmooth optimization with global convergence guarantees. *Mathematical Programming Computation*, 7(4):399–428, 2015.

[14] Frank E. Curtis and Daniel P. Robinson. *Practical Nonconvex Nonsmooth Optimization*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2025.

[15] Frank E. Curtis, Daniel P. Robinson, and Baoyu Zhou. A self-correcting variable-metric algorithm framework for nonsmooth optimization. *IMA Journal of Numerical Analysis*, 40(2):1154–1187, 2020.

[16] Frank E. Curtis and Lara Zebiane. NonOpt: Nonconvex, nonsmooth optimizer. arXiv 2503.22826, 2025.

[17] Damek Davis and Dmitriy Drusvyatskiy. Stochastic model-based minimization of weakly convex functions. *SIAM Journal on Optimization*, 29(1):207–239, 2019.

[18] Damek Davis, Dmitriy Drusvyatskiy, Sham Kakade, and Jason D. Lee. Stochastic subgradient method converges on tame functions. *Foundations of Computational Mathematics*, 20(1):119–154, 2020.

[19] Damek Davis, Dmitriy Drusvyatskiy, Yin Tat Lee, Swati Padmanabhan, and Guanghao Ye. A gradient sampling method with complexity guarantees for lipschitz functions in high and low dimensions. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[20] Welington de Oliveira, Claudia Sagastizábal, and Claude Lemaréchal. Convex proximal bundle methods in depth: a unified analysis for inexact oracles. *Mathematical Programming*, 148(1):241–277, 2014.

[21] Richard Durrett. *Probability: Theory and Examples*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 5th edition, 2019.

[22] Yu. M. Ermolev and V. I. Norkin. Stochastic generalized gradient method for nonconvex nonsmooth stochastic optimization. *Cybernetics and Systems Analysis*, 34(2):196–215, 1998.

[23] Lawrence C. Evans and Ronald F. Gariepy. *Measure Theory and Fine Properties of Functions*. Mathematics & Statistics. Chapman and Hall/CRC, New York, NY, USA, Revised edition, 2015.

[24] A. A. Goldstein. Optimization of Lipschitz continuous functions. *Mathematical Programming*, 13(1):14–22, 1977.

[25] Mert Gurbuzbalaban and Michael L. Overton. On nesterov's nonsmooth chebyshev–rosenbrock functions. *Nonlinear Analysis: Theory, Methods and Applications*, 75(3):1282–1289, 2012. Variational Analysis and Its Applications.

[26] W. Hare, C. Sagastizábal, and M. Solodov. A proximal bundle method for nonsmooth nonconvex functions with inexact information. *Computational Optimization and Applications*, 63(1):1–28, 2016.

[27] Michael Hintermüller. A proximal bundle method based on approximate subgradients. *Computational Optimization and Applications*, 20(3):245–266, 2001.

[28] Ming Huang, Hui-Min Niu, Si-Da Lin, Zi-Ran Yin, and Jin-Long Yuan. A redistributed proximal bundle method for nonsmooth nonconvex functions with inexact information. *Journal of Industrial and Management Optimization*, 19(12):8691–8708, 2023.

[29] Andras Janosi, William Steinbrunn, Matthias Pfisterer, and Robert Detrano. Heart Disease. UCI Machine Learning Repository, 1989. DOI: https://doi.org/10.24432/C52P4X.

[30] Krzysztof C. Kiwiel. A proximal bundle method with approximate subgradient linearizations. *SIAM Journal on optimization*, 16(4):1007–1023, 2006.

[31] Krzysztof C. Kiwiel. Convergence of the gradient sampling algorithm for nonsmooth nonconvex optimization. *SIAM Journal on Optimization*, 18(2):379–388, 2007.

[32] Jorge J. Moré and Stefan M. Wild. Estimating computational noise. *SIAM Journal on Scientific Computing*, 33(3):1292–1314, 2011.

[33] Angelia Nedić and Dimitri P Bertsekas. The effect of deterministic noise in subgradient methods. *Mathematical programming*, 125(1):75–99, 2010.

[34] Dominikus Noll. Bundle method for non-convex minimization with inexact subgradients and function values. In *Computational and Analytical Mathematics: In Honor of Jonathan Borwein's 60th Birthday*, pages 555–592. 2013.

[35] V. I. Norkin. Stochastic generalized-differentiable functions in the problem of nonconvex nonsmooth stochastic optimization. *Cybernetics*, 22(6):804–809, 1986.

[36] V. I. Norkin. Stochastic generalized gradient methods for training nonconvex nonsmooth neural networks. *Cybernetics and Systems Analysis*, 57(5):714–729, 2021.

[37] E. A. Nurminskii. Minimization of nondifferentiable functions in the presence of noise. *Cybernetics*, 10(4):619–621, 1974.

[38] Figen Oztoprak, Richard Byrd, and Jorge Nocedal. Constrained optimization in the presence of noise. *SIAM Journal on Optimization*, 33(3):2118–2136, 2023.

[39] H. Rademacher. Über partielle und totale differenzierbarkeit von Funktionen mehrerer Variabeln und über die Transformation der Doppelintegrale. *Mathematische Annalen*, 79(4):340–359, 1919.

[40] R. T. Rockafellar and R. J. B. Wets. *Variational Analysis*, volume 317 of *Grundlehren Der Mathematischen Wissenschaften*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[41] Andrzej Ruszczyński. Convergence of a stochastic subgradient method with averaging for nonsmooth nonconvex constrained optimization. *Optimization Letters*, 14(7):1615–1625, 2020.

[42] Andrzej Ruszczyński. A stochastic subgradient method for nonsmooth nonconvex multilevel composition optimization. *SIAM Journal on Control and Optimization*, 59(3):2301–2320, 2021.

[43] Mikhail V. Solodov. On approximations with finite precision in bundle methods for nonsmooth optimization. *Journal of Optimization Theory and Applications*, 119(1):151–165, 2003.

[44] Mikhail V. Solodov and S. K. Zavriev. Error stability properties of generalized gradient-type algorithms. *Journal of Optimization Theory and Applications*, 98(3):663–680, 1998.

[45] Georg Stadler. Elliptic optimal control problems with l^1-control cost and applications for the placement of control devices. *Computational Optimization and Applications*, 44(2):159–181, 2009.

[46] Fredi Tröltzsch. *Optimal Control of Partial Differential Equations: Theory, Methods and Applications*, volume 112 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, USA, 2010. Translated from the 2005 German original by Jürgen Sprekels.

[47] Jingzhao Zhang, Hongzhou Lin, Stefanie Jegelka, Suvrit Sra, and Ali Jadbabaie. Complexity of finding stationary points of nonconvex nonsmooth functions. In *International Conference on Machine Learning*, pages 11173–11182. PMLR, 2020.