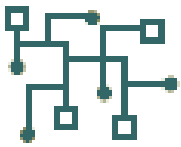


ISE

Industrial and
Systems Engineering

Stochastic Optimization Using a Trust-Region Method and Random Models

Ruobing Chen, Matt Menickelly , and Katya Scheinberg

Department of Industrial and Systems Engineering, Lehigh University, USA

ISE Technical Report 15T-002

LEHIGH
UNIVERSITY

Stochastic Optimization Using a Trust-Region Method and Random Models

Ruobing Chen*[1], Matt Menickelly †[1], and Katya Scheinberg ‡[1]

[1]Department of Industrial and Systems Engineering, Lehigh University, USA

April 15, 2015

Abstract

In this paper, we propose and analyze a trust-region model-based algorithm for solving unconstrained stochastic optimization problems. Our framework utilizes random models of an objective function $f(x)$, obtained from stochastic observations of the function or its gradient. Our method also utilizes estimates of function values to gauge progress that is being made. The convergence analysis relies on requirements that these models and these estimates are sufficiently accurate with sufficiently high, but fixed, probability. Beyond these conditions, no assumptions are made on how these models and estimates are generated. Under these general conditions we show an almost sure global convergence of the method to a first order stationary point. In the second part of the paper, we present examples of generating sufficiently accurate random models under biased or unbiased noise assumptions. Lastly, we present some computational results showing the benefits of the proposed method compared to existing approaches that are based on sample averaging or stochastic gradients.

1 Introduction

Derivative free optimization (DFO) [7] has recently grown as a field of nonlinear optimization which addressed optimization of black-box functions, that is functions whose value can be (approximately) computed by some numerical procedure or an experiment, while their closed-form expressions and/or derivatives are not available and cannot be approximated accurately or efficiently. Although the role of derivative-free optimization is particularly important when objective functions are noisy, traditional DFO methods have been developed primarily for deterministic functions. The fields of stochastic optimization [16, 22] and stochastic approximation [25] on the other hand focus on optimizing functions that are stochastic in nature. Much of the focus of these methods depend on the availability and use of stochastic derivatives, however, some work has addressed stochastic black box functions, typically by some sort of a finite differencing scheme [14].

*E-mail: Ruobing.Chen@us.bosch.com

†E-mail: mjm412@lehigh.edu

‡E-mail: katyas@lehigh.edu

In this paper, using methods developed for DFO, we aim to solve

$$\min_{x \in \mathbf{R}^n} \; f(x) \tag{1}$$

where $f(x)$ is a function which is assumed to be smooth and bounded from below, and whose value can only be computed with some noise. Let $\tilde{f}$ be the noisy computable version of f, which takes the form

$$\tilde{f}(x) = f(x, \varepsilon)$$

where the noise ε is a random variable.

In recent years, some DFO methods have been extended to and analyzed for stochastic functions [8]. Additionally, stochastic approximation methodologies started to incorporate techniques from the DFO literature [4]. The analysis in all that work assumes some particular structure of the noise, including the assumption that the noisy function values give an unbiased estimator of the true function value.

There are two main classes of methods in this setting of stochastic optimization: stochastic gradient (SG) methods (such as the well known Robbins-Monro method) and sample averaging (SA) methods. The former (SG) methods roughly work as follows: they obtain one realization of an unbiased estimator of the gradient at each iteration and take a step in the direction of the negative gradient. The step sizes progressively diminish and the iterates are averaged to form a sequence that converges to a solution. These methods typically have very inexpensive iterations, but exhibit slow convergence and are strongly dependent on the choice of algorithmic parameters, particularly the sequence of step sizes. Many variants exist that average the gradient information from past iterations and are able to accept sufficiently small, but nondecreasing step sizes [13, 1]. However, the convergence remains slow, and parameter tuning remains necessary in most cases. Moreover, the majority of these methods have been developed exclusively for convex functions and may not converge in non convex settings. An exception is the randomized stochastic (accelerated) gradient (RS(A)G) method presented in [11], [10]. This method employs random stopping criteria to provide theoretical first-order convergence even in nonconvex cases, but do not appear to be very practical.

The second class of methods, (SA), is based on sample averaging of the function and gradient estimators,which is applied to reduce the variance of the noise. These methods repeatedly sample the function value at a set of points in hopes to ensure sufficient accuracy of the function and gradient estimates. For a thorough introduction and references therein, see [18]. The optimization method and sampling process are usually tightly connected in these approaches, hence, again, algorithmic parameters need to be specially chosen and tuned. These methods tend to be more robust with respect to parameters and enjoy faster convergence at a cost of more expensive iterations. However, none of these methods are applicable in the case of biased noise and they suffer significantly in the presence of outliers.

The goal of this paper is to show that a *standard efficient unconstrained optimization method*, such as a trust region method, can be applied, with very small modifications, to stochastic nonlinear (not necessarily convex) functions and can be guaranteed to converge to first order stationary points as long as certain conditions are satisfied. We present a general framework, where we do not specify any particular sampling technique. The framework is based on the trust region DFO framework [7], and its extension to probabilistic models [2]. In terms of this framework and the certain conditions that must be satisfied, we essentially assume that

- the local models of the objective function constructed on each iteration satisfy some first order accuracy requirement with sufficiently high probability,
- and that function estimates at the current iterate and at a potential next iterate are sufficiently accurate with sufficiently high probability.

The main novelty of this work is the analysis of the framework and the resulting weaker, more general, conditions compared to prior work. In particular,

- we do not assume that the probabilities of obtaining sufficiently accurate models and estimates are increasing (they simply need to be above a certain constant) and
- we do not assume any distribution of the random models and estimates. In other words, if a model or estimate is inaccurate, it can be arbitrarily inaccurate, i.e. the noise in the function values can have nonconstant bias.

It is also important to note that while our framework and model requirements are borrowed from prior work on DFO, this framework applies to derivative-based optimization as well. Later in the paper we will discuss different settings which will fit into the proposed framework.

This paper consists of two main parts. In the first part we propose and analyze a trust region framework, which utilizes random models of $f(x)$ at each iteration to compute the next potential iterate. It also relies on (random, noisy) estimates of the function values at the current iterate and the potential iterate to gauge the progress that is being made. The convergence analysis then relies on requirements that these models and these estimates are sufficiently accurate with sufficiently high probability. Beyond these conditions, no assumptions are made about how these models and estimates are generated. The resulting method is a stochastic process that is analyzed with the help of martingale theory. The method is shown to converge to first order stationary points with probability one.

In the second part of the paper we consider various scenarios under different assumptions on the noise inducing component ε and discuss how sufficiently accurate random models can be generated. In particular, we show that in the case of unbiased noise, that is when $\mathbb{E}[f(x,\varepsilon)] = f(x)$ and $\mathrm{Var}[f(x,\varepsilon)] \leq \sigma^2 < \infty$ for all x, sample averaging techniques give us sufficiently accurate models. Although we will prove convergence under the mentioned framework that essentially says we have the ability to compute both sufficiently accurate models and estimates with constant, separate probabilities, it is not necessarily easy to estimate what these probabilities ought to be for a given problem. While we provide some guidance on the selection of sampling rates in an unbiased noise setting in Section 5, our numerical experiments show that the bounds on probabilities suggested by our theory to be necessary for almost sure convergence are far from tight.

We also discuss the case where $\mathbb{E}[f(x,\varepsilon)] \neq f(x)$, and where the noise bias may depend on x or on the method of computation of the function values. One simple setting, which is illustrative, is as follows. Suppose we have an objective function, which is computed by a numerical process, whose accuracy can be controlled (such as tightening a convergence criterion within this numerical process). Suppose now that this numerical process involves some random component (such as sampling from some large data and/or utilizing a randomized algorithm). It may be known that with sufficiently high probability this numerical process produces a sufficiently accurate function value - however, with some small (but nonzero) probability the numerical process may fail and hence no reasonable value is guaranteed. Moreover, such failures may become more likely as more

accurate computations are required (for instance because an upper bound on the total number of iterations is reached inside the numerical process). Hence the probability of failure may depend on the current iterate and state of the algorithm. Here we simply assume that such failures do not occur with probability higher than some constant (which will be specified in our analysis), conditioned on the past. However, we do not assume anything about the inaccurate function values. As we will demonstrate later in this paper, in this setting, $\mathbb{E}[f(x,\varepsilon)] \neq f(x)$.

1.1 Comparison with related work

There is a very large volume of work on SA and SG, most of which is quite different from our proposed analysis and method. However, we will mention a few works here that are most closely related to this paper and highlight the differences. The three methods existing in the literature we will compare with are Deng and Ferris [8], SPSA (simultaneous perturbations stochastic approximation) [23, 24], and SCSR (sampling controlled stochastic recursion) [12]. These three settings and methods are most closely related to our work because they all rely on using models of the objective function that can both incorporate second-order information and whose accuracy with respect to a "true" model can be dynamically adjusted. In particular, Deng and Ferris apply the trust-region model-based derivative free optimization method UOBYQA [19] in a setting of sample path optimization [21]. In [23] and [24], the author applies an approximate gradient descent and Newton method, respectively, with gradient and Hessian estimates computed from specially designed finite differencing techniques, with decaying finite differencing parameter. In [12] a very general scheme is presented, where various deterministic optimization algorithms are generalized as stochastic counterparts, with the stochastic component arising from the stochasticity of the models and the resulting step of the optimization algorithm. We now compare some key components of these three methods with those of our framework, which we hereforth refer to as STORM (STochastic Optimization with Random Models).

Deng and Ferris: The assumptions of the sample path setting are roughly as follows: on each iteration k, given a collection of points $X^k = \{x_1^k, \ldots, x_p^k\}$ one can compute noisy function values $f(x_1^k, \varepsilon^k), \ldots, f(x_p^k, \varepsilon^k)$. The noisy function values are assumed to be realizations of an unbiased estimator of true values $f(x_1^k), \ldots, f(x_p^k)$. Then, using multiple, say N_k, realizations of ε^k, average function values $f^{N_k}(x_1^k), \ldots, f^{N_k}(x_p^k)$ can be computed. A quadratic model $m_k^{N_k}(x)$ is then fit into these function values, and so a sequence of models $\{m_k^{N_k}(x)\}$ is created using a nondecreasing sequence of sampling rates $\{N_k\}$. The assumption on this sequence of models is that each of them satisfies a sufficient decrease condition (with respect to the true model of the true function f) with probability $1 - \alpha_k$, such that $\sum_{k=1}^{\infty} \alpha_k < \infty$. The trust region maintenance follows the usual scheme like that in UOBYQA, hence the steps taken by the algorithm can be increased or decreased depending on the observed improvement of the function estimates.

SPSA: The first order version of this method assumes that $f(x,\varepsilon)$ is an unbiased estimate of $f(x)$, and the second order version, 2SPSA, assumes that an unbiased estimate of $\nabla f(x)$, $g(x,\varepsilon) \in \mathbf{R}^n$, can be computed. Gradient (in the first order case) and Hessian (in the second order case) estimates are constructed using an interesting randomized finite differencing scheme. The finite difference step is assumed to be decaying to zero. An approximate steepest descent direction or approximate Newton direction are then constructed and a step of length t_k is taken along this direction. The sequence $\{t_k\}$ is assumed to be decaying in the usual Robbins-Monro way, that is $t_k \to 0$, $\sum_k t_k = \infty$. Hence, while no increase in accuracy of the models is assumed (they only

need to be accurate in expectation), the step size parameter and the finite differencing parameter need to be tuned. Decaying step sizes often lead to slow convergence, as has been observed often in stochastic optimization literature.

SCSR: This is a very general scheme which can include multiple optimization methods and sampling rates. The key ingredients of this scheme are a deterministic optimization method, and a stochastic variant that approximates it. The stochastic step (recursion) is assumed to be a sufficiently accurate approximation of the deterministic step with increasing probability (the probabilities of failure for each iteration are summable). This assumption is stronger than the one in this paper. In addition, another key assumption made for SCSR is that the iterates produced by the base deterministic algorithm converge to the unique optimal minimizer x^*. Not only we do not assume here that the minimizer/stationary point is unique, but we also do not assume a priori that the iterates form a convergent sequence, since they may not do so in a non convex setting, while every iterate subsequence converges to a stationary point.

STORM: Like the Deng and Ferris method, we utilize a trust-region, model-based framework, where the size of the trust region can be increased or decreased according to empirically observed function decrease and the size of the observed approximate gradients. The desired accuracy of the models is tied only to the trust region radius in our case, while for Deng and Ferris, it is tied to both the radius and the size of true model gradients (the second condition is harder to ensure). In either method, this desired accuracy is assumed to hold with some probability - in STORM, this probability remains constant throughout the progress of the algorithm, while for Deng and Ferris it has to converge to 1 sufficiently rapidly.

There are three major advantages to our results. First of all, in the case of unbiased noise, the sampling rate is directly connected to the desired accuracy of the estimates and the probability with which this accuracy is achieved. Hence, for the STORM method, the sampling rate may increase or decrease according to the trust region radius, eventually increasing only when necessary, i.e. when the noise become dominating. For all the other methods listed here, the sampling rate is assumed to increase monotonically. Secondly, in the case of biased noise, we can still prove convergence of our method, as long as the desired accuracy is achieved with a fixed probability. In other words, we allow for the noise to be arbitrarily large with a small, but fixed probability, on each iteration. This allows us to consider new models of noise which cannot be handled by any of the other methods discussed here. In addition, STORM incorporates first and second order models without change to the algorithm - the step size parameter (i.e., the trust region radius) and other parameters of the method are chosen almost identically to the standard practices of the trust region methods, which have proved to be very effective in practice for unconstrained nonlinear optimization. In Section 6 we show that the STORM method is very effective in different noise settings and is very robust with respect to sampling strategies.

Finally, we want to point to an unpublished work [15], which proposes a very similar method to the one in this paper. Both methods were developed based on the trust region DFO method with random models for deterministic functions analyzed in [2] and extended to the stochastic setting. Some of the assumptions in this paper were inspired by an early version of [15]. However, the assumptions and the analysis in [15] are quite different from the ones in this paper. In particular, they rely on the assumption that $f(x, \varepsilon)$ is an unbiased estimate of $f(x)$, hence their analysis does not extend to the biased case. Also they assume that the probability of having an accurate model at the k-th iteration is at least $1 - \alpha_k$, such that $\alpha_k \to 0$, while for our method this probability can remain bounded away from zero. Similarly, they assume that the probability of having accurate

function estimates at the k-th iteration also converges to 1 sufficiently rapidly, while in our case it is again constant. Their analysis, as a result, is very different from ours, and does not generalize to various stochastic settings (they only focus on the derivative free setting with additive noise). The one advantage of their method is that they do not need to put a restriction on acceptable step sizes, when the norm of the gradient of the model is small. We, on the other hand, impose such restriction in our method and use it in the proof of our main result. However, as we discuss later in the paper this restriction can be relaxed at the cost of more complex algorithm and analysis. In practice, we do not implement this restriction, hence our basic implementation is virtually identical to that in [15] except that we implement a variety of model building strategies, while only one such strategy (regression models based on randomly rotated orthogonal samples sets) is implemented in [15]. Thus we do not directly compare the empirical performance of our method with the method in [15] since we view them as more or less the same method.

We conclude this section by introducing some frequently used notations and their meanings. The rest of the paper is organized as follows. In Section 2 we introduce the trust region framework, followed by Section 3, where we discuss the requirements on our random models and function estimates. The main convergence results are presented in Section 4. In Section 5 we discuss various noise scenarios and how sufficiently accurate models and estimates can be constructed in these cases. Finally, we present computational experiments based on these various noise scenarios in Section 6.

Notations. Let $\|\cdot\|$ denote the Euclidean norm and $B(x, \Delta)$ denote the ball of radius Δ around x, i.e., $B(x, \Delta) : \{y : \|x - y\| \leq \Delta\}$. Ω denotes the probability sample space, according to the context, and a sample from that space is denoted by $\omega \in \Omega$. As a rule, when we describe a random process within the algorithmic framework, uppercase letters, e.g. the k-th iterate X_k, will denote random variables, while lowercase letters will denote realizations of the random variable, e.g. $x_k = X_k(\omega)$ is the k-th iterate for a particular realization of our algorithm.

We also list here, for convenience, several constants that are used in the paper to bound various quantities. These constants are denoted by κ with subscripts indicating quantities that they are meant to bound.

κ_{ef}	"error in the function value",
κ_{eg}	"error in the gradient",
κ_{Eef}	"expectation of the error in the function value",
κ_{fcd}	"fraction of Cauchy decrease",
κ_{bhm}	"bound on the Hessian of the models",
κ_{et}	"error in Taylor expansion".

2 Trust Region Method

We consider the trust-region class of methods for minimization of unconstrained functions. They operate as follows: at each iteration k, given the current iterate x_k and a trust-region radius δ_k, a (random) model $m_k(x)$ is built, which serves as an approximation of $f(x)$ in $B(x_k, \delta_k)$. The model is assumed to be of the form

$$m_k(x_k + s) = f_k + g_k^\top s + s^\top H_k s. \tag{2}$$

It is possible to generalize our framework to other forms of models, as long as all conditions on the models, described below, hold. We consider quadratic models for simplicity of the presentation and because they are the most common. The model $m_k(x)$ is minimized (approximately) in $B(x_k, \delta_k)$ to produce a step s_k and (random) estimates of $f(x_k)$ and $f(x_k + s_k)$ are obtained, denoted by f_k^0 and f_k^s respectively. The achieved reduction is measured by comparing f_k^0 and f_k^s and if reduction is deemed sufficient, then $x_k + s_k$ is chosen as the next iterate x_{k+1}. Otherwise the iterate remains fixed as x_k. The trust-region radius δ_{k+1} is then chosen by either increasing of decreasing δ_k according to the outcome of the iteration. The details of the algorithm are presented in Algorithm 1.

Algorithm 1 STOCHASTIC DFO WITH RANDOM MODELS

1: (Initialization): Choose an initial point x_0 and an initial trust-region radius $\delta_0 \in (0, \delta_{\max})$ with $\delta_{\max} > 0$. Choose constants $\gamma > 1$, $\eta_1 \in (0, 1)$, $\eta_2 > 0$; Set $k \leftarrow 0$.
2: (Model construction): Build a model $m_k(x_k + s) = f_k + g_k^\top s + s^\top H_k s$ that approximates $f(x)$ on $B(x_k, \delta_k)$ with $s = x - x_k$.
3: (Step calculation) Compute $s_k = \arg\min_{s:\|s\| \le \delta_k} m_k(s)$ (approximately) so that it satisfies condition (3).
4: (Estimates calculation) Obtain estimates f_k^0 and f_k^s of $f(x_k)$ and $f(x_k + s_k)$, respectively.
5: (Acceptance of the trial point): Compute $\rho_k = \dfrac{f_k^0 - f_k^s}{m_k(x_k) - m_k(x_k + s_k)}$.
If $\rho_k \ge \eta_1$ and $\|g_k\| \ge \eta_2 \delta_k$, set $x_{k+1} = x_k + s_k$; otherwise, set $x_{k+1} = x_k$.
6: (Trust-region radius update): If $\rho_k \ge \eta_1$ and $\|g_k\| \ge \eta_2 \delta_k$, set $\delta_{k+1} = \min\{\gamma \delta_k, \delta_{\max}\}$; otherwise $\delta_{k+1} = \gamma^{-1} \delta_k$; $k \leftarrow k + 1$ and go to step 2.

The trial step computed on each iteration has to provide sufficient decrease of the model; in other words it has to satisfy the following standard fraction of Cauchy decrease condition:

Assumption 2.1. *For every k, the step s_k is computed so that*

$$m_k(x_k) - m_k(x_k + s_k) \ge \frac{\kappa_{fcd}}{2} \|g_k\| \min\left\{ \frac{\|g_k\|}{\|H_k\|}, \delta_k \right\} \tag{3}$$

for some constant $\kappa_{fcd} \in (0, 1]$.

If progress is achieved and a new iterate is accepted in the k-th iteration then we call this a *successful iteration*. Otherwise, the iteration is unsuccessful (and no step is taken). Hence a successful iteration occurs when $\rho_k \ge \eta_1$ and $\|g_k\| \ge \eta_2 \delta_k$. However, a successful iteration does not necessarily yield an actual reduction in the true function f. This is because the values of $f(x)$ are not accessible in our stochastic setting and the step acceptance decision is made merely based on the estimates of $f(x_k)$ and $f(x_k + s_k)$. If these estimates, f_k^0 and f_k^s, are not accurate enough, a successful iteration can result in an increase of the true function value. Hence we consider two types of successful iterations - those where $f(x)$ is in fact decreased proportionally to $f_k^0 - f_k^s$, which we call *true* successful iterations, and all other successful iterations, where the decrease of $f(x)$ can be arbitrarily small or even negative, which we call *false* successful iterations. Our setting and algorithmic framework does not allow us to determine which successful iterations are true and which ones are false, however, we will be able to show that true successful iterations

occur sufficiently often for convergence to hold, if the random estimates f_k^0 and f_k^s are sufficiently accurate.

A trust region framework based on random models was introduced and analyzed in [2]. In that paper, the authors introduced the concept of probabilistically fully-linear models to determine the conditions that random models should satisfy for convergence of the algorithm to hold. However, the randomness in the models in their setting arises from the the construction process, and not from the noisy objective function. It is assumed in [2] that the function values at the current iterate and the trial point can be computed exactly and hence all successful iterations are true in that case. In our case, it is necessary to define a measure for the accuracy of the estimates f_k^0 and f_k^s (which, as we will see, generally has to be tighter than the measure of accuracy of the model). We will use a modified version of the probabilistic estimates introduced in [15].

3 Probabilistic Models and Estimates

The models in this paper are functions which are constructed on each iteration, based on some random samples of stochastic function $\tilde{f}(x)$. Hence, the models themselves are random and so is their behavior and influence on the iterations. Hence, M_k will denote a random model in the k-th iteration, while we will use the notation $m_k = M_k(\omega)$ for its realizations. As a consequence of using random models, the iterates X_k, the trust-region radii Δ_k and the steps S_k are also random quantities, and so $x_k = X_k(\omega)$, $\delta_k = \Delta_k(\omega)$, $s_k = S_k(\omega)$ will denote their respective realizations. Similarly, let random quantities $\{F_k^0, F_k^s\}$ denote the estimates of $f(X_k)$ and $f(X_k + S_k)$, with their realizations denoted by $f_k^0 = F_k^0(\omega)$ and $f_k^s = F_k^s(\omega)$. In other words, Algorithm 1 results in a stochastic process $\{M_k, X_k, S_k, \Delta_k, F_k^0, F_k^s\}$. Our goal is to show that under certain conditions on the sequences $\{M_k\}$ and $\{F_k^0, F_k^s\}$ the resulting stochastic process has desirable convergence properties with probability one. In particular, we will assume that models M_k and estimates F_k^0, F_k^s are sufficiently accurate with sufficiently high probability, conditioned on the past.

To formalize conditioning on the past, let $\mathcal{F}_{k-1}^{M\cdot F}$ denote the σ-algebra generated by $M_0, \cdots, M_{k-1}$ and $F_0, \cdots, F_{k-1}$ and let $\mathcal{F}_{k-1/2}^{M\cdot F}$ denote the σ-algebra generated by $M_0, \cdots, M_k$ and $F_0, \cdots, F_{k-1}$.

To formalize sufficient accuracy, let us recall a measure for the accuracy of deterministic models introduced in [7] and [6] (with the exact notation introduced in [3]).

Definition 3.1. *Suppose ∇f is Lipschitz continuous. A function m_k is a κ-*fully linear model of f on $B(x_k, \delta_k)$ *provided, for $\kappa = (\kappa_{ef}, \kappa_{eg})$ and $\forall y \in B$,*

$$\begin{aligned} \|\nabla f(y) - \nabla m_k(y)\| &\leq \kappa_{eg}\delta_k, \text{ and} \\ |f(y) - m_k(y)| &\leq \kappa_{ef}\delta_k^2. \end{aligned} \tag{4}$$

In this paper we extend the following concept of probabilistically fully-linear models which is proposed in [2].

Definition 3.2. *A sequence of random models $\{M_k\}$ is said to be α-probabilistically κ-fully linear with respect to the corresponding sequence $\{B(X_k, \Delta_k)\}$ if the events*

$$I_k = \{M_k \text{ is a } \kappa\text{-fully linear model of } f \text{ on } B(X_k, \Delta_k)\} \tag{5}$$

satisfy the condition

$$P(I_k | \mathcal{F}_{k-1}^M) \geq \alpha,$$

where $\mathcal{F}_{k-1}^M$ is the σ-algebra generated by $M_0, \cdots, M_{k-1}$.

These probabilistically fully-linear models have the very simple properties that they are fully-linear (i.e., accurate enough) with sufficiently high probability, conditioned on the past, and they can be arbitrarily inaccurate otherwise. This property is somewhat different from the properties of models typical to stochastic optimization (such as, for example, stochastic gradient based models), where assumptions on the expected value and the variance of the models is imposed. We will discuss this in more detail in Section 5.

In this paper, aside from sufficiently accurate models, we require estimates of the function values $f(x_k)$, $f(x_k + s_k)$ that are sufficiently accurate. This is needed in order to evaluate whether a step is successful, unlike the case in [2] where the exact values $f(x_k)$ and $f(x_k + s_k)$ are assumed to be available. The following definition of accurate estimates is a modified version of that used in [15].

Definition 3.3. *The estimates f_k^0 and f_k^s are said to be ϵ_F-accurate estimates of $f(x_k)$ and $f(x_k + s_k)$, respectively, for a given δ_k if*

$$|f_k^0 - f(x_k)| \leq \epsilon_F \delta_k^2 \text{ and } |f_k^s - f(x_k + s_k)| \leq \epsilon_F \delta_k^2. \tag{6}$$

We now modify Definitions 3.2 and 3.3 and introduce definitions of probabilistically accurate models and estimates which we will use throughout the remainder of the paper.

Definition 3.4. *A sequence of random models $\{M_k\}$ is said to be α-probabilistically κ-fully linear with respect to the corresponding sequence $\{B(X_k, \Delta_k)\}$ if the events*

$$I_k = \{M_k \text{ is a } \kappa\text{-fully linear model of } f \text{ on } B(X_k, \Delta_k)\} \tag{7}$$

satisfy the condition

$$P(I_k | \mathcal{F}_{k-1}^M) \geq \alpha,$$

where $\mathcal{F}_{k-1}^{M \cdot F}$ is the σ-algebra generated by $M_0, \cdots, M_{k-1}$ and $F_0, \cdots, F_{k-1}$.

Definition 3.5. *A sequence of random estimates $\{F_k^0, F_k^s\}$ is said to be β-probabilistically ϵ_F-accurate with respect to the corresponding sequence $\{X_k, \Delta_k, S_k\}$ if the events*

$$J_k = \{F_k^0, F_k^s \text{ are } \epsilon_F\text{-accurate estimates of } f(x_k) \text{ and } f(x_k + s_k), \text{ respectively, for } \Delta_k\} \tag{8}$$

satisfy the condition

$$P(J_k | \mathcal{F}_{k-1/2}^{M \cdot F}) \geq \beta,$$

where ϵ_F is a fixed constant and $\mathcal{F}_{k-1/2}^{M \cdot F}$ is the σ-algebra generated by $M_0, \cdots, M_k$ and $F_0, \cdots, F_{k-1}$.

As can be seen from Definitions 3.4 and 3.5 the accuracy of the models and estimates with respect to $f(x)$ is required to be proportional to δ_k^2. However, as will be evident from our analysis below, in the case of fully-linear models, all that is required is the existence of fixed constants κ_{ef} and κ_{eg} such that (4) holds, while in the case of ϵ_F-accurate estimates it is required that (6) holds for some given, small enough, ϵ_F. The latter, by comparison, is a tighter requirement. However, we will see that an upper bound on ϵ_F that is sufficient for convergence is reasonable.

Procedures for obtaining probabilistically fully-linear models and probabilistically accurate estimates under different models of noise are discussed in Section 5.

4 Convergence Analysis

We now present first-order convergence analysis for the general framework described in Algorithm 1. Towards that end, we assume that the function f and its gradient are Lipschitz continuous in regions considered by the algorithm realizations. We follow the process in [6] to define this region.

Assumption 4.1 (Assumptions on f)**.** *Let x_0 and $\delta_{\max}$ be given. Assume that f is bounded below on the level set*

$$L(x_0) = \{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}.$$

Assume also that the function f and its gradient ∇f are Lipschitz continuous on the set $L_{enl}(x_0)$, where $L_{enl}(x_0)$ defines the region considered by the algorithm realizations

$$L_{enl}(x_0) = \bigcup_{x \in L(x_0)} B(x; \delta_{\max}).$$

The second assumption provides a uniform upper bound on the model Hessian.

Assumption 4.2. *There exists a positive constant κ_{bhm} such that, for every k, the Hessian H_k of all realizations m_k of M_k satisfy*

$$\|H_k\| \leq \kappa_{bhm}.$$

Note that since we are concerned with convergence to a first order stationary point in this paper, the bound κ_{bhm} can be chosen to be any nonnegative number, including zero. Allowing a larger bound will give more flexibility to the algorithm and may allow better Hessian approximations, but as we will see in the convergence analysis, this imposes restrictions on the trust region radius and some other algorithmic parameters.

We now state the following result from martingale literature [9] that will be useful later in our analysis.

Theorem 4.3. *Let G_k be a submartingale, i.e., a sequence of random variables which, for every k,*

$$E[G_k | \mathcal{F}^G_{k-1}] \geq G_{k-1},$$

where $\mathcal{F}^G_{k-1} = \sigma(G_0, \cdots, G_{k-1})$ is the σ-algebra generated by $G_0, \cdots, G_{k-1}$ and $E[G_k|\mathcal{F}^G_{k-1}]$ denotes the conditional expectation of G_k given the past history of events $\mathcal{F}^G_{k-1}$.

Assume further that $G_k - G_{k-1} \leq M < \infty$, for every k. Then,

$$P\left(\left\{\lim_{k\to 0} G_k < \infty\right\} \cap \left\{\limsup_{k\to 0} G_k = \infty\right\}\right) = 1. \tag{9}$$

We now prove some auxiliary lemmas that provide conditions under which decrease of the true objective function $f(x)$ is guaranteed. The first lemma states that if the trust region radius is small enough relative to the size of the model gradient and if the model is fully linear, then the step s_k provides a decrease in $f(x)$ proportional to the size of the model gradient. Note that the trial step may still be rejected if the estimates f^0_k and f^s_k are not accurate enough.

Lemma 4.4. *Suppose that a model m_k of the form (2) is a $(\kappa_{ef}, \kappa_{eg})$-fully linear model of f on $B(x_k, \delta_k)$. If*

$$\delta_k \leq \min\left\{\frac{1}{\kappa_{bhm}}, \frac{\kappa_{fcd}}{8\kappa_{ef}}\right\}\|g_k\|,$$

then the trial step s_k leads to an improvement in $f(x_k + s_k)$ such that

$$f(x_k + s_k) - f(x_k) \leq -\frac{\kappa_{fcd}}{4}\|g_k\|\delta_k. \tag{10}$$

Proof. Using the Cauchy decrease condition, the upper bound on model Hessian and the fact that $\|g_k\| \geq \kappa_{bhm}\delta_k$, we have

$$m_k(x_k) - m_k(x_k + s_k) \geq \frac{\kappa_{fcd}}{2}\|g_k\| \min\left\{\frac{\|g_k\|}{\|H_k\|}, \delta_k\right\} = \frac{\kappa_{fcd}}{2}\|g_k\|\delta_k.$$

Since the model is κ-fully linear, one can express the improvement in f achieved by s_k as

$$\begin{aligned} & f(x_k + s_k) - f(x_k) \\ = \; & f(x_k + s_k) - m(x_k + s_k) + m(x_k + s_k) - m(x_k) + m(x_k) - f(x_k) \\ \leq \; & 2\kappa_{ef}\delta_k^2 - \frac{\kappa_{fcd}}{2}\|g_k\|\delta_k \\ \leq \; & -\frac{\kappa_{fcd}}{4}\|g_k\|\delta_k, \end{aligned}$$

where the last inequality is implied by $\delta_k \leq \frac{\kappa_{fcd}}{8\kappa_{ef}}\|g_k\|$. □

The next lemma shows that for δ_k small enough relative to the size of the true gradient $\nabla f(x_k)$, the guaranteed decrease in the objective function, provided by s_k, is proportional to the size of the true gradient.

Lemma 4.5. *Under Assumptions 4.1 and 4.2, suppose that a model is $(\kappa_{ef}, \kappa_{eg})$-fully linear on $B(x_k, \delta_k)$. If*

$$\delta_k \leq \min\left\{\frac{1}{\kappa_{bhm} + \kappa_{eg}}, \frac{1}{\frac{8\kappa_{ef}}{\kappa_{fcd}} + \kappa_{eg}}\right\}\|\nabla f(x_k)\|, \tag{11}$$

then the trial step s_k leads to an improvement in $f(x_k + s_k)$ such that

$$f(x_k + s_k) - f(x_k) \leq -C_1\|\nabla f(x_k)\|\delta_k, \tag{12}$$

where $C_1 = \frac{\kappa_{fcd}}{4} \cdot \max\left\{\frac{\kappa_{bhm}}{\kappa_{bhm}+\kappa_{eg}}, \frac{8\kappa_{ef}}{8\kappa_{ef}+\kappa_{fcd}\kappa_{eg}}\right\}$.

Proof. The definition of a κ-fully-linear model yields that

$$\|g_k\| \geq \|\nabla f(x)\| - \kappa_{eg}\delta_k.$$

Since condition (11) implies that $\|\nabla f(x_k)\| \geq \max\left\{\kappa_{bhm} + \kappa_{eg}, \frac{8\kappa_{ef}}{\kappa_{fcd}} + \kappa_{eg}\right\}\delta_k$, we have

$$\|g_k\| \geq \max\left\{\kappa_{bhm}, \frac{8\kappa_{ef}}{\kappa_{fcd}}\right\}\delta_k.$$

Hence, the conditions of Lemma 4.4 hold and we have

$$f(x_k + s_k) - f(x_k) \leq -\frac{\kappa_{fcd}}{4}\|g_k\|\delta_k. \tag{13}$$

Since $\|g_k\| \geq \|\nabla f(x)\| - \kappa_{eg}\delta_k$ in which δ_k satisfies (11), we also have

$$\|g_k\| \geq \max\left\{\frac{\kappa_{bhm}}{\kappa_{bhm} + \kappa_{eg}}, \frac{8\kappa_{ef}}{8\kappa_{ef} + \kappa_{fcd}\kappa_{eg}}\right\}\|\nabla f(x_k)\|. \tag{14}$$

Combining (13) and (14) yields (12). □

We now prove the lemma that states that, if a) the estimates are sufficiently accurate, b) the model is fully-linear and c) the trust-region radius is sufficiently small relatively to the size of the model gradient, then a successful step is guaranteed.

Lemma 4.6. *Under Assumptions 4.1 and 4.2, suppose that m_k is $(\kappa_{ef}, \kappa_{eg})$-fully linear on $B(x_k, \delta_k)$ and the estimates $\{f_k^0, f_k^s\}$ are ϵ_F-accurate with $\epsilon_F \leq \kappa_{ef}$. If*

$$\delta_k \leq \min\left\{\frac{1}{\kappa_{bhm}}, \frac{1}{\eta_2}, \frac{\kappa_{fcd}(1-\eta_1)}{8\kappa_{ef}}\right\}\|g_k\|, \tag{15}$$

then the k-th iteration is successful.

Proof. Since $\delta_k \leq \frac{\|g_k\|}{\kappa_{bhm}}$, the Cauchy decrease condition and the uniform bound on H_k immediately yield that

$$m_k(x_k) - m_k(x_k + s_k) \geq \frac{\kappa_{fcd}}{2}\|g_k\| \min\left\{\frac{\|g_k\|}{\kappa_{bhm}}, \delta_k\right\} = \frac{\kappa_{fcd}}{2}\|g_k\|\delta_k. \tag{16}$$

The model m_k being $(\kappa_{ef}, \kappa_{eg})$-fully linear implies that

$$|f(x_k) - m_k(x_k)| \leq \kappa_{ef}\delta_k^2, \text{ and} \tag{17}$$

$$|f(x_k + s_k) - m_k(x_k + s_k)| \leq \kappa_{ef}\delta_k^2. \tag{18}$$

Since the estimates are ϵ_F-accurate with $\epsilon_F \leq \kappa_{ef}$, we obtain

$$|f_k^0 - f(x_k)| \leq \kappa_{ef}\delta_k^2, \text{ and } |f_k^s - f(x_k + s_k)| \leq \kappa_{ef}\delta_k^2. \tag{19}$$

Combining (16)-(19), we have

$$\begin{aligned}
\rho_k &= \frac{f_k^0 - f_k^s}{m_k(x_k) - m_k(x_k + s_k)} \\
&= \frac{f_k^0 - f(x_k)}{m_k(x_k) - m_k(x_k + s_k)} + \frac{f(x_k) - m_k(x_k)}{m_k(x_k) - m_k(x_k + s_k)} + \frac{m_k(x_k) - m_k(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)} \\
&\quad + \frac{m_k(x_k + s_k) - f(x_k + s_k)}{m_k(x_k) - m_k(x_k + s_k)} + \frac{f(x_k + s_k) - f_k^s}{m_k(x_k) - m_k(x_k + s_k)},
\end{aligned}$$

which indicates that

$$|\rho_k - 1| \leq \frac{8\kappa_{ef}\delta_k^2}{\kappa_{fcd}\|g_k\|\delta_k} \leq 1 - \eta_1,$$

where we have used the assumption $\delta_k \leq \frac{\kappa_{fcd}(1-\eta_1)}{8\kappa_{ef}}\|g_k\|$ to deduce the last inequality. Hence, $\rho_k \geq \eta_1$. Moreover, since $\|g_k\| \geq \eta_2\delta_k$, the k-th iteration is successful. □

Finally, we state and prove the lemma which guarantees an amount of decrease of the objective function on a true successful iteration.

Lemma 4.7. *Under Assumptions 4.1 and 4.2, suppose that the estimates $\{f_k^0, f_k^s\}$ are ϵ_F-accurate with $\epsilon_F < \frac{1}{4}\eta_1\eta_2\kappa_{fcd}\min\left\{\frac{\eta_2}{\kappa_{bhm}}, 1\right\}$. If a trial step s_k is accepted (a successful iteration occurs), then the improvement in f is bounded below as follows*

$$f(x_{k+1}) - f(x_k) \leq -C_2\delta_k^2, \tag{20}$$

where $C_2 = \frac{1}{2}\eta_1\eta_2\kappa_{fcd}\min\left\{\frac{\eta_2}{\kappa_{bhm}}, 1\right\} - 2\epsilon_F > 0$.

Proof. An iteration being successful indicates that $\|g_k\| \geq \eta_2\delta_k$ and $\rho \geq \eta_1$. Thus,

$$\begin{aligned} f_k^0 - f_k^s &\geq \eta_1(m_k(x_k) - m_k(x_k + s_k)) \\ &\geq \eta_1\frac{\kappa_{fcd}}{2}\|g_k\|\min\left\{\frac{\|g_k\|}{\|H_k\|}, \delta_k\right\} \\ &\geq \frac{1}{2}\eta_1\eta_2\kappa_{fcd}\min\left\{\frac{\eta_2}{\kappa_{bhm}}, 1\right\}\delta_k^2. \end{aligned}$$

Then, since the estimates are ϵ_F-accurate, we have that the improvement in f can be bounded as

$$f(x_k + s_k) - f(x_k) = f(x_k + s_k) - f_k^s + f_k^s - f_k^0 + f_k^0 - f(x_k) \leq -C_2\delta_k^2,$$

where $C_2 = \frac{1}{2}\eta_1\eta_2\kappa_{fcd}\min\left\{\frac{\eta_2}{\kappa_{bhm}}, 1\right\} - 2\epsilon_F > 0$. □

To prove convergence of Algorithm 1 we will need to assume that models $\{M_k\}$ and estimates $\{F_k^0, F_k^s\}$ are sufficiently accurate with sufficiently high probability.

Assumption 4.8. *Given values of $\alpha, \beta \in (0, 1)$ and $\epsilon_F > 0$, there exist κ_{eg} and κ_{ef} such that the the sequence of models $\{M_k\}$ and estimates $\{F_k^0, F_k^s\}$ generated by Algorithm 1 are, respectively, α-probabilistically $(\kappa_{ef}, \kappa_{eg})$- fully-linear and β-probabilistically ϵ_F-accurate.*

Remark 4.9. *Note that this assumption is a statement about the existence of constants $\kappa = (\kappa_{ef}, \kappa_{eg})$ given an α, β and ϵ_F - we will determine exact conditions on α, β and ϵ_F in Lemma 4.11.*

The following theorem states that the trust-region radius converges to zero with probability 1.

Theorem 4.10. *Let Assumptions 4.1 and 4.2 be satisfied and assume that $\eta_2 > \kappa_{bhm}$ in Algorithm 1. Then $\alpha, \beta, \epsilon_F$ can be chosen so that, if Assumption 4.8 holds for these values, then the sequence of trust-region radii, $\{\Delta_k\}$, generated by Algorithm 1 satisfies*

$$\sum_{k=0}^{\infty}\Delta_k^2 < \infty \tag{21}$$

almost surely.

Proof. We base our proof on properties of the following random function $\Phi_k = \nu f(X_k) + (1-\nu)\Delta_k^2$, where $\nu \in (0,1)$ is a fixed constant, which will be specified later. A similar function is used in the analysis in [15], but the analysis itself is different. The overall goal is to show that there exists a constant $\sigma > 0$ such that for all k,

$$E[\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}^{M \cdot F}] \leq -\sigma \Delta_k^2 < 0. \tag{22}$$

Since f is bounded from below and $\Delta_k > 0$, we have that Φ_k is bounded from below for all k and hence if (22) holds on every iteration, then by summing (22) over $k \in (1, \infty)$ and taking expectations on both sides we can conclude that (21) holds with probability 1. Hence, to prove the theorem we need to show that (22) holds on each iteration.

Let us pick some constant ζ which satisfies

$$\zeta \geq \kappa_{eg} + \max\left\{\eta_2, \frac{8\kappa_{ef}}{\kappa_{fcd}(1-\eta_1)}\right\}.$$

We now consider two possible cases: $\|\nabla f(x_k)\| \geq \zeta\delta_k$ and $\|\nabla f(x_k)\| < \zeta\delta_k$. We will show that (22) holds in both cases and hence it holds on every iteration.

As usual, let x_k, δ_k, s_k, g_k and ϕ_k denote realizations of random quantities X_k, Δ_k, S_k, G_k and Φ_k, respectively.

Let us consider some realization of Algorithm 1. Note that on all successful iterations, $x_{k+1} = x_k + s_k$ and $\delta_{k+1} = \gamma\delta_k$ with $\gamma > 1$, hence

$$\phi_{k+1} - \phi_k = \nu(f(x_{k+1}) - f(x_k)) + (1-\nu)(\gamma^2 - 1)\delta_k^2. \tag{23}$$

On all unsuccessful iterations, $x_{k+1} = x_k$ and $\delta_{k+1} = \frac{1}{\gamma}\delta_k$, i.e.

$$\phi_{k+1} - \phi_k = (1-\nu)(\frac{1}{\gamma^2} - 1)\delta_k^2 \equiv b_1 < 0. \tag{24}$$

For each iteration and each of the two cases we consider, we will analyze the four possible combined outcomes of the events I_k and J_k as defined in (7) and (8) respectively.

Before presenting the formal proof let us outline the key ideas. We will show that, unless both the model and the estimates are bad on iteration k, we can select $\nu \in (0,1)$ sufficiently close to 1, so that the decrease in ϕ_k on a successful iteration is greater than the decrease on an unsuccessful iteration (which is equal to b_1, according to (24)). When the model and the estimates are both bad, an increase in ϕ_k may occur. This increase is bounded by something proportional to δ_k^2 when $\|\nabla f(x_k)\| < \zeta\delta_k$. When $\|\nabla f(x_k)\| \geq \zeta\delta_k$, though, the increase in ϕ_k may be proportional to $\|\nabla f(x_k)\|\delta_k$. However, since iterations with good models and good estimates will provide *decrease* in ϕ_k also proportional to $\|\nabla f(x_k)\|\delta_k$, by choosing values of α and β close enough to 1, we can ensure that in expectation ϕ_k decreases.

We now present the proof.

Case 1: $\|\nabla f(x_k)\| \geq \zeta\delta_k$**.**

a. I_k and J_k are both true, i.e., both the model and the estimates are good on iteration k. Assume

$$\epsilon_F \leq \kappa_{ef}. \tag{25}$$

From the definition of ζ, we know

$$\|\nabla f(x_k)\| \geq \left(\kappa_{eg} + \max\left\{\eta_2, \frac{8\kappa_{ef}}{\kappa_{fcd}(1-\eta_1)}\right\}\right)\delta_k.$$

Then since the model m_k is κ-fully linear and, from $\eta_2 > \kappa_{bhm}$ and $0 < \eta_1 < 1$, it is easy to show that the condition (11) in Lemma 4.5 holds. Therefore, the trial step s_k leads to a decrease in f as in (12).

Moreover, since

$$\|g_k\| \geq \|\nabla f(x_k)\| - \kappa_{eg}\delta_k \geq (\zeta - \kappa_{eg})\delta_k \geq \max\left\{\eta_2, \frac{8\kappa_{ef}}{\kappa_{fcd}(1-\eta_1)}\right\}\delta_k$$

and the estimates $\{f_k^0, f_k^s\}$ are ϵ_F-accurate, with $\epsilon_F \leq \kappa_{ef}$, the condition (15) in Lemma 4.6 holds. Hence, iteration k is successful, i.e. $x_{k+1} = x_k + s_k$ and $\delta_{k+1} = \gamma\delta_k$.

Combining (12) and (23), we get

$$\phi_{k+1} - \phi_k \leq -\nu C_1\|\nabla f(x_k)\|\delta_k + (1-\nu)(\gamma^2 - 1)\delta_k^2 \equiv b_2, \tag{26}$$

with C_1 defined in Lemma 4.5. Since $\|\nabla f(x_k)\| \geq \zeta\delta_k$ we have

$$b_2 \leq [-\nu C_1\zeta + (1-\nu)(\gamma^2 - 1)]\delta_k^2 < 0, \tag{27}$$

for $\nu \in (0,1)$ large enough, such that

$$\frac{\nu}{1-\nu} > \frac{\gamma^2 - 1}{\zeta C_1}. \tag{28}$$

b. I_k is true and J_k is false, i.e., we have a good model and bad estimates on iteration k.

In this case, Lemma 4.5 still holds, that is s_k yields a sufficient decrease in f, hence, if the iteration is successful, we obtain (26) and (27). However, the step can be erroneously rejected, because of inaccurate probabilistic estimates, in which case we have an unsuccessful iteration and (24) holds. By choosing $\nu \in (0,1)$ large enough so that

$$\frac{\nu}{1-\nu} > \frac{\gamma^2 - 1/\gamma^2}{\zeta C_1}, \tag{29}$$

we ensure that the right hand side of (27) is strictly smaller than the right hand side of (24) and therefore, (24) holds whether the iteration is successful or not.

c. I_k is false and J_k is true, i.e., we have a bad model and good estimates on iteration k.

Assume that

$$\epsilon_F < \frac{1}{8}\eta_1\eta_2\kappa_{fcd}. \tag{30}$$

In this case, iteration k can be either successful or unsuccessful. In the unsuccessful case (24) holds. When the iteration is successful, since the estimates are ϵ_F-accurate and (30) holds then by Lemma 4.7 (20) holds with $C_2 \geq \frac{1}{4}\eta_1\eta_2\kappa_{fcd}$. Hence, in this case we have

$$\phi_{k+1} - \phi_k \leq [-\nu C_2 + (1-\nu)(\gamma^2 - 1)]\delta_k^2. \tag{31}$$

Choosing $\nu \in (0,1)$ to satisfy

$$\frac{\nu}{1-\nu} \geq \frac{\gamma^2 - 1/\gamma^2}{C_2} \tag{32}$$

we have that, as in case (b), (24) holds whether the iteration is successful or not.

d. I_k and J_k are both false, i.e., both the model and the estimates are bad on iteration k.

Inaccurate estimates can cause the algorithm to accept a bad step, which may lead to an increase both in f and in δ_k. Hence in this case $\phi_{k+1} - \phi_k$ may be positive. However, combining the Taylor expansion of $f(x_k)$ at $x_k + s_k$ and the Lipschitz continuity of $\nabla f(x)$ we can bound the amount of increase in f, hence bounding $\phi_{k+1} - \phi_k$ from above. By adjusting the probability of outcome (d) to be sufficiently small, we can ensure that in expectation Φ_k is sufficiently reduced.

In particular, from Taylor's Theorem and Lipschitz continuity of $\nabla f(x)$ we have, respectively,

$$\begin{aligned} f(x_k) - f(x_k + s_k) &\geq \nabla f(x_k + s_k)^T(-s_k) - \frac{1}{2}L_1\delta_k^2, \text{ and} \\ \|\nabla f(x_k + s_k) - \nabla f(x_k)\| &\leq L_1 s_k \leq L_1\delta_k. \end{aligned}$$

From this we can derive that any increase of $f(x_k)$ is bounded by

$$f(x_k + s_k) - f(x_k) \leq C_3\|\nabla f(x_k)\|\delta_k,$$

where $C_3 = 1 + \frac{3L_1}{2\zeta}$. Hence, the change in function ϕ is bounded as follows

$$\phi_{k+1} - \phi_k \leq \nu C_3\|\nabla f(x_k)\|\delta_k + (1-\nu)(\gamma^2 - 1)\delta_k^2 \equiv b_3. \tag{33}$$

Now we are ready to take the expectation of $\Phi_{k+1} - \Phi_k$ for the case when $\|\nabla f(X_k)\| \geq \zeta\Delta_k$. We know that case (a) occurs with probability at least $\alpha\beta$ (conditioned on the past) and in that case $\phi_{k+1} - \phi_k = b_2 < 0$ with b_2 defined in (26), case (d) occurs with probability at most $(1-\alpha)(1-\beta)$ and that case $\phi_{k+1} - \phi_k$ is bounded from above by $b_3 > 0$, and cases (b) and (c) occur otherwise and in those cases $\phi_{k+1} - \phi_k$ is bounded from above by $b_1 < 0$, with b_1 defined in (24). Finally we note that $b_1 > b_2$ due to our choice of ν.

Hence, we can combine (24), (26), (31) and (33) and use B_1, B_2 and B_3 as random counterparts of b_1, b_2 and b_3, to obtain the following bound

$$\begin{aligned} & E[\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}^{M\cdot F}, \{\|\nabla f(X_k)\| \geq \zeta\Delta_k\}] \\ \leq\ & \alpha\beta B_2 + [\alpha(1-\beta) + (1-\alpha)\beta]B_1 + (1-\alpha)(1-\beta)B_3 \\ =\ & \alpha\beta[-\nu C_1\|\nabla f(X_k)\|\Delta_k + (1-\nu)(\gamma^2-1)\Delta_k^2] \\ & +[\alpha(1-\beta) + (1-\alpha)\beta](1-\nu)(\frac{1}{\gamma^2} - 1)\Delta_k^2 \\ & +(1-\alpha)(1-\beta)[\nu C_3\|\nabla f(X_k)\|\Delta_k + (1-\nu)(\gamma^2-1)\Delta_k^2]. \end{aligned}$$

Rearranging the terms we obtain

$$\begin{aligned} & E[\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}^{M \cdot F}, \{\|\nabla f(X_k)\| \geq \zeta \Delta_k\}] \\ \leq \; & [-\nu C_1 \alpha\beta + (1-\alpha)(1-\beta)\nu C_3] \|\nabla f(X_k)\| \Delta_k \\ & + [\alpha\beta - \frac{1}{\gamma^2}(\alpha(1-\beta) + (1-\alpha)\beta) + (1-\alpha)(1-\beta)](1-\nu)(\gamma^2-1)\Delta_k^2 \\ \leq \; & [-C_1\alpha\beta + (1-\alpha)(1-\beta)C_3]\nu\|\nabla f(X_k)\|\Delta_k + (1-\nu)(\gamma^2-1)\Delta_k^2, \end{aligned}$$

where the last inequality holds because $\alpha\beta - \frac{1}{\gamma^2}(\alpha(1-\beta) + (1-\alpha)\beta) + (1-\alpha)(1-\beta) \leq [\alpha + (1-\alpha)][(\beta + (1-\beta)] = 1$.

Recall that $\|\nabla f(X_k)\| \geq \zeta\Delta_k$, Hence if we choose $0 < \alpha \leq 1$ and $0 < \beta \leq 1$ so that they satisfy

$$\frac{\alpha\beta}{(1-\alpha)(1-\beta)} \;\; \geq \;\; \frac{\frac{2(1-\nu)}{\nu} \cdot \frac{\gamma^2-1}{\zeta} + C_3}{C_1} \tag{34}$$

which implies

$$[C_1\alpha\beta - (1-\alpha)(1-\beta)C_3] > 2\frac{(1-\nu)(\gamma^2-1)}{\nu\zeta},$$

we have

$$E[\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}^{M \cdot F}, \{\|\nabla f(X_k)\| \geq \zeta\Delta_k\}] \leq -[C_1\alpha\beta - (1-\alpha)(1-\beta)C_3]\nu\|\nabla f(X_k)\|\Delta_k. \tag{35}$$

and

$$E[\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}^{M \cdot F}, \{\|\nabla f(X_k)\| \geq \zeta\Delta_k\}] \leq -(1-\nu)(\gamma^2-1)\Delta_k^2. \tag{36}$$

For the purposes of this lemma and the lim inf-type convergence result, which will follow, bound (36) is sufficient. We will use bound (35) in the proof of the lim-type convergence result.

Case 2: Let us consider now the iterations when $\|\nabla f(x_k)\| < \zeta\delta_k$. First we note that if $\|g_k\| < \eta_2\delta_k$, then we have an unsuccessful step and (24) holds. Hence, we now assume that $\|g_k\| \geq \eta_2\delta_k$ and again consider four possible outcomes. We will show that in all situations, except when both the model and the estimates are bad, (24) holds. In the remaining case, because $\|\nabla f(x_k)\| < \zeta\delta_k$ the increase in ϕ_k can be bounded from above by a multiple of δ_k^2. Hence by selecting appropriate values for probabilities α and β we will be able to establish the bound on expected decrease in Φ_k as in Case 1.

a. I_k and J_k are both true, i.e., both the model and the estimates are good on iteration k.

   The iteration may or may not be successful, even though I_k is true. On successful iteration good model ensures reduction in f. Applying the same argument as in the case 1(c) we establish (24).

b. I_k is true and J_k is false, i.e., we have a good model and bad estimates on iteration k.

   On unsuccessful iterations, (24) holds. On successful iterations, $\|g_k\| \geq \eta_2\delta_k$ and $\eta_2 \geq \kappa_{bhm}$ imply that

$$\begin{aligned} m_k(x_k) - m_k(x_k + s_k) \;\; & \geq \;\; \frac{\kappa_{fcd}}{2}\|g_k\| \min\left\{\frac{\|g_k\|}{\|H_k\|}, \delta_k\right\} \\ & \geq \;\; \eta_2\frac{\kappa_{fcd}}{2}\delta_k^2. \end{aligned}$$

Since I_k is true, the model is κ-fully-linear, and the function decrease can be bounded as

$$\begin{aligned} & f(x_k) - f(x_k + s_k) \\ = \; & f(x_k) - m_k(x_k) + m_k(x_k) - m_k(x_k + s_k) + m_k(x_k + s_k) - f(x_k + x_k) \\ \geq \; & (\eta_2 \frac{\kappa_{fcd}}{2} - 2\kappa_{ef})\delta_k^2 \geq \kappa_{ef}\delta_k^2 \end{aligned}$$

as long as

$$\eta_2 \geq \frac{6\kappa_{ef}}{\kappa_{fcd}}. \tag{37}$$

It follows that, if k-th iterate is successful, then

$$\phi_{k+1} - \phi_k \leq [-\nu\kappa_{ef} + (1-\nu)(\gamma^2 - 1)]\delta_k^2. \tag{38}$$

Again choosing $\nu \in (0, 1)$ large enough so that

$$\frac{\nu}{1-\nu} > \frac{\gamma^2 - 1/\gamma^2}{\kappa_{ef}}, \tag{39}$$

we ensure that right hand side of (38) is strictly smaller than that of (24), hence (24) holds, whether the iteration is successful or not.

Remark: η_2 may need to be a relatively large constant to satisfy (37). This is due to the fact that the model has to be sufficiently accurate to ensure decrease in the function if a step is taken, since the step is accepted based on poor estimates. Note that η_2 restricts the size of Δ_k, which is used both as a bound on the step size and the control of the accuracy. In general it is possible to have two separate quantities (related by a constant) - one to control the step size and another to control the accuracy. Hence, it is possible to modify our algorithm to accept steps larger than $\|g_k\|/\eta_2$. This will make the algorithm more practical, but the analysis much more complex. In this paper, we choose to stay with the simplest version, but keep in mind that the condition (37) is not terminally restrictive.

c. I_k is false and J_k is true, i.e., we have a bad model and good estimates on iteration k.

This case is analyzed identically to the case 1(c).

d. I_k and J_k are both false, i.e., both the model and the estimates are bad on iteration k.

Here we bound the maximum possible increase in ϕ_k using the Taylor expansion and the Lipschitz continuity of $\nabla f(x)$.

$$f(x_k + s_k) - f(x_k) \leq \|\nabla f(x_k)\|\delta_k + \frac{1}{2}L_1\delta_k^2 < C_3\zeta\delta_k^2.$$

Hence, the change in function ϕ is

$$\phi_{k+1} - \phi_k \leq [\nu C_3\zeta + (1-\nu)(\gamma^2 - 1)]\delta_k^2. \tag{40}$$

We are now ready to bound the expectation of $\phi_{k+1} - \phi_k$ as we did in Case 1, except that in Case 2 we simply combine (40), which holds with probability at most $(1-\alpha)(1-\beta)$ and (24) which holds otherwise.

$$\begin{aligned} & E[\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}^{M \cdot F}, \{\|\nabla f(X_k)\| < \zeta \Delta_k\}] \\ \leq \; & [\alpha\beta + \alpha(1-\beta) + (1-\alpha)\beta](1-\nu)(\frac{1}{\gamma^2} - 1)\Delta_k^2 \\ & + (1-\alpha)(1-\beta)[\nu C_3 \zeta + (1-\nu)(\gamma^2 - 1)]\Delta_k^2. \end{aligned}$$

if we choose probabilities $0 < \alpha \leq 1$ and $0 < \beta \leq 1$ so that the following holds,

$$\frac{1 - (1-\alpha)(1-\beta)(1+\gamma^2)}{(1-\alpha)(1-\beta)} \geq \frac{2\nu\gamma^2 C_3 \zeta}{(1-\nu)(\gamma^2 - 1)}, \tag{41}$$

that is,

$$(1-\alpha)(1-\beta) \leq \frac{\gamma^2 - 1}{\gamma^4 - 1 + 2\gamma^2 C_3 \zeta \cdot \frac{\nu}{1-\nu}}, \tag{42}$$

then

$$E[\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}^{M \cdot F}, \{\|\nabla f(X_k)\| < \zeta \Delta_k\}] \leq -\nu C_3 \zeta \Delta_k^2. \tag{43}$$

In conclusion, combining (36) and (43), we have

$$E[\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}^{M \cdot F}\}] \leq -\min\{(1-\nu)(\gamma^2 - 1), \nu C_3 \zeta)\Delta_k^2 < 0,$$

which concludes the proof of the theorem, given that $0 < \nu < 1$, $\gamma^2 > 1$ and $C_3 > 0$.

□

To summarize the conditions on all of the constants involved in Theorem 4.10 to ensure that the theorem holds, we state the following additional lemma.

Lemma 4.11. *Let all assumptions of Theorem 4.10 hold. The statement of Theorem 4.10 holds if the parameters are chosen to satisfy the following conditions:*

- *The step acceptance parameter η_2 is chosen so that*

$$\eta_2 \;\; \geq \;\; \max\left\{\kappa_{bhm}, \frac{8\kappa_{ef}}{\kappa_{fcd}(1-\eta_1)}\right\}. \tag{44}$$

- *The accuracy parameter of the estimates satisfies*

$$\epsilon_F \;\; \leq \;\; \min\left\{\kappa_{ef}, \frac{1}{8}\eta_1\eta_2\kappa_{fcd}\right\}. \tag{45}$$

- *The probability parameters of the model and estimates, α, β respectively, satisfy*

$$\frac{\alpha\beta}{(1-\alpha)(1-\beta)} \geq \frac{1 + \frac{8(\gamma^2-1)+3L_1}{2\zeta}}{C_1} \tag{46}$$

and

$$(1-\alpha)(1-\beta) \leq \frac{\gamma^2-1}{\gamma^4-1+\gamma^4\left(3L_1+2\zeta\right)\cdot\max\left\{\frac{1}{\zeta C_1},\frac{1}{2\eta_1\kappa_{ef}},\frac{1}{2}\right\}}, \tag{47}$$

with $C_1 = \frac{\kappa_{fcd}}{4}\cdot\max\left\{\frac{\kappa_{bhm}}{\kappa_{bhm}+\kappa_{eg}},\frac{8\kappa_{ef}}{8\kappa_{ef}+\kappa_{fcd}\kappa_{eg}}\right\}$ *and* $\zeta \geq \kappa_{eg}+\eta_2$.

Proof. Consider the proof of Theorem 4.10. Condition (45) follows from conditions (25) and (30).

Condition (44) implies that $\eta_2 \geq \kappa_{bhm}$ and condition (37) hold, since $\eta_1 < 1$. Moreover, we now have

$$\zeta \geq \kappa_{eg} + \max\left\{\eta_2, \frac{8\kappa_{ef}}{\kappa_{fcd}(1-\eta_1)}\right\} \geq \kappa_{eg}+\eta_2. \tag{48}$$

Under condition (45) we have,

$$C_2 = \frac{1}{2}\eta_1\eta_2\kappa_{fcd} - 2\epsilon_F \geq \frac{1}{4}\eta_1\eta_2\kappa_{fcd} \geq 2\eta_1\kappa_{ef}. \tag{49}$$

Hence, combining conditions (28), (29), (32) and (39) with (48) and (49) with $\gamma > 1$, we derive

$$\begin{aligned}\frac{\nu}{1-\nu} &> \max\left\{\frac{\gamma^2-1}{\zeta C_1},\frac{\gamma^2-1/\gamma^2}{\zeta C_1},\frac{\gamma^2-1/\gamma^2}{C_2},\frac{\gamma^2-1/\gamma^2}{\kappa_{ef}}\right\}\\ &> \max\left\{\frac{\gamma^2}{\zeta C_1},\frac{\gamma^2}{2\eta_1\kappa_{ef}},\frac{\gamma^2}{\kappa_{ef}}\right\} \geq \max\left\{\frac{\gamma^2}{\zeta C_1},\frac{\gamma^2}{2\eta_1\kappa_{ef}}\right\},\end{aligned}$$

the last inequality holding under the assumption that $\eta_1 \leq 1/2$ which is the usual choice in trust region methods.

Now we are ready to derive conditions on α and β which imply (34) and (42) with $C_1 = \frac{\kappa_{fcd}}{4}\cdot\max\left\{\frac{\kappa_{bhm}}{\kappa_{bhm}+\kappa_{eg}},\frac{8\kappa_{ef}}{8\kappa_{ef}+\kappa_{fcd}\kappa_{eg}}\right\}$ and $C_3 = 1+\frac{3L_1}{2\zeta}$. Firstly, if α and β satisfy (46) then

$$\begin{aligned}\frac{\alpha\beta}{(1-\alpha)(1-\beta)} &\geq \frac{1+\frac{8(\gamma^2-1)+3L_1}{2\zeta}}{C_1}\\ &\geq \frac{4\cdot\frac{\gamma^2-1}{\zeta}+C_3}{C_1}\\ &\geq \frac{\frac{2(1-\nu)}{\nu}\cdot\frac{\gamma^2-1}{\zeta}+C_3}{C_1},\end{aligned}$$

last inequality following from $\frac{1-\nu}{\nu} < 2$. Hence (34) holds.

Condition (42) with $C_3 = 1+\frac{3L_1}{2\zeta}$ and

$$\frac{\nu}{1-\nu} > \max\left\{\frac{1}{\zeta C_1},\frac{1}{2\eta_1\kappa_{ef}},\frac{1}{2}\right\}$$

can be rewritten as

$$\begin{aligned}(1-\alpha)(1-\beta) &= \frac{\gamma^2-1}{\gamma^4-1+2\gamma^4 C_3\zeta\cdot\frac{\nu}{1-\nu}}\\ &\leq \frac{\gamma^2-1}{\gamma^4-1+\gamma^4\left(3L_1+2\zeta\right)\cdot\max\left\{\frac{1}{\zeta C_1},\frac{1}{2\eta_1\kappa_{ef}},\frac{1}{2}\right\}}.\end{aligned}$$

Hence, we obtain (47).

□

Clearly, choosing α and β sufficiently close to 1 will satisfy this condition.

Remark 4.12. *We will briefly illustrate through a simple example how these algorithmic parameters scale with problem data.*

Suppose $L = \max\{L_0, L_1\}$ is attained as L_1, that is the Lipschitz constant of the gradient of f is larger than the Lipschitz constant of f over $L_{enl}(x^0)$. It is reasonable to expect that κ_{ef} and κ_{eg} are quantities that scale with L, since Taylor models satisfy this condition, as do polynomial interpolation and regression models based on well-poised data sets [7]. Let us assume for the sake of an example that $\kappa_{ef} = \kappa_{eg} = 10L$. The bound on model Hessians κ_{bhm} can be chosen to be arbitrarily small, at the expense of limiting the class of models, however, it is clearly reasonable to choose it as something that scales with L, if this information is available. Let us assume that $\kappa_{bhm} = 10L$, as well. In a standard trust region method, a common choice of algorithmic parameters would use $\kappa_{fcd} = \frac{1}{2}$, $\gamma = 2$, and $\eta_1 = \frac{1}{2}$.

The reader can easily verify that with these parameter choices and previous assumptions, Lemma 4.11 states that we must choose $\eta_2 \geq 32L$. The intermediate constants satisfy $\zeta \geq 42L$ and $C_1 = \frac{2}{17}$. Without loss of generality, we will simply accept $\zeta = 42L$.

From this, we immediately get that the accuracy parameter ϵ_F can be chosen as L. In general, we obtain that ϵ_F scales with L, whenever $\kappa = (\kappa_{ef}, \kappa_{eg})$ scales with L, but with constants strictly smaller than those in κ, a mathematical statement of our claim that our point estimates must be more accurate than our models.

Continuing the analysis of this example, supposing $L > \frac{1}{5}$, a very reasonable assumption for a nonlinear function, we get bounds affecting the probability parameters

$$\frac{\alpha\beta}{(1-\alpha)(1-\beta)} \geq \frac{17}{7L} + \frac{493}{56} \tag{50}$$

and

$$(1-\alpha)(1-\beta) \leq \frac{1}{5 + 232L} \tag{51}$$

We note that as η_1 and κ_{fed} constants are driven closer to 1, the constant 232 can be reduced by up to a factor of 4. Asymptotically, as $L \to \infty$, (50) is constant, and (51) scales as $\frac{1}{L}$. Thus, the probability with which we require accurate models and estimates increases with the nonlinearity of the function being minimized, as one would intuitively expect. We record this more generally in the following corollary, whose proof can be extended from the discussion just given.

Corollary 4.13. *Supposing that κ_{ef}, κ_{eg}, and κ_{bhm} scale linearly with L, then η_2, ϵ_F ,and the expressions relating α and β in Lemma 4.11 are all functions in L satisfying*

$$\eta_2 \in \Theta(L), \tag{52}$$

$$\epsilon_F \in \Theta(L), \tag{53}$$

$$\frac{\alpha\beta}{(1-\alpha)(1-\beta)} \in \Theta(1), \tag{54}$$

and

$$(1-\alpha)(1-\beta) \in \Theta(\frac{1}{L}), \tag{55}$$

where the constants in Θ are moderate in size.

Remark 4.14. *Recall our remark made earlier in Theorem 4.10, case 2b, on how η_2 bounds our step sizes. Indeed, if $\eta_2 \in \Theta(L)$ has to be imposed this may force the algorithm to take small step sizes throughout. However, as mentioned earlier, the analysis of Theorem 4.10 can be modified by introducing tradeoff between the size of η_2 and the accuracy parameters ϵ_F and κ_{ef} (as both of these constant parameters can be made smaller). It also may be advantageous to choose η_2 dynamically. Exploring this is a subject for future work. In the practical implementations that we will discuss in Section 6, we do not make use of the algorithmic parameter η_2 at all, and so even though η_2 is effectively arbitrarily close to 0, the algorithm still works.*

Remark 4.15. *Note that if $\beta = 0$, then $\Delta_k \to 0$ for any value of α, which is the case shown in [2].*

4.1 The liminf-type convergence

We are ready to prove a lim inf-type first-order convergence result, i.e., that a subsequence of the iterates drive the gradient of the objective function to zero. The proof follows closely that in [2], the key difference being the assumption on the function estimates that are needed to ensure that a good step gets accepted by Algorithm 1.

Theorem 4.16. *Let the assumptions of Lemmas 4.10 and 4.11 hold. Suppose additionally that $\alpha\beta \geq \frac{1}{2}$. Then the sequence of random iterates generated by Algorithm 1, $\{X_k\}$, almost surely satisfies*

$$\liminf_{k\to 0} \|\nabla f(X_k)\| = 0.$$

Proof. We prove this result by contradiction conditioned on the almost sure event $\Delta_k \to 0$. Let us thus assume that there exists ϵ' such that, with positive probability, we have

$$\|\nabla f(X_k)\| \geq \epsilon', \quad \forall k.$$

Let $\{x_k\}$ and $\{\delta_k\}$ be realizations of $\{X_k\}$ and $\{\Delta_k\}$, respectively for which $\|\nabla f(x_k)\| \geq \epsilon', \quad \forall k$. Since $\lim_{k\to 0} \delta_k = 0$, there exists k_0 such that for all $k \geq k_0$,

$$\delta_k < b := \min\left\{ \frac{\epsilon'}{2\kappa_{eg}}, \frac{\epsilon'}{2\kappa_{bhm}}, \frac{\kappa_{fcd}(1-\eta_1)\epsilon'}{16\kappa_{ef}}, \frac{\epsilon'}{2\eta_2}, \frac{\delta_{\max}}{\gamma} \right\}. \tag{56}$$

We define a random variable R_k with realizations $r_k = \log_\gamma\left(\frac{\delta_k}{b}\right)$. Then for the realization $\{r_k\}$ of $\{R_k\}$, $r_k < 0$ for $k \geq k_0$. The main idea of the proof is to show that such realizations occur only with probability zero, hence obtaining a contradiction with the initial assumption of $\|\nabla f(x_k)\| \geq \epsilon'$ $\forall k$.

We first show that R_k is a submartingale. Recall the events I_k and J_k in Definitions 3.4 and 3.5. Consider some iterate $k \geq k_0$ for which I_k and J_k both occur, which happens with probability $P(I_k \cap J_k) \geq \alpha\beta$. Since (56) holds we have exactly the same situation as in Case 1(a) in the proof of Lemma 4.10. In other words, we can apply Lemmas 4.5 and 4.6 to conclude that the k-th iteration is successful, hence, the trust-region radius is increased. In particular, since $\delta_k \leq \frac{\delta_{\max}}{\gamma}$, $\delta_{k+1} = \gamma\delta_k$. Consequently, $r_{k+1} = r_k + 1$.

Let $\mathcal{F}_{k-1}^{I\cdot J} = \sigma(I_0, \cdots, I_{k-1}) \cap \sigma(J_0, \cdots, J_{k-1})$. For all other outcomes of I_k and J_k, which occur with total probability of at most $1 - \alpha\beta$, we have $\delta_{k+1} \geq \gamma^{-1}\delta_k$. Hence

$$\mathbb{E}[r_{k+1} | \mathcal{F}_{k-1}^{I\cdot J}] = \alpha\beta(r_k + 1) + (1 - \alpha\beta)(r_k - 1) \geq r_k,$$

as long as $\alpha\beta \geq 1/2$, which implies that R_k is a submartingale.

Now let us construct another submartingale W_k, on the same probability space as R_k which will serve as a lower bound on R_k and for which $\left\{\limsup_{k\to 0} W_k = \infty\right\}$ holds almost surely. Define indicator random variables $\mathbf{1}_{I_k}$ and $\mathbf{1}_{J_k}$ such that $\mathbf{1}_{I_k} = 1$ if I_k occurs, $\mathbf{1}_{I_k} = 0$ otherwise, and similarly, $\mathbf{1}_{J_k} = 1$ if J_k occurs, $\mathbf{1}_{J_k} = 0$ otherwise. Then define

$$W_k = \sum_{i=0}^{k} (2 \cdot \mathbf{1}_{I_k} \cdot \mathbf{1}_{J_k} - 1).$$

Notice that W_k is a submartingale since

$$\begin{aligned}
\mathbb{E}[W_k | \mathcal{F}_{k-1}^{I \cdot J}] &= E[W_{k-1} | \mathcal{F}_{k-1}^{I \cdot J}] + E[2 \cdot \mathbf{1}_{I_k} \cdot \mathbf{1}_{J_k} - 1 | \mathcal{F}_{k-1}^{I \cdot J}] \\
&= W_{k-1} + 2E[\mathbf{1}_{I_k} \cdot \mathbf{1}_{J_k} | \mathcal{F}_{k-1}^{I \cdot J}] - 1 \\
&= W_{k-1} + 2P(I_k \cap J_k | \mathcal{F}_{k-1}^{I \cdot J}) - 1 \\
&\geq W_{k-1},
\end{aligned}$$

where the last inequality holds because $\alpha\beta \geq 1/2$. Since W_k only has ± 1 increments, it has no finite limit. Therefore, by Theorem 4.3, we have $\left\{\limsup_{k\to 0} W_k = \infty\right\}$.

By the construction of R_k and W_k, we know that $r_k - r_{k_0} \geq w_k - w_{k_0}$. Therefore, R_k has to be positive infinitely often with probability one. This implies that the sequence of realizations r_k such that $r_k < 0$ for $k \geq k_0$ occurs with probability zero. Therefore our assumption that $\|\nabla f(X_k)\| \geq \epsilon'$ hold for all k with positive probability is false and

$$\liminf_{k\to 0} \|\nabla f(X_k)\| = 0$$

holds almost surely.

□

4.2 The lim-type convergence

In this subsection we show that $\lim_{k\to\infty} \|\nabla f(X_k)\| = 0$ almost surely.

We now state an auxiliary lemma, which is similar to the one in [2], but requires a different proof because in our case the function values $f(X_k)$ can increase with k, while in the case considered in [2], function values are monotonically nonincreasing.

Lemma 4.17. *Let the same assumptions made in Theorem 4.16 hold. Let $\{X_k\}$ and $\{\Delta_k\}$ be sequences of random iterates and random trust-region radii generated by Algorithm 1. Fix $\epsilon > 0$ and define the sequence $\{K_\epsilon\}$ consisting of the natural numbers k for which $\|\nabla f(X_k)\| > \epsilon$ (note that K_ϵ is a sequence of random variables). Then,*

$$\sum_{k \in \{K_\epsilon\}} \Delta_k \ < \ \infty$$

almost surely.

Proof. From Theorem 4.10 we know that $\sum \Delta_k^2 < \infty$ and hence $\Delta_k \to 0$ almost surely. For each realization of Algorithm 1 and a sequence $\{\delta_k\}$, there exists k_0 such that $\delta_k \leq \epsilon/\zeta$, $\forall k \geq k_0$, where ζ is defined as in Theorem 4.10. Let K_0 be the random variable with realization k_0 and let K denote the sequence of indices k such that $k \in K_\epsilon$ and $k \geq K_0$. Then for all $k \in K$, Case 1 of Theorem 4.10 holds, i.e., $\|\nabla f(X_k)\| \geq \zeta \Delta_k$, since $\|\nabla f(X_k)\| \geq \epsilon$ for all $k \in K$. From this and from (35) we have

$$E[\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}^{M \cdot F}] \leq -[C_1 \alpha\beta - (1-\alpha)(1-\beta)C_3]\nu\epsilon\Delta_k, \quad \forall k \geq k_0.$$

Recall that Φ_k is bounded from below. Hence, summing up the above inequality for all $k \in K$ and taking the expectation, we have that

$$\sum_{k \in K} \Delta_k \ < \ \infty$$

almost surely. Since $K_\epsilon \subseteq K \cap \{k \leq K_0\}$ and K_0 is finite almost surely then the statement of the lemma holds.

We are now ready to state the lim-type result.

Theorem 4.18. *Let the same assumptions of Theorem 4.16 hold. Let $\{X_k\}$ be a sequence of random iterates generated by Algorithm 1. Then, almost surely,*

$$\lim_{k \to \infty} \|\nabla f(X_k)\| \ = \ 0.$$

Proof. The proof of this result, is almost identical to the proof of the same theorem in [2] hence we will not present the proof here. The key idea of the proof is to show that if the theorem does not hold, then with positive probability

$$\sum_{k \in \{K_\epsilon\}} \Delta_k \ = \ \infty,$$

with K_ϵ defined as in Lemma 4.17. This result is shown using Lipschitz continuity of the gradient and does not depend on the stochastic nature of the algorithm. Since this result contradicts the almost sure result of Lemma 4.17, we can conclude that the statement of the theorem holds almost surely.

5 Constructing models and estimates in different stochastic settings.

We now discuss various settings of stochastic noise in the objective function and how α-probabilistically κ-fully linear models and β-probabilistically ϵ_F-accurate estimates can be obtained is these settings.

Recall that we assume that for each x we can compute the value of $\tilde{f}(x)$, which is the noisy version of f,

$$\tilde{f}(x) = f(x, \omega)$$

where ω is a random variable which induces the noise.

Simple stochastic noise. The example of noise which is typically considered in stochastic optimization is the "i.i.d." noise, that is noise with distribution independent of the function values. Here we consider a somewhat more general setting, where the noise is unbiased for all f, i.e.,

$$E_\omega[f(x,\omega)] = f(x), \quad \forall x,$$

and

$$\mathrm{Var}_\omega[f(x,\omega)] \leq V < \infty, \quad \forall x.$$

This is the typical noise assumption in stochastic optimization literature. In the case of unbiased noise as above, constructing estimates and models that satisfy our assumptions is fairly straightforward. First, let us consider the case when only the noisy function values are available (without any gradient information), where we want to construct a model that is κ-fully linear in a given trust region $B(x^0,\delta)$ with some reasonably large probability, α.

One can employ standard sample averaging approximation techniques to reduce the variance of the function evaluations. In particular, let $\bar{f}_p(x,\omega) = \frac{1}{p}\sum_{i=1}^p f(x,\omega_i)$, where ω_i are the i.i.d. realizations of the noise ω. Then, by Chebyshev inequality, for any $v > 0$,

$$P(|\bar{f}_p(x,\omega) - f(x)]| > v) = P(|\bar{f}_p(x,\omega) - E_\omega[f(x,\omega)]| > v) \leq \frac{V}{pv^2}.$$

In particular, we want $v \leq \kappa'_{ef}\delta^2$ for some $\kappa'_{ef} > 0$ and $\frac{V}{pv^2} \leq 1-\alpha'$ for some α', which can be ensured by choosing $p \geq \frac{V}{\kappa'_{ef}(1-\alpha')\delta^4}$.

We now construct a fully linear model as follows: given a well-poised set[1] Y of $n+1$ points in $B(x^0,\delta)$, at each point $y^i \in Y$, we compute $\bar{f}_p(y^i,\omega)$ and build a linear interpolation model $m(x)$ such that $m(y^i) = \bar{f}_p(y^i,\omega)$, for all $i = 1,\ldots,n+1$. Hence, for any $y^i \in Y$, we have

$$P(|m(y^i) - f(y^i)]| > \kappa'_{ef}\delta^2) \leq 1-\alpha'.$$

Moreover, the events $\{|m(y^i) - f(y^i)]| > \kappa'_{ef}\delta^2\}$ are independent, hence

$$P(\max_{i=1..n+1}\{|m(y^i) - f(y^i)]|\} > \kappa'_{ef}\delta^2) \leq 1-(\alpha')^{n+1}.$$

It is easy to show using, for example, techniques described in [7], that $m(x)$ is a κ-fully linear model of $E_\omega[f(x,\omega)]$ in $B(x^0,\delta)$ for appropriately chosen $\kappa = (\kappa_{eg},\kappa_{ef})$, with probability at least $\alpha = (\alpha')^{n+1}$.

The majority of stochastic optimization and sample average approximation methods focus on derivative based optimization where it is assumed that, in addition to $f(x,\omega)$, $\nabla_x f(x,\omega)$ is also available, and that the noise in the gradient computation is also independent of x, that is

$$E_\omega[\nabla_x f(x,\omega)] = \nabla f(x), \quad \forall x$$

and

$$\|\mathrm{Var}_\omega[\nabla_x f(x,\omega)]\| \leq V < \infty, \quad \forall x,$$

(in general the variance of the gradient and the function value are not the same, but here for simplicity we bound both by V).

[1] See [7] for details on well-poised sets and how they can be obtained.

In the case when the noisy gradient values are available, the construction of fully linear models in $B(x^0, \delta)$ is simpler. Let $\bar{\nabla} f_p(x, \omega) = \frac{1}{p} \sum_{i=1}^{p} \nabla f(x, \omega_i)$. Again, by extension of Chebychev inequality, for p such that

$$p \geq \max\{\frac{V}{\kappa_{ef}(1-\alpha')\delta^4}, \frac{V}{\kappa_{eg}(1-\alpha')\delta^2}\},$$

$$P(\|\bar{\nabla} f_p(x^0, \omega) - \nabla f(x^0)]\| > \kappa_{eg}\delta) = P(\|\bar{\nabla} f_p(x^0, \omega) - E_\omega[\nabla f(x^0, \omega)]\| > \kappa_{eg}\delta) \leq 1 - \alpha',$$

and

$$P(|\bar{f}_p(x^0, \omega) - f(x^0)| > \kappa_{ef}\delta^2) = P(|\bar{f}_p(x^0, \omega) - E_\omega[f(x^0, \omega)]| > \kappa_{ef}\delta^2) \leq 1 - \alpha'.$$

Hence the linear expansion $m(x) = \bar{f}_q(x^0, \omega) + \bar{\nabla} f_p(x^0, \omega)^T (x - x^0)$ is a κ-fully linear model of $f(x) = E_\omega[f(x, \omega)]$ on $B(x^0, \delta)$ for appropriately chosen $\kappa = (\kappa_{eg}, \kappa_{ef})$, with probability at least $\alpha = (\alpha')^2$.

In [15] it is shown that least squares regression models based on sufficiently large strongly poised [7] sample sets are α-probabilstically κ-fully linear models.

Computing the β-probabilistically ϵ_F-accurate estimates of $f(x, \omega)$ can be done analogously to the construction of the models in all of the above cases.

There are many existing methods and convergence results using sample average approximations and stochastic gradients for stochastic optimization with i.i.d. or unbiased noise. Some of these methods have been shown to achieve optimal sampling rate [12], that is they converge to the optimal solution while sampling the gradient at the best possible rate. We do not provide convergence rates in this paper (it is a subject for future research), hence it remains to be seen if our algorithm can achieve the optimal rate. Our contribution here is the method which applies beyond the i.i.d. case, as we will discuss below. In Section 6, however, we demonstrate that our method can have superior numerical behavior compared to standard sample averaging even in the case of the i.i.d. noise, so it is at least competitive in practice.

Function computation failures. Now, let us consider a more complex noise case. Suppose that the function $f(x)$ is computed (as it often is, in black-box optimization) by some numerical method that includes a random component. Such examples are common in machine learning applications, for instance, where x is the vector of hyperparameters of a learning algorithm and $f(x)$ is the expected error of the resulting classifier. In this case, for each value of x, the classifier is obtained by solving an optimization problem, up to a certain accuracy, on a given training set. Then the classifier is evaluated on the testing set. If a randomized coordinate descent or a stochastic gradient descent is applied to train the classifier for a given vector x, then the resulting classifier is sufficiently close to the optimal classifier with some known probability. However, in the case when the training of the classifier fails to produce a sufficiently accurate solution, the resulting error is difficult to estimate. Usually, it is possible to know the upper bound on the value of this inaccurate objective function but nothing else may be known about the distribution of this value. Moreover, it is likely that the probability of the accurate computation depends on x; for example, after k iterations of randomized coordinate descent, the error between the true $f(x)$ and the computed $\tilde{f}(x)$ is bounded by some value, with probability α, where the value depends on α, k and x [20].

Another example is solving a system of nonlinear black-box equations. Assume that we seek x such that $\sum_i (f_i(x))^2 = 0$, for some functions $f_i(x)$, $i = 1, \ldots, m$ that are computed by numerical simulation, with noise. As is often done in practice (and is supported by our theory) the noise in

the function computation is reduced as the algorithm progresses, for example, by reducing the size of a discretization, step size, or convergence tolerance within the black-box computation. These adjustments for noise reduction usually increase the workload of the simulation. With the increase of the workload, there is an increased probability of failure of the code. Hence, the smaller the values of $f_i(x)$, the more likely the computation of $f_i(x)$ will fail and some inaccurate value is returned.

These two examples show that the noise in $\tilde{f}(x)$ may be large with some positive probability, which may depend on x. Hence, let us consider the following, idealized, noise model

$$\tilde{f}(x) = f(x,\omega) = \begin{cases} f(x), & \text{w.p. } 1-\sigma(x) \\ \omega(x) \leq V & \text{w.p. } \sigma(x), \end{cases}$$

where $\sigma(x)$ is the probability with which the function $f(x)$ is computed inaccurately, and $\omega(x)$ is some random function of x, for which only an upper bound V is known. This case is idealized, because we assume that with probability $1-\sigma(x)$, $f(x)$ is computed exactly. It is trivial to extend this example to the case when $f(x)$ is computed with an error, but this error can be made sufficiently small.

For this model of function computation failures we have

$$E_\omega[f(x,\omega)] = (1-\sigma(x))f(x) + \sigma(x)E[\omega(x)] \neq f(x), \quad \forall \sigma(x) > 0.$$

and it is clear, that for any $\sigma(x) > 0$, unless $E[\omega(x)] \equiv$ some constant, optimizing $E_\omega[f(x,\omega)]$ does not give the same result as optimizing $f(x)$. Hence applying Monte-Carlo sampling within an optimization algorithm solving this problem is not a correct approach.

We now observe that constructing α-probabilistically κ-fully linear models and β-probabilistically ϵ_F-accurate estimates is trivial in this case, assuming that $\sigma(x) \leq \sigma$ for all x, when σ is small enough. In particular, given a trust region $B(x^0,\delta)$, sampling a function $f(x)$ on a sample set $Y \subset B(x^0,\delta)$ well-poised for linear interpolation will produce a κ-fully linear model in $B(x^0,\delta)$ with probability at least $(1-\sigma)^{|Y|}$, since with this probability all of the function values are computed exactly. Similarly, for any $s \in B(x^0,\delta)$, the function estimates F^0 and F^s are both correct with probability at least $(1-\sigma)^2$. Assuming that $(1-\sigma)^{|Y|} \geq \alpha$ and $(1-\sigma)^2 \geq \beta$ where α and β satisfy the assumptions of Lemmas 4.10 and 4.11 and $\alpha\beta \geq \frac{1}{2}$ as in Theorem 4.16, we observe that the resulting models satisfy our theory.

Remark 5.1. *We assume here that the probability of failure to compute $f(x)$ is small enough for all x. In the machine learning example above, it is often possible to control the probability $\sigma(x)$ in the computation of $f(x)$, for example by increasing the number of iterations of a randomized coordinate descent or stochastic gradient method. In the case of the black-box nonlinear equation solver, the probability of code failure is expected to be quite small. There are, however, examples of black box optimization problems where the computation of $f(x)$ fails all the time for specific values of x. This is often referred to as hidden constraints [17]. Clearly our theory does not apply here, but we believe there is no local method that can provably converge to a local minimizer in such a setting without additional information about these specific values of x.*

Processor failures. Consider now another example of derivative based approaches. Let $f(x)$ be a black box function, which can be computed sufficiently accurately, by some numerical process. Assume that we employ a finite difference scheme to compute approximate derivatives of $f(x)$,

and that we use parallel computations to do this. For simplicity let us assume that the number of processors is exactly the same as the number of function evaluations needed to compute each model (i.e., $n+1$). Let us now assume that one of the processors tends to be slower than others and with probability σ it does not compute an accurate function value in the expected time. In this case some fixed value V is returned instead. Let x^0 be the current iterate and let μ be a finite differencing parameter. Hence we compute function values at points $x^0 + \mu e_i$, where e_i is the i-th element of the elementary basis. Let $i^* \leq n$ be the index of the faulty processor. Then we have

$$\tilde{f}(x^0 + \mu e_i) = \begin{cases} f(x^0 + \mu e_i), & \forall i \neq i^* \\ f(x^0 + \mu e_{i^*}) & \text{w.p. } 1-\sigma \text{ for } i = i^* \\ V & \text{w.p. } \sigma \text{ for } i = i^*, \end{cases}$$

and

$$\tilde{f}(x^0) = f(x^0).$$

Let $\nabla_{x,\mu}\tilde{f}(x^0)$ be the finite difference gradient of $\tilde{f}$ obtained from the function values which are computed as above and let $\nabla_{x,\mu}f(x^0)$ be the finite difference gradient of the true function $f(x)$ with the same differing step μ. It is clear that $E[\nabla_{x,\mu}\tilde{f}(x^0)] \neq \nabla_{x,\mu}f(x^0)$ in this case, hence neither a sample averaging method nor a stochastic gradient method will be applicable in this case. On the other hand, $\nabla_{x,\mu}\tilde{f}(x^0) = \nabla_{x,\mu}f(x^0)$ with probability $1-\sigma$ and hence $m(x) = f(x^0) + \nabla_{x,\mu}\tilde{f}(x^0)^T(x - x^0)$ is a κ-fully linear model on $B(x^0, \delta)$ for any $\delta \geq \mu$, with probability at least $1-\sigma$.

In the next section we test the performance of our random model trust-region algorithm against a stochastic averaging trust region method and a simple stochastic gradient descent method. What we observe is that in the case of unbiased noise, the random model method is simple, efficient and competitive and in the biased noise cases, it is the only viable method.

6 Computational Experiments

In this section, we will discuss the performance of an implementation of our proposed method, henceforth only referred to as STORM (STochastic Optimization using Random Models), across the various noisy situations discussed in the previous section. We wish to note that there are further numerical results comparing an earlier implementation of STORM in [5] to the SPSA method of [23] and the classical Kiefer-Wolfowitz method in [14]. STORM significantly outperformed these two methods in that setting, but no special tuning of SPSA or Kiefer-Wolfowitz was applied. However, it is reasonable to expect that a trust-region based method, which is also able to use second order information will outperform stochastic gradient-like methods in many settings. Here we choose to compare our method with another trust-region based method for the most fair comparison. Details on that are below.

6.1 Gradient-free method comparison

Simple stochastic noise. In these experiments, we used a set of 53 unconstrained problems adapted from the CUTEr test set, each being in the form of a sum of squares problem, i.e.

$$f(x) = \sum_{i=1}^{m} (f_i(x))^2, \tag{57}$$

where for each $i \in \{1, \ldots, m\}$, $f_i(x)$ is a smooth function. Two different types of noise will be used in this first section, which we will refer to as *relative* noise and *additive* noise. In the relative noise case, for each $i \in \{1, \ldots, m\}$, we generate some ϵ_i from the uniform distribution on $[-\sigma, \sigma]$ for some parameter $\sigma > 0$, and then compute the noisy

$$\tilde{f}(x, \omega) = \sum_{i=1}^{m} (1 + \omega_i) f_i(x). \tag{58}$$

The key characteristic of this noise is that for each x, we have $E_\omega[f(x, \omega)] = f(x)$, however the variance is nonconstant over x and scales *relatively* with the magnitudes of the components $f_i(x)$. Thus, while this is not exactly an i.i.d. noise, expectations are unbiased for each x and so sample means can be used to control noise. Moreover, one should expect that if an algorithm is minimizing a function of the form (58), the quality of the estimates of $f(x)$ based on a constant number of samples of $\tilde{f}(x, \omega)$ should increase over time, assuming that the algorithm produces a decreasing sequence $\{f(x^k)\}_{k=1}^{\infty}$. While this behavior is not supported by theory, because we do not know how quickly $f(x^k)$ decreases, our computational results show that a constant number of samples is indeed sufficient for convergence.

The other type of noise we will test is *additive*, i.e. we additively perturb each component in (57) by some ω_i uniformly generated in $[-\sigma, \sigma]$ for some parameter $\sigma > 0$. That is,

$$\tilde{f}(x, \omega) = \sum_{i=1}^{m} (f_i(x) + \omega_i)^2 \tag{59}$$

Note that the noise is additive only in terms of the component functions, but not in terms of the objective function, moreover $E_\omega[f(x, \omega)] = f(x) + \sum_i^m E(\omega_i)^2$. However, a constant bias term should not affect optimization methods, since $\min_x E_\omega[f(x, \omega)] = \min_x f(x)$.

In our first set of experiments, we will test several versions of STORM against several versions of an implementation of an algorithm presented in [8]. We note that our implementation of what we call 'Deng and Ferris' is not exactly the one proposed in their paper, which is a sample path optimization algorithm using a trust region framework. In their work, they use a Bayesian scheme to estimate a sufficiently large sample complexity for computing average function values at a current interpolation set that ensures sufficient decrease relative to a quadratic model of an expectation function. In our implementation of the Deng and Ferris algorithm, rather than infer distributions of samples to estimate necessary sample complexity, we simply increase sample complexity proportional to the decrease of the trust region radius δ. Based on our discussion of relative and additive noise above, such an algorithm is appropriate in those cases, as average function values at a given point x serve as an unbiased estimator of the expected function value at x. Furthermore, the trust-region framework used in 'Deng and Ferris' to control step sizes makes it easily (and fairly) comparable to STORM.

Our theory dictates that in STORM, like in 'Deng and Ferris', we should also increase sample size as δ decreases in the same way that we do for Deng and Ferris, however, we would like to demonstrate in these experiments that we can use a constant sample rate (i.e. we take only one

noisy function evaluation of $\tilde{f}(y,\omega)$ at each point y in a given interpolation set and we keep the size of the interpolation set bounded) and find "acceptable", if necessarily accurate solutions, quickly relative to Deng and Ferris. As one would expect, on any problem with i.i.d. noise, STORM with a constant sampling rate manages to find decrease until it reaches a level set where the changes in the function value or gradient approximation in the current $B(x_k,\delta_k)$ become so small that the noise becomes dominating. In future work, we would like to investigate this further to determine optimal sampling rates in terms of assumptions on probabilistically fully linear models and probabilistically accurate estimates.

We will first present results in the relative noise case in the performance profiles below. Eight different solvers were tested, 4 variations each of STORM and of the Deng and Ferris methods. All eight of these solvers follow the general framework given below, and we will comment on the specific choices of how to initialize and perform steps 2 and 7.

Algorithm 2 Practical Implementation of STORM

1: (Initialization): Choose an initial point x_0 and an initial trust-region radius $\delta_0 \in (0,\delta_{\max})$ with $\delta_{\max} > 0$. Choose constants $\gamma > 1$, $\eta_1 \in (0,1)$, Y_{max}; Set $k \leftarrow 0$. Choose some initial interpolation set $Y_0 \subset B(x_k,\delta_k)$ so that $|Y_0| \leq Y_{max}$ and compute some noisy evaluation $\tilde{f}(y)$ at each $y \in Y_0$.
2: Build a model $m_k(x_k+s) = f_k + g_k^\top s + s^\top H_k s$ with $s = x - x_k$ that interpolates $\tilde{f}(y)$ at the points of Y_k.
3: (Step calculation) Compute $s_k = \arg\min_{s:\|s\|\leq\delta_k} m_k(s)$ (approximately) so that it satisfies condition (3).
4: (Estimates calculation) Compute new estimates f_k^0 and f_k^s of $f(x_k)$ and $f(x_k+s_k)$, respectively. In this implementation, we simply reevaluate $\tilde{f}(x_k)$ and $\tilde{f}(x_k+s_k)$ once each.
5: (Acceptance of the trial point): Compute $\rho_k = \dfrac{f_k^0 - f_k^s}{m_k(x_k) - m_k(x_k+s_k)}$. If $\rho_k \geq \eta_1$, then $x_{k+1} \leftarrow x_k + s_k$; otherwise, $x_{k+1} \leftarrow x_k$.
6: (Trust-region radius update): If $\rho_k \geq \eta_1$, $\delta_{k+1} \leftarrow \min\{\gamma\delta_k, \delta_{\max}\}$; otherwise $\delta_{k+1} \leftarrow \gamma^{-1}\delta_k$.
7: (Interpolation set update): Acquire a new interpolation set Y_k and acquire a noisy evaluation $\tilde{f}(y)$ at each $y \in Y_k$.
8: (Iterate): $k \leftarrow k+1$ and go to step 2.

STORM-A: In initialization, $Y_{max} \leftarrow n+1$. In Step 2, $H_k = 0$, and so we are effectively building linear interpolation models. In Step 7 of each iteration, Y_k is a set of $n+1$ random points generated uniformly in a hypersphere of radius δ_k. At each of the $n+1$ points, the noisy function $\tilde{f}(y,\omega)$ is evaluated exactly once.

STORM-B: In initialization, $Y_{max} \leftarrow (n+1)(n+2)/2$. In Step 2 of each iteration, we are building quadratic interpolation models. In Step 7, we augment Y_k with $n+1$ many new random points and select the remaining $(n+1)(n+2)/2-(n+1)$ as previously evaluated points closest to the current iterate. The $n+1$ 'new' points are evaluated as $\tilde{f}(y,\omega)$ exactly once, while the 'old' points retain their previous function evaluations.

STORM-C: In initialization, $Y_{max} \leftarrow (n+1)(n+2)/2$. In Step 2 of each iteration, we are building quadratic interpolation models. In Step 7, we augment Y_k with only $x_k + s_k$, and we compute a new noisy function evaluation there, $\tilde{f}(x_k + s_k, \omega)$ as the function value to be interpolated. We delete the point of Y_k furthest from the new trust region center x_{k+1}.

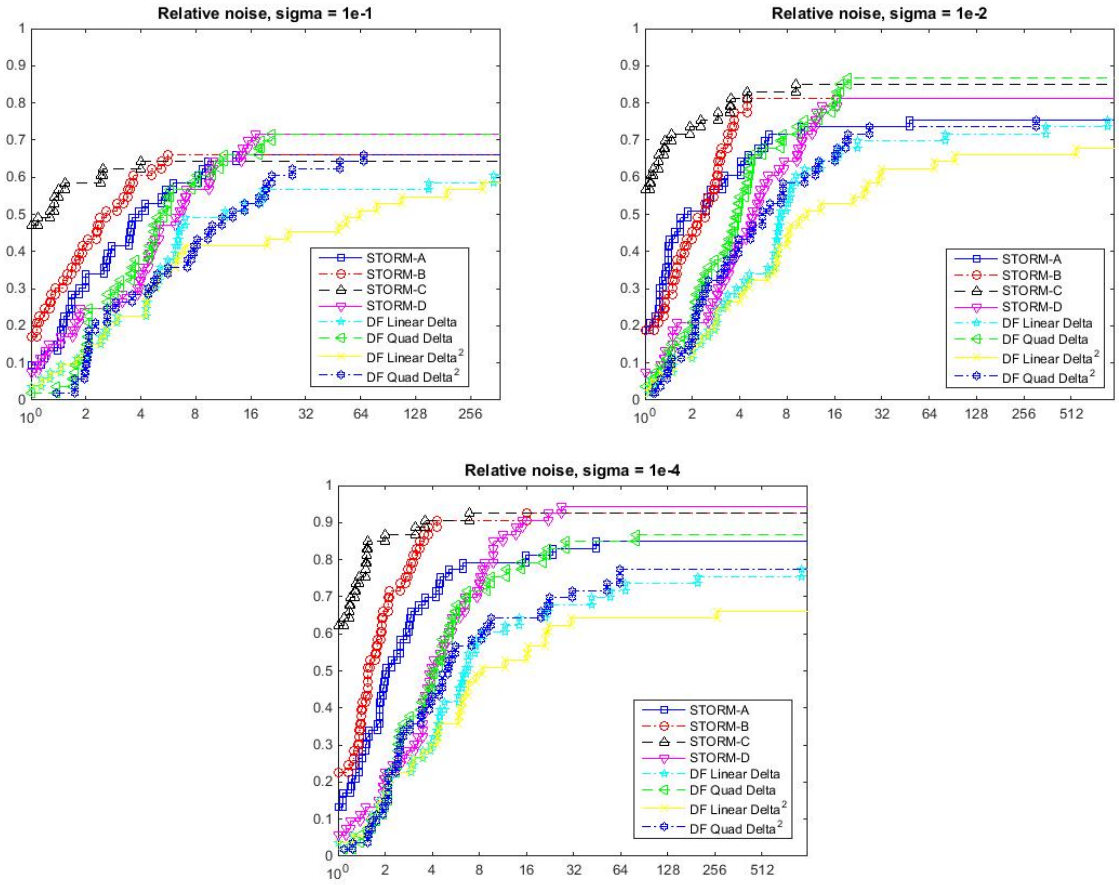
STORM-D: In initialization, $Y_{max} \leftarrow (n+1)(n+2)/2$. In Step 2 of each iteration, we are building quadratic interpolation models. In Step 7, Y_k is a set of $(n+1)(n+2)/2$ random points generated uniformly in a hypersphere of radius δ_k and we compute a new noisy function evaluation $\tilde{f}(y, \omega)$ at each $y \in Y_k$.

As for the Deng and Ferris implementations, in the implementations marked 'DF Linear', we build linear interpolation models in Step 2 (so $H_k = 0$) and initialize $Y_{max} = n + 1$; in the implementations marked 'DF Quad', quadratic interpolation models are built and we initialize $Y_{max} = (n+1)(n+2)/2$. In the implementations marked 'Delta', on the kth iteration a sampling rate $p_k = \max\{n + 1, 1/\delta_k\}$ is selected so that we can define an average function value $\tilde{f}_{p_k}(y) = \frac{1}{p_k} \sum_{i=1}^{n} \tilde{f}(y, \omega)$ for any given y, which will serve as a function value interpolated by a model. Similarly, in the implementations marked 'Delta2', a sampling rate of $p_k = \max\{n+1, 1/\delta_k^2\}$ is used. In all of the implementations of Deng and Ferris, in Step 7, we use the same incremental strategy as in STORM-C. However, in Step 7, Deng and Ferris will improve the quality of the sample mean $\tilde{f}_{p_k}(y)$ at each $y \in Y_k$ by weighting $\tilde{f}_{p_k}(y)$ with $p_{k+1} - p_k$ new samples of $\tilde{f}(y, \omega)$ to obtain $\tilde{f}_{p_{k+1}}(y)$ (provided y is still in Y_{k+1}). Furthermore, in any Deng and Ferris implementation, we do not perform Step 4, as the idea of independent model and estimation accuracies as this is STORM-specific; instead, Deng and Ferris uses the model value $m_k(x_k) = \tilde{f}_{p_k}(x_k)$ and a newly obtained $\tilde{f}_{p_k}(x_k + s_k)$ as f_k^0 and f_k^s, respectively.

For each level of noise σ, each of the 53 problems was solved 10 times with the same initial point x^0, but with different random seeds. For each problem, the best value of the noiseless $f(x)$ obtained by a solver is recorded as f^*. For each run of a solver on a problem, we are interested in the number of function evaluations required by a solver to obtain a function value $f(x^k) < f'$ such that

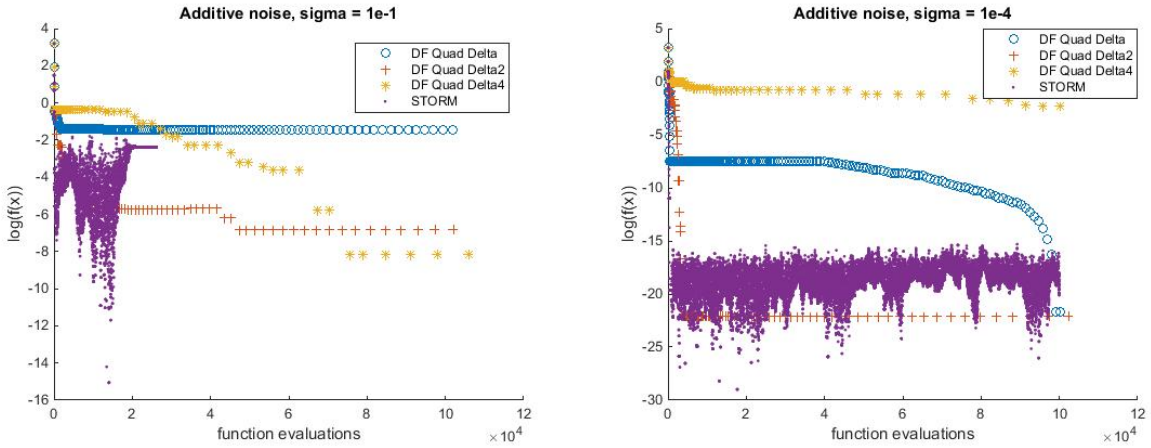
$$1 - \tau < \frac{f(x^0) - f'}{f(x^0) - f*}.$$

In other words, a solver is said to have passed a convergence test for a given problem once its reduction of the objective value is within $100\tau\%$ of the best reduction of the objective value of that problem obtained by any solver. In the profiles shown below for the relative noise case, assume $\tau = .01$, so that the reduction is within 1% of the best reduction. The performance profiles are in terms of the average number of function evaluations over the 10 runs necessary to pass a convergence test. In all the experiments, a budget of 100000 function evaluations was given to each solver for each problem. For the remainder of this numerical results section, all performance profiles should be interpreted in this sense of average function evaluation efficiency.



It is clear from these performance profiles that the first three variants of STORM solve the most problems to the 1% tolerance the fastest over all three levels of noise, with perhaps some preference for the incremental variant, STORM-C. To some extent, the general superiority of STORM in the relative noise case could be explained by the "self-controlled" variance property of relative noise, which we previously mentioned, making the need for iteratively increased sampling in the Deng and Ferris methods perhaps unnecessary as $\delta_k \to 0$. This confounding property is why we choose to present some simpler results for additive noise to exhibit typical long term behavior of the two types of algorithms in an additive noise setting. We should expect a STORM variant, as presented with a constant sampling rate, to eventually find a neighborhood where the underlying function is sufficiently flat so that the noise in the function evaluations will dominate the relative difference between function values in the neighborhood, and the STORM method will be unable to make progress. On the other hand, a Deng and Ferris method will be able to obtain more accurate estimates at the cost of increasingly many function evaluations to make progress, rather than stall.

This is exhibited by comparing sample trajectories of true function values at current iterates of STORM-C, and versions of quadratic Deng and Ferris methods with the shown sampling rates, on the classical 2-dimensional Rosenbrock function perturbed by additive noise:

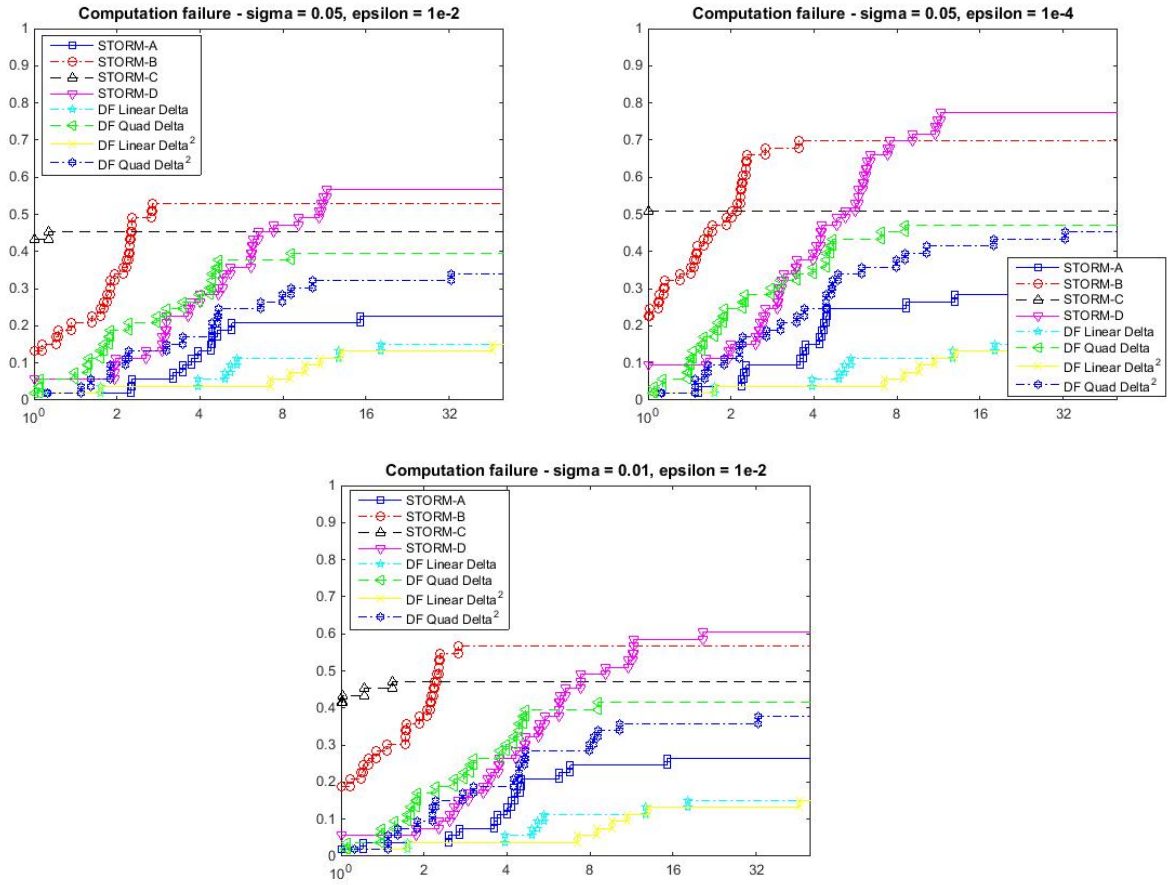


As suggested and expected, although STORM may occasionally find very accurate solutions, it need not accept nor retain them due to noise in the computation of the acceptance ratio ρ_k, hence it will stall at a not-so-accurate, although reasonable, solution. Note that STORM terminates earlier than the Deng and Ferris methods - although parameters were fairly well-tuned, eventually noise dominates function value difference so strongly that the probability of a successful step becomes extremely low. On the other hand, the increasing accuracy of the Deng and Ferris methods mandate more function evaluations, but the trajectory is more stable and tend towards much more accurate solutions. Clearly increasing sampling sizes in STORM will have the same stabilizing effect, hence it will behave at least as robustly as Deng and Ferris.

Function computation failures. In these experiments, we used the same 53 sum of squares problems as in the unbiased noise experiments described above, but introduced biased noise. For each component in the sum in (57), if $|f_i(x)| < \epsilon$ for some parameter $\epsilon > 0$, then $f_i(x)$ is computed as

$$f_i(x) = \begin{cases} f_i(x) & \text{w.p. } 1-\sigma \\ V & \text{w.p. } \sigma \end{cases}$$

for some parameter $\sigma > 0$ and for some "garbage value" V. If $f_i(x) \geq \epsilon$, then it is deterministically computed as $f_i(x)$. This noise is biased, with bias depending on x, and we should not expect any sort of averaging approximation to work well here. This is indeed indicated in our experiments, where various levels of σ and ϵ are shown below. There does not seem to be much dependence on what the garbage value is, but in the experiments illustrated below, a garbage value of $V = 10000$ was used. In these experiments, we wish to demonstrate how the STORM variants are superior in terms of accuracy, and so in the performance profiles, we set $\tau = 10^{-5}$.



In these experiments, there is a clear indication that STORM methods collectively solve a much greater percentage of the prolems than the Deng and Ferris methods to the greatest level of accuracy, especially when the noise is most prevalent in the experiments where $\sigma = 0.05$ and $\epsilon = 0.01$.

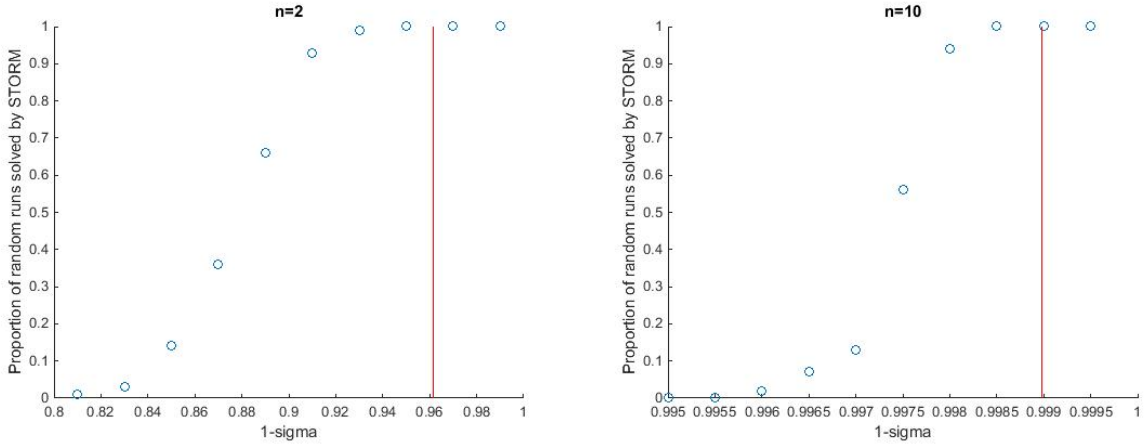
In this noise setting, the probability of the occurrence of a garbage value increases as iterates get closer to the solution. Depending on what the maximum eventual probability of obtaining a garbage value is, our theory will or will not apply for STORM. It turns out that the practical threshold of this probability at which STORM fails to progress is looser than that suggested by our theory. We will illustrate this through a simple example. Consider the minimization of the simple quadratic function

$$f(x) = \sum_{i=1}^{n} (x_i - 1)^2. \tag{60}$$

The minimizer uniquely occurs at the vector of all 1s. Now consider the minimization of this function under our setting of computation failure where we vary the probability parameter σ and fix $\epsilon = 0.1$. Suppose on each iteration of STORM, we perform the model-building step as we did in STORM-D, i.e. we sample a new $(n+1)(n+2)/2$ points on each iteration and interpolate a model through the obtained values. Then, the probability of obtaining the correct quadratic model in the worst case where for all i, $|x_i - 1| < \epsilon$ is precisely $\alpha = ((1-\sigma)^n)^{\frac{(n+1)(n+2)}{2}}$. Likewise, the probability

of obtaining the correct function evaluation for F_0 and F_s on each iteration in the worst case is $\beta = ((1-\sigma)^n)^2$. Now, supposing we initialize STORM with the zero vector in $\mathbb{R}^n$, it is reasonable to assume that all iterates will occur near the unit cube $[0,1]^n$, and so we can use simple calculus to estimate a Lipschitz constant of the function over the relevant domain as $2\sqrt{n}$, and the Lipschitz constant of the gradient is constantly 2. Thus, using all of the parameter choices specified in the remark following Lemma 4.11 with the exception that we don't use η_2, we can use $L = 2\sqrt{n}$ in (50) and (51) and impose the additional constraint that $\alpha\beta = (1-\sigma)^{\frac{n(n+1)(n+2)+4n}{2}} \geq \frac{1}{2}$ and solve for the smallest allowable $(1-\sigma)$ for which our algorithm can guarantee convergence. As two particular examples, for $n = 2$, our theory suggests that we should pick $(1-\sigma) > 0.961317$ (which implies $\alpha \approx 0.622873$ and $\beta \approx 0.854017$), and for $n = 10$, we should pick $(1-\sigma) > 0.998981$ (which implies $\alpha \approx 0.510298$ and $\beta \approx 0.979820$), since then α and β will satisfy (50), (51), and $\alpha\beta \geq \frac{1}{2}$.

Below, for $n = 2, 10$, we plot an indicated level of $(1-\sigma)$ on the x-axis against the proportion of 100 randomly seeded instances with that level of $(1-\sigma)$ that STORM-D managed to solve to the 10^{-5} level of accuracy within 10^5 function evaluations using the discussed parameter choices. The red line shows the level of $(1-\sigma)$ that our theory predicted in the previous paragraph. As we can see, $(1-\sigma)$ can be quite smaller than predicted by our theory before the failure rate becomes unsatisfactory. As a particular example, in the $n = 10$ case, when $(1-\sigma) = .998$, the corresponding probabilities are $\alpha \approx 0.266782$ and $\beta \approx 0.960751$, and yet 94% of the instances are solved to a somewhat high level of accuracy. In other words, even though the models are eventually only accurate on roughly 27% of the iterations, we still see satisfactory performance.



6.2 Gradient based method comparison

Processor failures. For these experiments, we designed a scenario very much like the one described in the previous section. In all implementations, on each iteration, a finite central differencing scheme was used to obtain a noisy gradient on a small neighborhood. Each of the $2n$ points used in computing the finite difference gradient aproximation were computed by a separate processor in parallel. To simulate the random failure of a single processor, the nth processor in each run computed its assigned function value as

$$\tilde{f}(x+\mu e_n) = \begin{cases} f(x+\mu e_n) & \text{w.p. } 1-\sigma \\ V & \text{w.p. } \sigma \end{cases} \tag{61}$$

where V is some garbage value wholly unrelated to the computation of $f(x+\mu e_n)$ (in these experiments, $V=0$) and σ is some probability parameter.

We tested three different, but comparable, approaches to using finite difference gradient approximations in an iterative scheme. The first, which we call 'STORM', is an implementation of our random model philosophy. It wholly resamples the finite difference gradient approximation on every iteration, so that the quality of an approximation on a given iteration is unaffected by the quality of an approximation on a previous iteration.

Algorithm 3 STORM, Processor Failure Experiments

1: (Initialization): Choose an initial point x_0 and an initial trust-region radius $\delta_0 \in (0, \delta_{\max})$ with $\delta_{\max} > 0$. Choose constants $\gamma > 1$, $\eta_1 \in (0,1)$. Choose finite differencing parameter $\mu > 0$. Set $k \leftarrow 0$, $x \leftarrow x_0$.
2: Compute $\tilde{f}(x \pm \mu e_i)$, one at each of the $2n$ processors.
3: Compute approximate gradient $\tilde{g}_k(x) = \frac{1}{2\mu}[\tilde{f}(x+\mu e_1) - \tilde{f}(x-\mu e_1), \ldots, \tilde{f}(x+\mu e_n) - \tilde{f}(x-\mu e_n)]^\top$.
4: Build a model $m_k(x_k + s_k) = f_k + \tilde{g}_k^\top s_k$, where $s_k = x - x_k$.
5: Solve TR subproblem $\min_{\|s\| \le \delta_k} m_k(s)$ (so $s_k = -\delta_k \tilde{g}_k / \|\tilde{g}_k\|$).
6: Evaluate $\tilde{f}(x_k)$ and $\tilde{f}(x_k + s_k)$ once each and set $f_k^0 \leftarrow \tilde{f}(x_k)$ and $f_k^s \leftarrow \tilde{f}(x_k + s_k)$.
7: Compute $\rho_k = \dfrac{f_k^0 - f_k^s}{m_k(x_k) - m_k(x_k + s_k)}$.
If $\rho_k \ge \eta_1$, then $x_{k+1} \leftarrow x_k + s_k$; otherwise, $x_{k+1} \leftarrow x_k$.
8: : If $\rho_k \ge \eta_1$, then $\delta_{k+1} \leftarrow \min\{\gamma \delta_k, \delta_{\max}\}$; otherwise $\delta_{k+1} \leftarrow \max\{\mu, \gamma^{-1}\delta_k\}$.
9: : $k \leftarrow k+1$ and return to Step 2.

The second implementation, which we shall call 'Deng and Ferris' is the same as STORM except in Steps 2 and 3. Rather than compute the value at the $2n$ processors once and use those values to get the approximate gradient $\tilde{g}_k(x)$, we will use a sample mean at the ith processor, $\tilde{f}_{p_k}(x) = \frac{1}{p_k}\sum_{j=1}^{p_k} \tilde{f}(x \pm \mu e_i)$ obtained from each of the $2n$ processors. Here, $p_k \leftarrow \max\{1, 1/\delta_k\}$. Between failed iterations, where the $2n$ compass points do not move, we update $\tilde{g}(x)$ by taking $p_k - p_{k-1}$ new samples at each of the $2n$ points and computing the new approximations at compass points as $\tilde{f}_{p_k}(x \pm \delta_k e_i) = \frac{p_{k-1}}{p_k}\tilde{f}_{p_{k-1}} + \frac{1}{p_k}\sum_{j=1}^{p_k - p_{k-1}} \tilde{f}(x \pm \mu e_i)$. We should note, though, that while this is inocuous in this experiment at the processors that "work correctly", a sample mean is a biased estimator of the true value of $f(x+\mu e_n)$ being computed at the nth processor. In particular, $E[\tilde{f}(x+\mu e_n)] = (1-\sigma)f(x+\mu e_n) + \sigma V$. In Step 3, the approximate gradient is then computed as

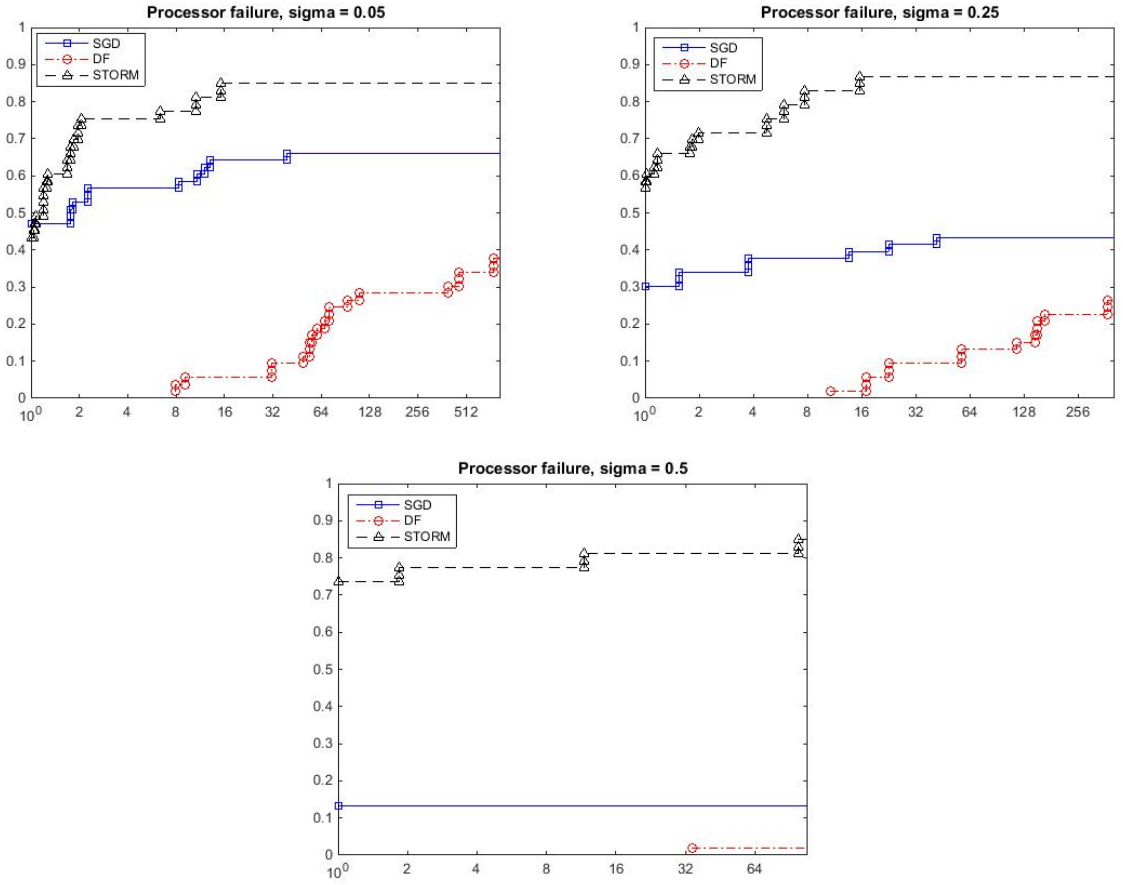
$$\tilde{g}_k(x) = \frac{1}{2\mu}[\tilde{f}_{p_k}(x+\mu e_1) - \tilde{f}_{p_k}(x+\mu e_1), \ldots, \tilde{f}_{p_k}(x+\mu e_n) - \tilde{f}_{p_k}(x-\mu e_n)]^\top$$

Finally, we compare these two methods against a naive implementation of stochastic gradient descent, shown in Algorithm 4.

Once again, since we wish to stress that the most accurate solutions are found by our solver the fastest, these performance profiles have tolerance $\tau = 10^{-5}$.

Algorithm 4 NAIVE STOCHASTIC GRADIENT DESCENT

1: (Initialization): Choose an initial point x_0. Choose step size constant $\gamma > 0$. Choose finite differencing parameter $\mu > 0$. Set $k \leftarrow 0$, $x \leftarrow x_0$.
2: Compute $\tilde{f}(x \pm \mu e_n)$, one at each of the $2n$ processors.
3: Compute approximate gradient $\tilde{g}_k(x) = \frac{1}{2\mu}[\tilde{f}(x+\mu e_1) - \tilde{f}(x-\mu e_1), \ldots, \tilde{f}(x+\mu e_n) - \tilde{f}(x-\mu e_n)]^\top$.
4: $x_k \leftarrow x_k - \frac{\gamma}{k+1}\tilde{g}_k(x)$.
5: : $k \leftarrow k+1$ and return to Step 2.



The results of this experiment show that, as expected, a 'confidence-building' strategy like in Deng and Ferris is largely useless in this scenario. As noise increases, we also see a preference for STORM over stochastic gradient descent, which we argue is due to the step-accepting criteria in STORM.

References

[1] F. Bach and E. Moulines. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Advances in Neural Information Processing Systems 24: 25th Annual*

Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain., pages 451–459, 2011.

[2] A. Bandeira, K. Scheinberg, and L.N. Vicente. Convergence of trust-region methods based on probabilistic models. *SIAM Journal on Optimization*, 24(3):1238–1264, 2014.

[3] S.C. Billups, P. Graf, and J. Larson. Derivative-free optimization of expensive functions with computational error using weighted regression. *SIAM Journal on Optimization*, 23(1):27–53, 2013.

[4] K.H. Chang, M.K. Li, and H. Wan. Stochastic trust-region response-surface method (strong) - a new response-surface framework for simulation optimization. *INFORMS Journal on Computing*, 25(2):230–243, 2013.

[5] R. Chen. *Stochastic Derivative-Free Optimization of Noisy Functions*. PhD thesis, Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, USA, 2015.

[6] A.R. Conn, K. Scheinberg, and L.N. Vicente. Global convergence of general derivative-free trust-region algorithms to first- and second-order critical points. *SIAM J. on Optimization*, 20(1):387–415, April 2009.

[7] A.R. Conn, K. Scheinberg, and L.N. Vicente. *Introduction to Derivative-Free Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2009.

[8] G. Deng and M.C. Ferris. Variable-number sample-path optimization. *Mathematical Programming*, 117:81–109, 2009.

[9] R. Durrett. Probability: theory and examples. cambridge series in statistical and probabilistic mathematics. *Cambridge University Press, Cambridge*, 105, 2010.

[10] S. Ghadimi and G. Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Mathematical Programming*. To appear.

[11] S. Ghadimi and G. Lan. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

[12] S. Ghosh, P.W. Glynn, F. Hashemi, and R. Pasupathy. How much to sample in simulation-based stochastic recursions? *SIAM Journal on Optimization*, To appear.

[13] A.B. Juditsky and B.T. Polyak. Acceleration of stochastic approximation by averaging. *SIAM J. Control Optim.*, 30(4):838–855, July 1992.

[14] J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 22(3):462–466, 1952.

[15] J. Larson and S.C. Billups. Stochastic derivative-free optimization using a trust region framework. 2013. Technical report.

[16] S. Monro and H. Robbins. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

[17] J.J. Moré and S.M. Wild. Benchmarking derivative-free optimization algorithms. *SIAM Journal on Optimization*, 20(1):172–191, 2009.

[18] R. Pasupathy and S. Ghosh. Simulation optimization: A concise overview and implementation guide. In *TutORials in Operations Research*, chapter 7, pages 122–150. INFORMS, 2013.

[19] M.J.D. Powell. UOBYQA: unconstrained optimization by quadratic approximation. *Mathematical Programming*, 92(3):555–582, 2002.

[20] P. Richtarik and M. Takac. Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function. *Mathematical Programming*, 144(1,2):1–38, 2014.

[21] S.M. Robinson. Analysis of sample-path optimization. *Mathematics of Operations Research*, 21(3):513–528, 1996.

[22] A. Ruszczynski and A. Shapiro, editors. *Stochastic Programming*. Handbooks in Operations Research and Management Science, Volume 10. Elsevier, Amsterdam, 2003.

[23] J.C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37:332–341, 1992.

[24] J.C. Spall. Adaptive stochastic approximation by the simultaneous perturbation method. *Automatic Control, IEEE Transactions on*, 45(10):1839–1853, 2000.

[25] J.C. Spall. *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. Wiley Series in Discrete Mathematics and Optimization. Wiley, 2005.