2015

Stochastic Derivative-Free Optimization of Noisy Functions

Ruobing Chen
Lehigh University

Stochastic Derivative-Free Optimization of Noisy Functions

by

Ruobing Chen

Presented to the Graduate and Research Committee

of Lehigh University

in Candidacy for the Degree of

Doctor of Philosophy

in

Industrial Engineering

Lehigh University

May 2015

Approval of the Doctoral Committee

" Stochastic Derivative-Free Optimization

of Noisy Functions"

Received and approved by the Doctoral Committee Directing the proposed program of study for Ruobing Chen, a Ph.D. Canditate in the Department of Industrial and Systems Engineering, on this date of _____.

_____
Dissertation Advisor

Committee Members:

_____
Dr. Katya Scheinberg, Committee Chair

_____
Dr. Brian Y. Chen, Dissertation Advisor

_____
Dr. Tamás Terlaky, Dissertation Advisor

_____
Dr. Stefan M. Wild, External Member

Dedication

To my parents, Yuman Chen and Xiaoling Zhang.

Acknowledgments

I am grateful to my advisor, Prof. Katya Scheinberg, for her years of guidance and support throughout my Ph.D. studies. For me, she is not only an advisor in research, but also a mentor in many aspects of my life. I own my gratitude to Dr. Stefan Wild and Prof. Brian Chen for their patient help and assistance in researching and writing parts of this thesis. I am also grateful to Prof. Tamás Terlaky, for his encouragement, insightful comments on earlier versions of the thesis and for serving on my committee.

I would like to thank Thomas Charisoulis for accompanying me through this journey with his understanding and patience. His optimism in life and hard-working spirit has greatly influenced me to thrive through the difficult times of my life. I would like to thank my parents for always being there for me as my ultimate source of love and strength during these five years away from home.

I am also grateful for support from grant number AFOSR FA9550-11-1-0239 and FIG grant from Lehigh which helped support the research in this thesis, as well as travel awards from SIAM, ICCOPT, WiML/NIPS, Rossin Doctoral Fellowship and Lehigh Graduate Student Senate.

Contents

List of Figures

List of Tables

Abstract

Optimization problems with numerical noise arise from the growing use of computer simulation of complex systems. This thesis concerns the development, analysis and applications of randomized derivative-free optimization (DFO) algorithms for noisy functions. The first contribution is the introduction of DFO-VASP, an algorithm for solving the problem of finding the optimal volumetric alignment of protein structures. Our method compensates for noisy, variable-time volume evaluations and warm-starts the search for globally optimal superposition. These techniques enable DFO-VASP to generate practical and accurate superpositions in a timely manner. The second algorithm, STARS, is aimed at solving general noisy optimization problems and employs a random search framework while dynamically adjusting the smoothing step-size using noise information. rate analysis of this algorithm is provided in both additive and multiplicative noise settings. STARS outperforms randomized zero-order methods in both additive and multiplicative settings and has an advantage of being insensitive to the level noise in terms of number of function evaluations and final objective value. The third contribution is a trust-region model-based algorithm STORM, that relies on constructing random models and estimates that are sufficiently accurate with high probability. This algorithm is shown to converge with probability one. Numerical experiments show that STORM outperforms other stochastic DFO methods in solving noisy functions.

Chapter 1

Introduction

Derivative-free optimization (DFO) is a field of nonlinear optimization that studies with methods that do not require explicit computations of the derivative information. Formally, we consider the unconstrained optimization problem

$$\min_{x \in R^n} f(x) \tag{1.1}$$

where the first (and second, in some cases) derivatives of the objective function $f(x)$ are assumed to exist and be Lipschitz continuous. However, explicit evaluation of these derivatives is assumed to be impossible. The particular focus of this thesis is the case when they are unavailable due to the noise in the objective function evaluations. This means the algorithm only has access to noise-corrupted values

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

where ε represents the noise. Hence, the goal is to minimize the true underlying function f using only its noisy version $\tilde{f}$.

1.1 Problems with Numerical Noise

Problems with numerical noise form the key domain of Derivative-free Optimization (*DFO*) algorithms and response surface methodology [32,34,36,37,51]. The presence of random noise in the objective function, in various practical applications, is often a result of simulating large complex systems. Such computer simulations produce underlying function values, but often do not provide derivatives of these outputs with respect to the decision variables of interest. It is typical for these problems to be intrinsically nonlinear, costly to evaluate and not sufficiently explicitly defined to provide reliable derivatives. This means that approximating the derivatives of such functions by traditional finite-differencing techniques or Automatic Differentiation (AD) [12] becomes prohibitive or problematic. Though it is often, but not always, theoretically possible in these cases to extract derivative information efficiently using AD, the associated implementation procedures are typically non-trivial and time-consuming.

Designing practically efficient and theoretically tractable algorithms for solving noisy optimization problems is essential for increasing solution accuracy in many fields of science and engineering, such as biology, medicine, computer science and industrial design, to name a few. One such example that arises from the area of structural biology is the problem of finding the optimal volumetric alignment of protein structures, where the noise emerges when the overlapping volume is being approximated [22]. Other ways that the noise enters the objective function can be seen in an expensive simulation of a vehicle model as a part of a larger effort to improve fuel economy of the next generation of vehicles in the automotive design industry [84], or in the automatic tuning of algorithmic hyper-parameters where randomness comes from the stochastic nature of both the training algorithm and

sample set [88].

There are two types of noise that need to be addressed. The first type is *deterministic* noise, which often results from a discretization procedure or the finite tolerances on termination criteria in a simulator. The other type of noise is called *stochastic* noise, which may arise if there are random fluctuations or measurement errors within the simulator, for example, Monte-Carlo simulation. Figure 1.1 helps visualize the noisy function we are interested in optimizing. It is a plot of the objective of the protein alignment problem, which will be described in detail in Chapter 2. The goal is to find superpositions of protein-ligand binding cavities that maximize their overlapping volume. This problem can be restated as an optimization problem where the variable x is a vector of rotation and translation parameters of one (or more) cavities with respect to another. The objective function $f(x)$ is the negative of the overlapping volume of two or more protein structures and its evaluations are done by VASP [20] given their relative positions. Figure 1.1 shows the surface of the noisy function computed by VASP, with respect to two of the parameters, with the others fixed. It can be observed that the objective function is highly noisy, non-smooth and nonlinear. The noise can be deterministic due to the discretization precision in the protein volume approximation, or stochastic as a result of the varying random seeds within the simulator.

1.2 Derivative-free Optimization Methodologies

Motivated by applications like these, researchers in the field of derivative-free optimization have invested substantial efforts in proposing new algorithms to better optimize problems with random noise. A fairly straightforward technique to deal with the noise is a Monte Carlo procedure that relies on repeated random sampling

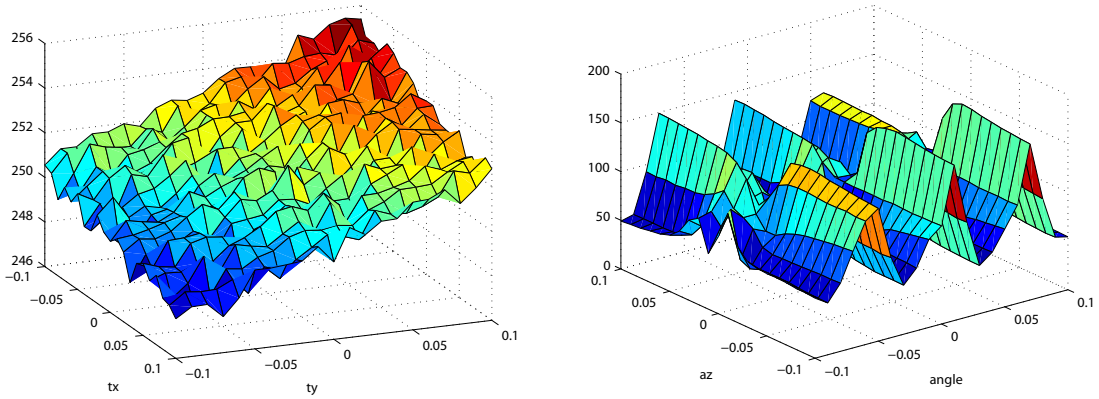


Figure 1.1: Noisy objective function.

to reduce the variance of the noise. However, one can rarely afford to do this when function evaluations are computationally costly. Some recent works have addressed the issue of noise in DFO framework, for instance, [14] proposes the use of weighted regression in a trust-region framework, [52] discusses termination criteria in the noisy environment. But there has been little theoretically sound and systematic study of approaches to noisy problems in DFO, and those that exist usually assume stochastic noise setting. For example, in [30] least square regression models for random i.i.d. noise were proposed, but no other models were analyzed or experiments were performed.

The main contribution of this thesis is the development and analysis of randomized algorithms for stochastic optimization. Algorithmically, our methods employ a rigid mechanism of choosing a search direction and a step size. While this may be sufficiently effective to produce a function decrease in the deterministic setting, it is much less likely to perform equally well in the stochastic setting. On the other hand, randomizing the traditional algorithms without substantially changing their main features, such as line search procedures and/or computation of candidate trial points, may produce robust methods when noise is present. Nevertheless, randomizing tradition algorithms while still maintaining their convergence properties is an

open question. This is precisely the topic of our work.

1.2.1 Direct Search and Random Search Methods

Randomized stochastic methods are popular alternatives to deterministic methods for simulation-based black-box problems, besides deterministic DFO algorithms such as *direct search* methods or model-based trust-region methods. The randomized schemes share a simple basic framework, allow fast initialization, and are good for large scale problems. Furthermore, there is a renewed interest in this topic in the recent literature, primarily because of their provable convergence rate. Complexity results for solving both convex and nonsmooth nonconvex functions are readily available for randomized algorithms [38, 63, 79]. However the practical usefulness of these methods are not as promising, with the fixed step sizes determined by the complexity analysis. Nonetheless, their work greatly inspired our own efforts in Chapter 3.

We now review the framework of randomized direction method and some recently developed algorithms of this type. Some of them are used for comparison with our proposed algorithm in Chapter 3. As introduced in [55], *Random optimization approach* applies to the problem $\min_{x \in R^n} f(x)$, where f is a differentiable function. At every iteration k, a point x_{k+1} is randomly sampled with Gaussian distribution around the current point x_k. If $f(x_{k+1}) < f(x_k)$, the current iterate is updated to x_{k+1}. Polyak [65] improved this scheme by describing step size rules

$$x_{k+1} = x_k - h_k \frac{f(x_k + \mu_k u) - f(x_k)}{\mu_k} u,$$

where convergence is proved for $\mu_k \to 0$ but no convergence rates are established nor specific rules given for choosing the parameters that are involved.

In [63], Nesterov recently presented four derivative-free random search schemes and obtained the theoretical bounds for their performances. In particular, the random gradient method (RG) for smooth optimization is a random version of the standard primal gradient method, while its accelerated version FG is a random variant of the fast gradient method. It was shown that the iteration complexity of FG for finding a solution x^* such that $f(x^*) - f^* \leq \epsilon$ can be bounded by $\mathcal{O}(n^2/\epsilon^2)$. Furthermore, the author extended the work by proposing random search for non-smooth and stochastic optimization, and random search for non-convex optimization.

Different improvements of these random search ideas emerge in the latest literature. For instance, incorporating the Gaussian smoothing technique [63], Ghadimi and Lan [38] presented a randomized stochastic gradient free (RSGF) method. It was shown that its iteration complexity for finding the ϵ-solution, i.e., a point $\bar{x}$ such that $\mathbb{E}[\|\nabla f(\bar{x})\|] \leq \epsilon$, can be bounded by $\mathcal{O}(n/\epsilon^2)$, and this rate, in the smooth convex cases, improves Nesterov's result in [63] by a factor of $\mathcal{O}(n)$.

Stich et al. [79] presented Random Pursuit algorithm (RP), which relaxes the requirement in [63] of approximating directional derivatives via a suitable oracle. Instead, after choosing direction uniformly at random from the hypersphere, the step sizes are determined by a line search procedure. In their implementation, the built-in MATLAB routine fminunc.m is used as the approximate line search oracle. It was shown that RP meets the convergence rates of the standard gradient method up to a factor of $\mathcal{O}(n)$. Furthermore, inspired by Nesterov's FG scheme, an accelerated Random Pursuit algorithm (ARP) was presented.

Another randomized method introduced in [74] is called Adaptive Step Size Random Search Method ((1+1)-Evolution Strategy (ES)). Instead of using pre-calculated step sizes or line search oracles, the adaptive step size random search method dy-

namically controls the step size as to approximately guarantee a certain probability of finding an improving iterate.

Encouraged by the success of random search methods, we propose a new algorithm for unconstrained derivative-free noisy optimization, Our algorithm, named STARS (STep-size Approximation in Randomized Search) relies on a near-optimal forward difference approximation of the directional derivative of a noisy function to determine the smoothing step size. The main idea is that with appropriately chosen adaptive step sizes, the randomized scheme can be utilized to reduce the effects of the noise in the objective function evaluations. We provide convergence rate analysis of our method in both additive and multiplicative settings. Computational experiments show positive results supporting this idea.

1.2.2 Trust-region Methods

All derivative-free methods rely on sampling the objective function at one or more points at each iteration. Starting from the early 90s a variety of direct search methods have been developed [1, 2, 6, 53, 82, 83] accompanied by convergence theory. These methods are inherently slow for problems of more than a few variables, because they are not able to use gradient or curvature information and they rarely reuse the sample points. New efficient model-based trust-region methods were developed in the second half of the 90's, by Powell (e.g. [66–68, 70, 71]).

With her colleagues, Scheinberg ([25], [26]) developed convergent model-based trust-region methods and a software package called "DFO", based on those methods in [29]. This package has being widely used for over a decade. The computational study of More and Wild [57] has shown that model based DFO methods are typically significantly superior in practical performance to the other existing approaches.

Most of the existing model-based DFO methods use polynomial interpolation models in place of the true objective function. The polynomial models are meant to approximate smooth functions, however, the function values produced by simulation packages are rarely smooth. As mentioned, there is often stochastic or deterministic noise added to an underlying (possibly) smooth true objective function.

While practical approaches for noisy derivative free problems have been studied extensively (e.g., see [4, 5]) most of the methods rely on a direct search framework. There has been relatively little theoretical development in the methods targeting noise in model-based derivative free optimization. Deng and Ferris [36, 37] have developed a method based on a method by Powell, which uses quadratic interpolation models. They use an average of multiple volume evaluations for each setting of the parameters to reduce the level of noise, which they assume to have the i.i.d. property. By reducing the noise to the desired level they can apply the convergence results developed for quadratic interpolation models in [26] and [31].

One of the limitations of their method is that it cannot be applied to the case of deterministic noise. Furthermore, in such a case interpolation models may not be the best choice for the approximation. One may prefer least square regression models, for instance. As the number of sample points increases, the least-squares regression solution to the noisy problem converges (in some senses and under reasonable assumptions) to the least-squares regression of the underlying true function. In [14], it has been shown that using least square regression models indeed can result in superior performance for noisy problems. Fortunately, useful model properties needed for the convergence theory in [31] can be extended to other classes of models, including the least squares regression models. Some of that theory has been further extended in [14].

In this thesis we address the above limitations. In Chapter 2, we integrate the deterministic noise level estimations into the trust-region algorithmic framework. Noise reduction strategies, such as reducing noise level and modifying the stopping criteria, are employed. In Chapter 4, we propose the use of probabilistic models and estimates in a trust-region for optimization of stochastic function. These random models and estimates are sufficiently accurate with sufficiently high probability. We prove that the trust region radii go to zero and the algorithm converges with probability one. We also discuss how to construct such models and estimates, as well as empirical performance of proposed algorithm.

1.2.3 Noise Reduction Techniques

Various ways of accounting for noise while optimizing have been explored in the literature, especially for optimization without derivatives. Some reduce noise to obtain more accurate function evaluations, for instance, by using different types of averaging techniques. Some others in fact do not directly reduce the noise at each point evaluated but instead use the noise information to adjust the algorithm for better solutions.

For functions with stochastic noise, computing replications of function evaluations is a simple way to modify existing algorithms. One can sample multiple replications per point and compute the average. There exist various methods for determining the number of replications and many of them reply on using probabilistic characterization of the variability. For instance, Deng and Ferris [36] modifies DIRECT [46]. Bayesian tools are used to analytically quantify the distributions of the functional output at each point. Acquired Bayesian sample information are used to determine appropriate numbers of replications. This sampling scheme may generate different

numbers of samples for different points. Deng and Ferris [34, 37] modifies Powells UOBYQA [69], which uses quadratic interpolation models. To reduce the variance of the quadratic model, they generate multiple function values for each point and use the averaged function values for interpolation. Bayesian posterior distributions of the model parameters are analytically quantified to help determine the appropriate number of evaluations. The noise is assumed to have the i.i.d. property. By reducing noise to the desired level they can apply the convergence results developed for quadratic interpolation models in [26] and [31]. Similar to these is [81] which modifies Nelder-Mead [62].

Other practical approaches for noisy derivative free problems without explicitly reducing the noise have been studied extensively. Most of the methods rely on a direct search framework. The implicit filtering algorithm described in [39] builds upon coordinate search and then constructs interpolation model to obtain an approximation of the gradient. It assumes that the noise goes to zero as x tends to the optimal points to obtain superlinear convergence in the terminal phase of the iteration. Many global optimization approaches for nonsmooth optimization are also shown to be effective in solving noisy problems. These algorithms are designed to avoid getting trapped in a local minima. [5] is hybrid algorithm for nonsmooth constrained optimization. It retains the convergence properties of Mesh Adaptive Direct Search (MADS), and allows the far reaching exploration features of Variable Neighborhood Search (VNS) to move away from local solutions. Another variation of the direct search algorithm [4] is proved to converge when the noise approaches zero faster than the step size.

One of the limitations of their method is that it cannot be applied to the case of deterministic noise. Kelley [47] considers a high-frequency low-amplitude perturbation of a smooth function and proposes a technique to detect and restart Nelder-Mead

methods, reinitializing the simplex to a smaller one with orthogonal edges which contains an approximate steepest descent step from the current best point. Neumaiers SNOBFIT [42] algorithm accounts for noise by combing a global search that proceeds by partitioning the search region into boxes, with a local search that fits a linear least squares model. Moreover, the computational results of these methods are not quite promising. Except [4], the other methods are not well-known and frequently used for efficiently solving noisy functions.

In model-based derivative free optimization, there has been relatively little theoretical development in the methods targeting noise. In such a case interpolation models may not be the best choice for the approximation. One may prefer least square regression models, for instance. As the number of sample points increases, the least-squares regression solution to the noisy problem converges (in some senses and under reasonable assumptions) to the least-squares regression of the underlying true function. In [14], it has been shown that using least square regression models indeed can result in superior performance for noisy problems. Fortunately, useful model properties needed for the convergence theory in [31] can be extended to other classes of models, including the least squares regression models. Some of that theory has been further extended in [14], where weighted regression models in a classic trust-region framework are tested out for optimization of functions with both stochastic and deterministic noise.The geometry of sample sets for least squares regression models for handling noise was discusses in [30].

Our noise reduction strategies in Chapter 3 [22] are a combination of these interesting ideas. They are effectively designed to tackle controllable, stochastic and biased noise associated with the VASP volume computation. We consider averaging over resolutions to introduce more randomness than regular averaging, so the noise gets smoothed out better. It is also considered to directly reduce the noise in

the exchange for reduced runtime. Least-squares regression is utilized in the classic trust-region framework. Moreover, as the noise is controllable unlike many other noise settings that have been studied, a dynamic accuracy increment technique is used to achieve better solutions. From a theoretical point of view, we prove the first-order convergence in Chapter 4, with probability one, of a simple trust-region method with random models for optimizing stochastic functions. Regarding to the proposed randomized search algorithm in Chapter 3, it is a modification of a standard random search method, where the variance of the noise is utilized to obtain optimal smoothing step size that best approximates the directional derivative at each iteration.

1.3 A Brief Outline of the Thesis

The remainder of this thesis is organized as follows.

In Chapter 2 (accepted in [22]; joint work with Dr. Brian Chen and Dr. Katya Scheinberg), we propose DFO-VASP as a specialized solver for the optimization of the protein alignment problem. First we review the background and how the objective function is a result of a complex noisy simulation. Then, we present a new DFO method that integrates the deterministic noise level estimations into the trust-region algorithmic framework. A noise-reduction strategy is employed to handle the presence of deterministic noise. Experiments on biological instances are presented to illustrate the accuracy and practical efficiency of our method.

In Chapter 3 (joint work with Dr. Stefan Wild), we introduce the STARS algorithm for optimizing general noisy functions with additive or multiplicative noise. STARS uses dynamic noise-adjusted smoothing step sizes and thus is specialized

for noisy functions. We start with a review of the random search methods and terminologies. Then, the STARS is described. The convergence rate analysis for both additive and multiplicative noise case is provided. Lastly, numerical studies reveal that STARS exhibits noise-invariant behavior with respect to different levels of stochastic noise and STARS outperforms selected randomized zero-order approaches on functions with additive and multiplicative noise.

In Chapter 4 (joint work with Dr. Katya Scheinberg and Matt Menickelly), encouraged by the success of the model-based method in Chapter 2, we propose an extension of this class of methods by incorporating probabilistic models. We first review trust-region methods and polynomial models. Then, we describe the STORM algorithm and give formal definitions of the probabilistic models and estimates. We provide convergence results and methods to construct probabilistic models via the use of error bounds from the literature on learning theory. Some preliminary computational results are presented.

Lastly Chapter 5 contains concluding remarks and directions for future research.

Chapter 2

Aligning Protein Cavities by Optimizing Superposed Volume [1]

In this chapter, an improved DFO algorithm, DFO-VASP, is proposed for solving the problem of finding *optimal superposition* in protein structure comparison. Algorithmically, this method takes care of both the stochastic noise and the controllable deterministic noise in the objective function evaluations. It incorporates noise-handling strategies that utilize the noise level estimations in the trust-region framework, and multi-start strategies to explore a globally optimal solution. Biologically, experimental results verify that the superpositions we discover are logical alignments of ligand binding sites, then we demonstrate that DFO-VASP generally discovers cavity superpositions with similar or occasionally larger overlapping volume than that of superpositions generated with existing means. Finally, we demonstrate on a large scale that similarities and variations discovered from DFO-VASP superpositions correspond to similarities and differences in ligand binding specificity.

[1] THIS CHAPTER IS AN EXPANDED VERSION OF A PAPER OF THE SAME TITLE [22] COAUTHORED BY KATYA SCHEINBERG AND BRIAN Y. CHEN.

2.1 Volumetric Alignment of Protein Binding Cavities

Many fields of molecular biology study the interaction of proteins with small molecules (*ligands*). The main focus is on determining how proteins function in a larger biological system. Proteins that perform the same function often further specialize by preferring to bind with certain molecules. This is called the property of preferential binding specificity. Understanding why proteins prefer to bind certain molecular partners and not others is the subject of tremendous scrutiny in many fields of biology and medicine. Preferential binding, or specificity, shapes the organization of molecular interactions in biological systems. Cavity regions that have similar shape may be essential for accommodating the same molecular fragment, while regions that do not may cause differences in binding specificity [16, 17, 20].

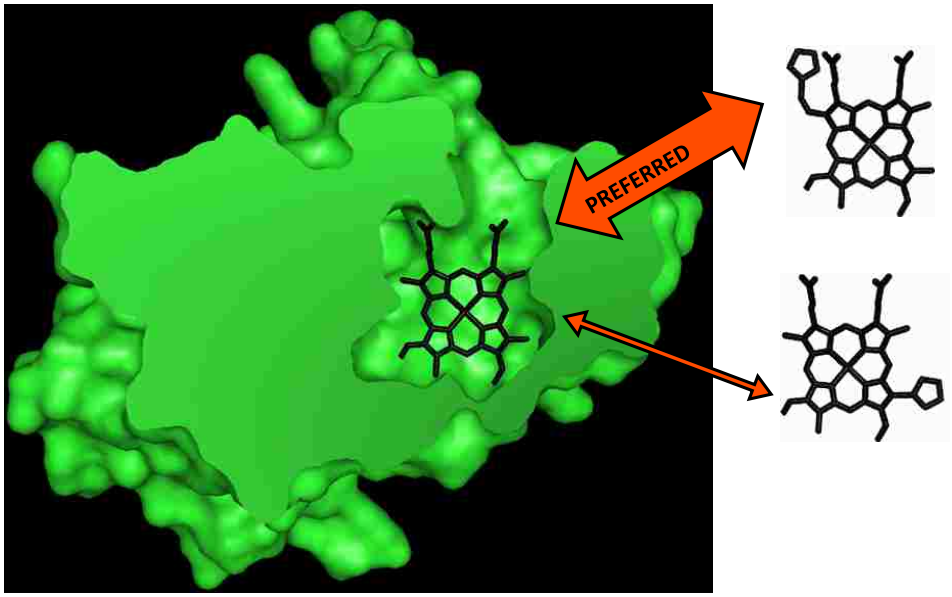
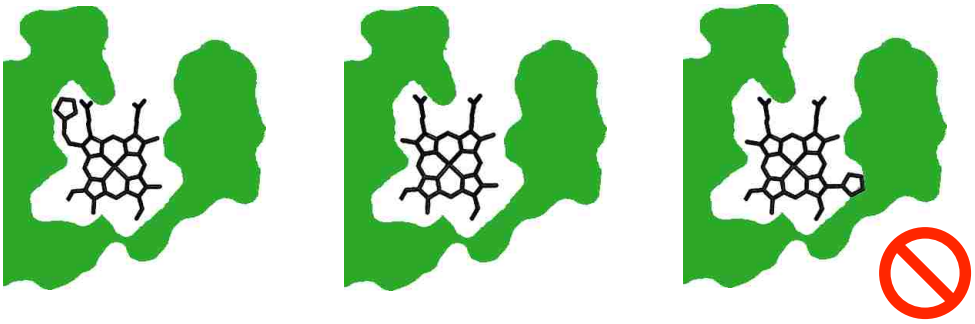
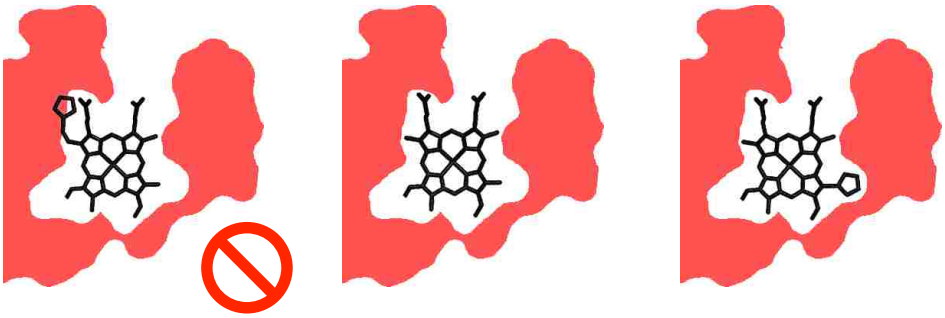


Figure 2.1: Protein-ligand binding.

To understand how specificity is achieved, structural biologists examine the molec-



(a) This protein structure does not bind with the third ligand.



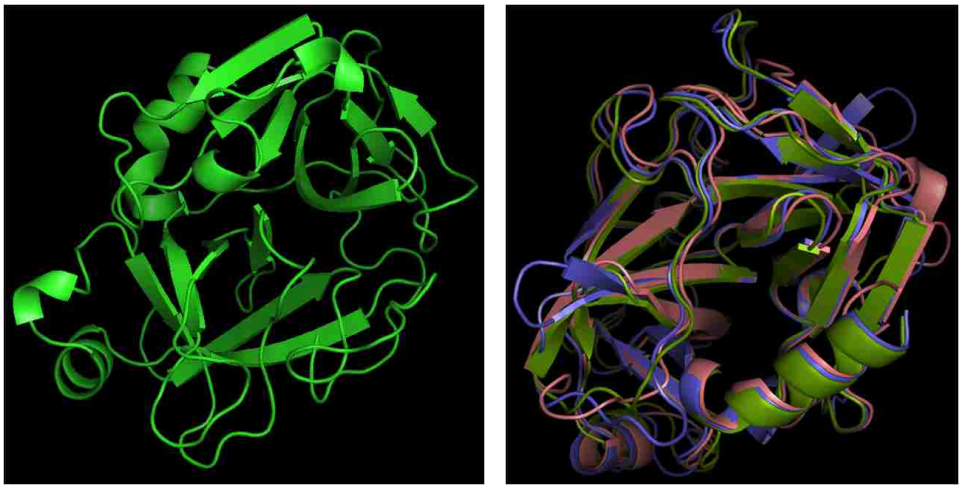
(b) A similar protein structure does not bind with the first ligand instead.

Figure 2.2: An illustration that shows that proteins with the same function can have different specificities.

ular shape, charge, and other biophysical properties of proteins to identify which parts of the protein influence specificity, and how they do so.

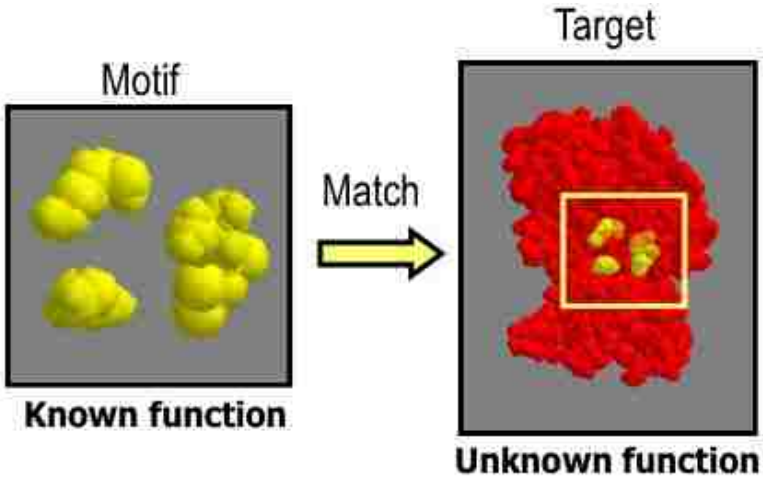
One way to examine these properties is to visualize three dimensional superpositions of two or more proteins. These superpositions can illustrate where the proteins are similar, and where they are different. At binding sites, where proteins interact with other molecules, similarities might assist in stabilizing similar molecules. Differences in shape or charge at other parts of a binding site can accommodate binding partners of one protein that cannot be accommodated by the other. One example of such a difference between two binding cavities could be a cleft in one cavity that creates more free space than in another, permitting differently shaped molecules to bind. Figure 2.2 illustrates such an example. Two proteins with the same function,

one colored in green and one colored in red, prefer to bind with different ligands due to subtle differences in cavities.



(a) The backbone - tertiary structures.

(b) Backbone alignments find similarity among this family of proteins.



(c) Other methods aligns motifs around active site. Similar functional sites imply similar function.

Figure 2.3: Atom-based alignment methods.

Making observations like these depends on accurate superpositions of protein structures. An accurate superposition should align similar elements of shape or charge as much as possible, to avoid mischaracterizing them as differences that accommodate different binding partners. An ideal superposition should also accen-

tuate actual structural and electrostatic differences and not let them be obscured by incidental similarities. Current techniques for generating superpositions are not equipped to detect all such similarities and differences, creating shortcomings in the design of structural alignment algorithms.

Existing protein structure comparison algorithms almost universally rely on geometric alignments of atomic coordinates, that is superposing corresponding atoms in two or more protein structures [15, 19, 41, 75, 85–87]. This kind of superposition ensures that many atoms overlap, enabling similar proteins to be well aligned. One class of such methods [64,75,86], as seen in Figure 2.3(a) and 2.3(b), examines protein evolution by generating and comparing alignments of whole protein structures based on their corresponding backbone atoms. These algorithms can find relationship in the continuous space of protein folds. Similar to these methods are algorithms [18, 19], illustrated in Figure 2.3(c), that find similar functional sites by using motifs to represent a known functional site and searching a target structure for a set of amino acids in the same configuration as the motif. If such amino acids are found, it suggests that the target has the same functional site as the motif.

However, these methods have two major shortcomings. First of all, the required correspondences between atoms cannot be fully constructed between sidechain atoms, because sidechains have different lengths. Two proteins might have different number of atoms, so it's impossible find a one-to-one comparison between atoms, i.e., an atomic alignment. This underlying variability forces atom-based superpositions to simplify amino acid geometry into backbone-only [19, 41, 75, 86, 87], surrendering detail. Second limitation shown in Figure 2.4 is that these methods align protein structures by the position of the atoms. They do not align protein cavities based on the open space inside the cavity. However, this open space is where the partner molecule binds and thus the similarity in that region is crucial. Moreover, while

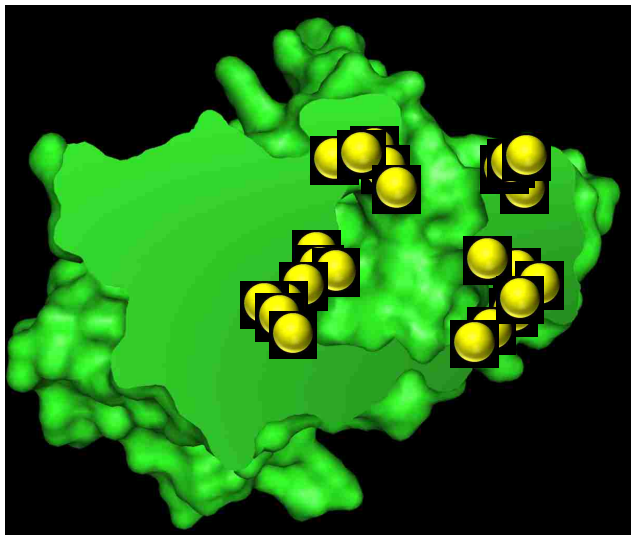


Figure 2.4: An illustration that shows the limitations of atom-based approaches. The position of atoms are not equivalent to the shape of the cavities.

electrostatic potentials can be represented at the molecular surface [49] or labeled on specific atoms, the electrostatic field is not represented at longer ranges that are remote from the protein. Superpositions are thus unable to incorporate the general shape of the electrostatic field into the alignment. The work described below explores an alternative approach to comparative superposition that mitigates these issues.

Our goal is to find superpositions of protein-ligand binding cavities that maximize their overlapping volume. And this problem can be restated as an optimization problem where the variable x is a vector of rotation and translation parameters of one (or more) cavities with respect to another. The number of parameters for optimization can range from seven (three specifying the rotation axis, one specifying the rotation angle, and three specifying the translation) to multiples of seven, depending on the number of structures we choose to align. The objective function $f(x)$ is the negative of the overlapping volume and its evaluations are done by VASP [20]. VASP approximately computes the volume of the intersection of two or more protein structures given their relative positions. Hence the task we face here is: given two or more protein structures find *optimal superposition* - the values of rotation and

translation parameters for each of them to maximize the volume of the intersection.

2.1.1 VASP Software

VASP [16,17,20] evaluates overlapping volume using marching cubes [54], a technique for generating a polyhedral surface for a closed three dimensional volume. In the abstract, this process identifies the overlapping region of two areas A and B by first decomposing space into a fine cubic lattice. A cubic lattice can be described as a series of cubes, segments of cubes, or a series of points that form the corners of cubes. Marching cubes operates by identifying the corner points that are inside both A and B. We refer to these as interior points. The cube segment between any interior point and a non-interior point must exit the overlapping region. For all such cube segments, marching cubes identifies the intersection points between the cube segment and the boundary of the intersecting region. Finally, the set of all intersection points are combined to create a polyhedral mesh that approximates the intersecting region. The volume of the intersecting region can be calculated using the Surveyor's Formula [72].

The nature of this approximation affects the accuracy of DFO-VASP: Intersec-

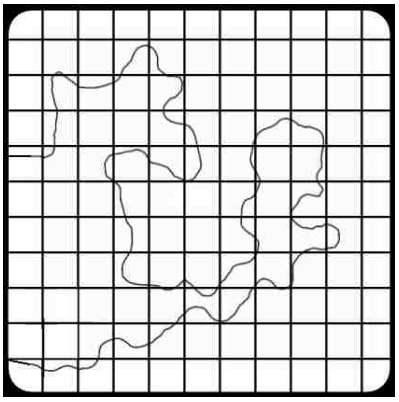


Figure 2.5: Marching cube method: how the protein volume is approximated.

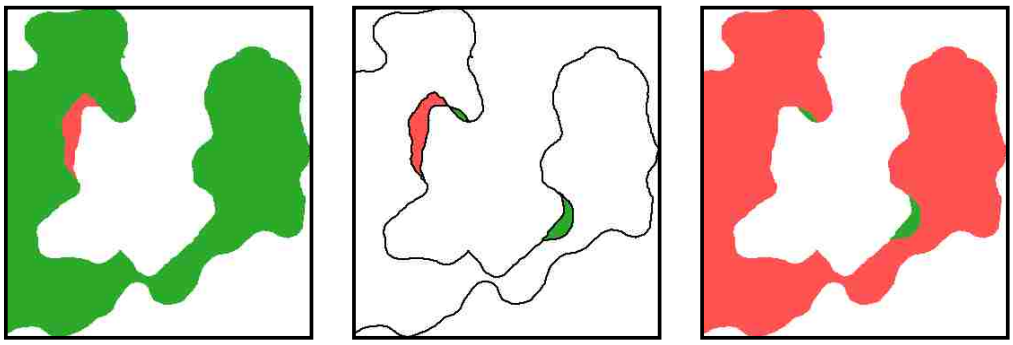


Figure 2.6: VASP isolates differences in cavity shapes.

tions computed on lattices with finely sized cubes are more precise approximations of the surface, while lattices with coarser cubes have greater, though bounded, inaccuracies. As one surface is rotated and translated in the search for increasingly greater overlapping volumes, the surface intersects the lattice, which remains axis aligned, in different ways, generating noise in the approximation. This noise can be substantial because of the complex shape of molecular surfaces and cavities based on the molecular surface. To capture the complexity of the molecular surface with higher accuracy requires higher resolution of the lattice used in DFO-VASP. Higher resolutions are essential when smaller cavities are aligned, but they also result in a larger computational burden, as we will illustrate bellow.

VASP presents us with an ideal testing environment for developing various robust DFO methods for noisy problems: this noise in VASP can be deterministic or stochastic (depending on the algorithmic setting) and it is significant enough to cause standard DFO implementations to fail to converge to the proximity of a local minimizer. On the other hand, since the noise comes from a 3D approximation of the volume, the deterministic noise component can be controlled to a certain degree at an additional computational cost spent on increasing the accuracy of the volume approximation. Furthermore, the number of parameters for optimization can range (starting from seven) depending on the number of structures we choose to align. This

allows us to perform quick testing of many ideas on small scale noisy problems and then expand the testing to similar problems of larger scale.

2.1.2 Alignments of Electrostatic Data

In addition to molecular shape, other electric fields also influence function. To consider this second range of data, DFO-VASP can also be used to superpose electrostatic isopotentials. Electrostatic isopotentials represent a spatial region where positive electrostatic potentials are greater than a given threshold, or negative electrostatic potentials are smaller than a given threshold. Because electrostatic isopotentials are necessarily closed regions, the superposition of two isopotentials can be achieved by the same general approach as the superposition of ligand binding cavities. Electrostatic potentials used here represent entire proteins rather than regional binding sites, and electrostatic potentials can be generated at different thresholds for different comparison purposes.

Electrostatic isopotentials, especially of whole proteins, can be dramatically larger than ligand binding cavities. Differences in size requires different resolution thresholds to be considered, to maintain efficiency. While coarser resolutions exhibit greater absolute inaccuracy, relative to isopotential volume, inaccuracy from noisy comparison is no larger than for ligand binding cavities. For this reason it is essential for DFO-VASP to adjust the range of resolutions considered in the superposition problem when considering isopotentials. In Section 2.3.2 we will illustrate the range of resolutions that we found efficient for aligning isopotentials. Considering the superposition of electrostatic isopotentials enables us to critically examine how DFO can be used to generate efficient superpositions, in spite of noise and very diverse data.

2.2 Description of the Basic DFO Method

2.2.1 Objective

We consider the problem of maximizing the overlapping volume in the protein alignment as an unconstrained minimization problem

$$\min_{x \in R^n} f(x). \tag{2.1}$$

VASP software approximately computes the volume of intersection of two or more protein structures (or their parts) given their relative positions, i.e., rotation and translation. Hence, in this case x defines the relative position and the number of parameters for optimization can range from seven (three specifying the rotation axis, one specifying the rotation angle, and three specifying the translation vector) to multiples of seven, depending on the number of structures one chooses to align. Figure 2.7 shows the meaning of the seven variables when aligning two protein structures. tx, ty, tz denote the translation parameters in the 3D space. ax, ay, az define a rotation vector around which a rotation indicated by the variable *angle* will be performed. These seven variables uniquely defines a relative superposition from the initial position.

In order to normalize the rotation axis, we need to add an equality constraint, which in the case of two protein alignment can be expressed as $\|x_a\| = 1$ where $x_a \in R^3$ is a vector with the entries being the first three entries of $x \in R^7$. However, to avoid solving problems with nonlinear constraints, we simply move the constraint into penalty function, $\lambda(\|x_a\| - 1)^2$. Since this constraint only serves to eliminate multiple and badly scaled solutions, it does not have to hold exactly. By choosing a small

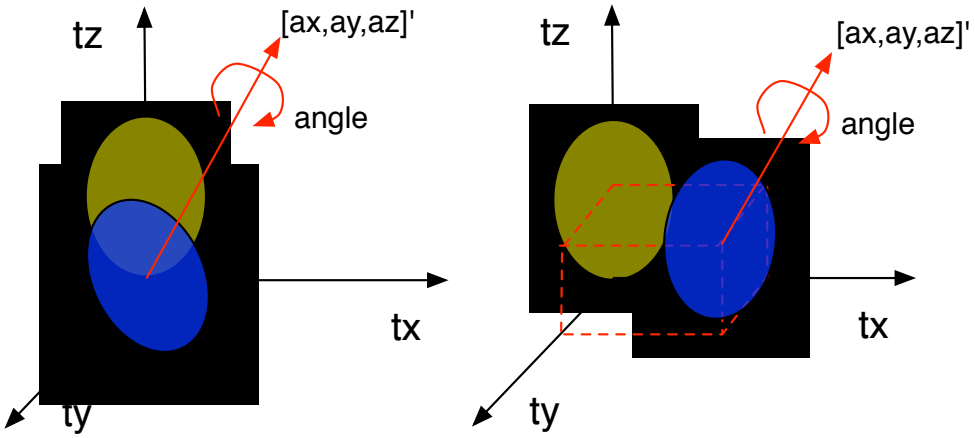


Figure 2.7: An illustration that shows the meaning of variables in the protein alignment problem.

and constant value for the penalty parameter λ we produce a stable unconstrained formulation for our problem.

2.2.2 Algorithmic Framework

Trust-region algorithms are iterative algorithms for solving (2.1). In each iteration of these algorithms, given current point x_k, one constructs a model $m_k(x_k + s)$ for the objective function that sufficiently approximates the objective for all perturbations s belonging to the "trust region" $\mathcal{B}(x_k, \Delta_k)$, where Δ_k is known as the radius of the trust region. The model function $m_k(x_k + s)$ is then minimized (possibly approximately) in $\mathcal{B}(x_k, \Delta_k)$ to define a trial step x_k^+, and a trial function value $f(x_k^+)$. If the change in the function value $f(x_k) - f(x_k^+)$ is bigger than a certain fraction of the change $m_k(x_k) - m_k(x_k^+)$ anticipated on the basis of the model, the iteration is deemed "successful", and the trial point is accepted as the new iterate, the model is updated and the trust-region radius is possibly increased. If, on the contrary, the reduction in the objective function is too small compared to that predicted by the

model, the iteration is deemed "unsuccessful", the trial point is rejected and the trust-region radius is decreased.

Eventually, the algorithm stops its execution when the step size parameter is below a given threshold. See [28] for a detailed description of trust-region algorithms. Thus, the trust region algorithmic framework can be roughly described as follows. The model-based DFO algorithm that we use is based on a trust-region framework described in [32]. This framework relies of constructing, so-called, fully-linear interpolation models of the objective function. The definition and details on fully-linear models can be found in [32]. This trust region algorithmic framework can be roughly described in Algorithm 1.

This algorithmic framework has been shown to converge to a local optimal solution in the absence of noise. The numerical implementation of this algorithm terminates its execution when the step size parameter falls below a given threshold. For the purposes of theoretical guarantees a different, more computationally costly stopping criterion needs to be employed, but in practice a simple threshold strategy is used [32]. See [28] for a detailed description of trust-region algorithms.

2.2.3 Polynomial Models

In model-based DFO, the function f is (locally) approximated using a class of models. For these models to be useful they need to be sufficiently accurate, i.e. they provide a Taylor series like approximation. In [30, 32] general concepts of fully-linear and fully-quadratic models were introduced. Loosely speaking, a model $m(x)$ is said to be a *fully-linear* model of $f(x)$ in $\mathcal{B}(x;\Delta) = \{y : \|x-y\| \leq \Delta\}$, if for all $y \in \mathcal{B}(x;\Delta)$, the error between the gradient and the value of the model and the gradient and the

Algorithm 1 DFO:Basic Trust Region Algorithm

1: (Initialization) Choose an initial point x_0, trust region radius Δ_0, and an initial interpolation set $Y_0 \subset \mathcal{B}(x_0, \Delta_0)$, which in turn defines as interpolation model m_0 around x_0. Choose $\eta > 0$ and $\gamma > 1$, $1 > \theta > 0$.

2: (Criticality Step) If $\|\nabla m_k(x_k)\| < \theta\Delta_k$, reduce Δ_k and recompute a fully-linear model in $\mathcal{B}(x_k, \Delta_k)$. Repeat until, $\|\nabla m_k(x_k)\| \geq \theta\Delta_k$.

3: (Compute a trial point) Let $m_k(x)$ be the model build around an iterate x_k that is assumed to represent this function sufficiently well in a "trust region" $\mathcal{B}(x_k, \Delta_k)$. Compute x_k^+ such that

$$m_k(x_k^+) = \min_{x \in \mathbb{B}_k} m_k(x),$$

and $m_k(x_k^+)$ is "sufficiently small compared to $m_k(x_k)$".

4: (Evaluate the objective function at the trial point) Compute $f(x_k^+)$ and

$$\rho_k = \frac{f(x_k) - f(x_k^+)}{m_k(x_k) - m_k(x_k^+)}.$$

5: (Define the next iteration)

4a: Successful iteration. If $\rho_k \geq \eta$, define $x_{k+1} = x_k^+$ and choose $\Delta_{k+1} \geq \Delta_k$. Obtain Y_{k+1} by including $\{x_k^+\}$ and dropping one of the existing interpolation points if necessary.

4b: Unsuccessful iteration. If $\rho_k < \eta$, then define $x_{k+1} = x_k$ and set $\Delta_{k+1} = \gamma^{-1}\Delta_k$ if $m_k(x)$ is fully-linear. Update Y_{k+1} to include x_{k+1}.

6: (Update the model) If the model m_k is not fully-linear, then improve Y_k to get Y_{k+1}. Update $k \leftarrow k + 1$.

value, respectively, of the function satisfies

$$\|\nabla f(y) - \nabla m(y)\| \;\leq\; \kappa_{eg}\,\Delta, \qquad |f(y) - m(y)| \;\leq\; \kappa_{ef}\,\Delta^2,$$

with κ_{ef} and κ_{eg} are independent of x and Δ.

Polynomials form a particular, useful model class. Let $\mathcal{P}_n^d$ denote the set of polynomials of degree $\leq d$ in $\mathbb{R}^n$ and let $q_1 = q + 1$ denote the dimension of this space. It is clear that the dimension of $\mathcal{P}_n^1$ is $q_1 = n + 1$ and the dimension of $\mathcal{P}_n^2$ is $q_1 = \dfrac{1}{2}(n+1)(n+2)$. Let $\bar{\Phi}$ be the natural basis for $\mathcal{P}_n^2$. That is,

$$\bar{\Phi} \;=\; \{1, x_1, x_2, \ldots, x_n, x_1^2/2, x_1x_2, \ldots, x_{n-1}x_nx_n^2/2\}.$$

Any polynomial $m(x) \in \mathcal{P}_n^d$ can be written as

$$m(x) = \sum_{j=0}^{q} \alpha_j \bar{\Phi}_j(x),$$

where the α_j's are real coefficients. We say that the polynomial $m(x)$ interpolates the function $f(x)$ at a given point y if $m(y) = f(y)$.

Given a set of $p_1 = p+1$ points $Y = \{y_0, y_1, \ldots, y_p\} \subset \mathbb{R}^n$, $m(x)$ is said to be the interpolation polynomial of $f(x)$ on Y if its coefficients vector α satisfies

$$M(\bar{\Phi}, Y)\alpha \;=\; f(Y),$$

where

$$M(\bar{\Phi}, Y) \;=\; \begin{bmatrix} \bar{\Phi}_0(y^0) & \bar{\Phi}_1(y^0) & \cdots & \bar{\Phi}_q(y^0) \\ \bar{\Phi}_0(y^1) & \bar{\Phi}_1(y^1) & \cdots & \bar{\Phi}_q(y^1) \\ \vdots & \vdots & \vdots & \vdots \\ \bar{\Phi}_0(y^p) & \bar{\Phi}_1(y^p) & \cdots & \bar{\Phi}_q(y^p) \end{bmatrix} \tag{2.2}$$

and $f(Y)$ is the p_1 dimensional vector whose entries are $f(y_i)$ for $i = 0, \ldots, p$.

It has been shown in [27, 32] that if for all $Y \subset \mathcal{B}(0; 1)$ such that the condition number of $M(\bar{\Phi}, Y)$ is uniformly bounded and $p \geq n$ then the interpolation models based on Y are fully linear (belong to a particular fully linear class).

Currently the best performing interpolation models used in DFO are underdetermined quadratic interpolation models with the smallest ℓ_2 norm or ℓ_1 norm of the vector of the model coefficients (Hessian of the quadratic, in particular).

Specifically, let us split the natural basis $\bar{\Phi}$ into linear and quadratic parts: $\bar{\Phi}_L = \{1, x_1, \ldots, x_n\}$, and $\bar{\Phi}_Q = \{\frac{1}{2}x_1^2, x_1x_2, \ldots, \frac{1}{2}x_n^2\}$. The interpolation model can thus be written as where α_L and α_Q are the appropriate parts of the coefficient vector α. The *minimum Frobenius norm* model are built based on the solution to the following optimization problem in α_L and α_Q:

$$\begin{aligned} \min \quad & \frac{1}{2}\|\alpha_Q\|^2 \\ \text{s.t.} \quad & M(\bar{\Phi}_L, Y)\alpha_L + M(\bar{\Phi}_Q, Y)\alpha_Q \;=\; f(Y). \end{aligned} \tag{2.3}$$

because, minimizing the norm of α_Q is equivalent to minimizing the Frobenius norm of the Hessian of $m(x)$.

Other alternative models will be further discussed in Chapter 4.

2.3 Noise Handling Strategies for VASP

As one can observe, the essential mechanism of the above algorithm lies in checking the function reduction in Step 5 by examining the ratio

$$\rho_k = \frac{f(x_k) - f(x_k^+)}{m_k(x_k) - m_k(x_k^+)}.$$

However, when the underlying function f is computed with noise, the ratio becomes

$$\rho_k' = \frac{f(x_k) + \epsilon_k - f(x_k^+) - \epsilon_k^+}{m_k(x_k) - m_k(x_k^+)},$$

where ϵ_k and ϵ_k^+ are unknown noise components. It is easy to see that if the noise level is comparable to $m_k(x_k) - m_k(x_k^+)$, then the information about the achieved reduction in f provided by the noisy estimate ρ_k' is possibly corrupted. Hence false steps can be taken by the algorithm; for example, a trial point x^+ may get accepted as the new iterate, while $f(x_k^+) > f(x_k)$ or, alternatively, the trust region radius may get reduced and the step may get rejected, while $f(x_k^+) < f(x_k)$.

Experiments also show that, running the Algorithm 1 for noisy optimization purposes frequently leads to unsatisfactory early termination. Due to the rapid reduction in trust region radius and dominating number of unsuccessful steps, the algorithm stops far away from the optimum most of the time. Therefore a special modification of this algorithm is necessary for the noisy VASP evaluations. In order to resolve this problem, our new DFO algorithm incorporates various trust-region maintenance strategies and noise reduction strategies that utilize the estimates of existing noise to produce sufficient successful reduction steps.

2.3.1 Noisy Analysis and Reduction

The presence of relative noise in the function values introduces a great deal of difficulty in optimization [58]. Fortunately, in the case of VASP volume evaluations, the level of noise can be reduced by two strategies, i.e. by *averaging* and *direct noise level reduction*. Both of these approaches control the noise level by utilizing "resolution", an input parameter for VASP. The "resolution" represents the lattice cube size which VASP uses to discretize the shapes of protein structures. Smaller resolution means a finer lattice is used to approximate the shapes, which in turn means high accuracy of the estimates, but also larger number of corner points that need to be examined and larger computation time. There is thus a trade-off between noise level and computational cost. We seek to exploit this tradeoff to reduce the overall computational effort.

The first approach to reducing the noise is by simple *averaging*. We can consider the standard Monte-Carlo simulation approach, used by Ferris and Deng [36,37], for instance. In this case by computing multiple function values and averaging them we can reduce the noise level. Since the noise is not random, we introduce a randomization component. We call our approach *averaging over resolution*.

We take advantage of the fact that the larger grid sizes (i.e., resolution) correspond to fast volume computations and that the noise produced by VASP using slightly different grid sizes is nearly random. The latter fact implies that a small change in the grid size results in a random change in the noise component, but the accuracy of the computation is roughly the same. Hence, by computing multiple function values with different resolutions and averaging these values, we can reduce the noise level and get better average estimate than each individual estimate. Figure 2.8 illustrates how the averaged function surfaces get smoother by averaging surfaces.

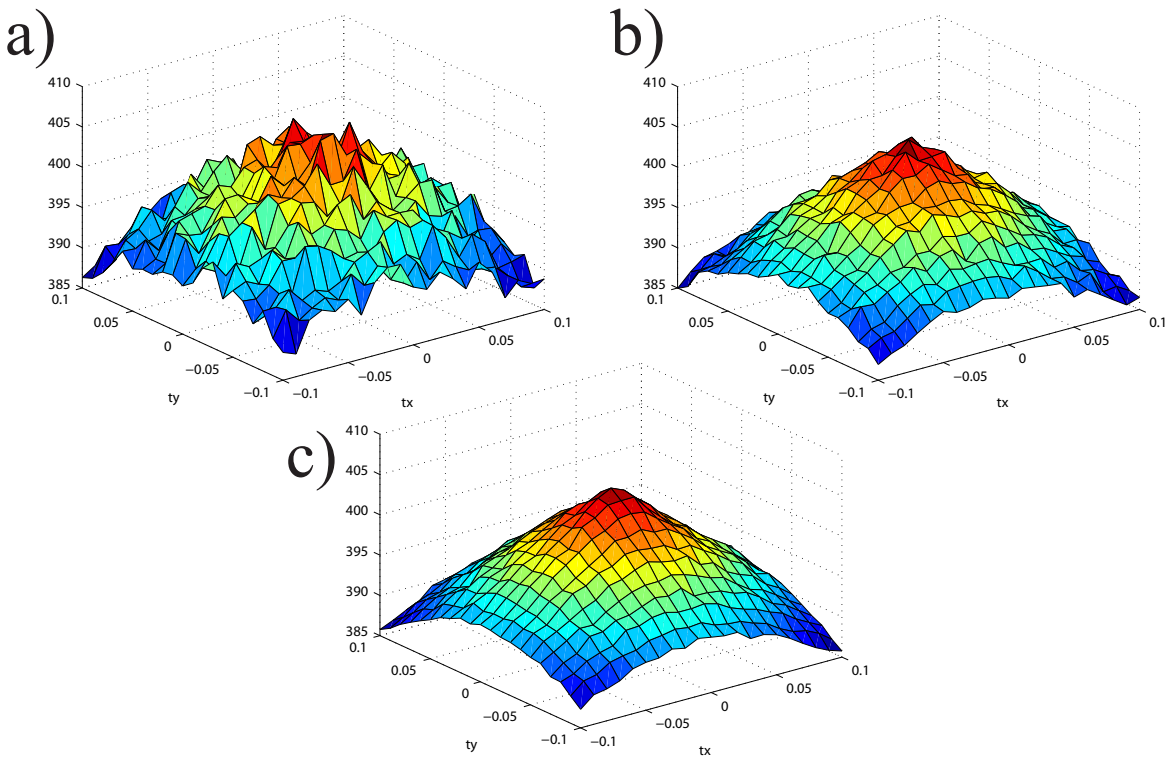


Figure 2.8: Averaging: MC simulation. (a) Resolution .5, time: 710 seconds. (b) Resolution .5, .53, .57, .6, Time: 1510 seconds. (c) Resolution .5, .51, .52, ...6, Time: 3250 seconds.

We plot the objective function by varying two variables tx and ty while fixing the remaining five variables as constants. (a)-(c) are computed with 1, 4 and 11 distinct "resolution" values. The corresponding runtimes are recorded.

It turns out that while *averaging* reduces the noise level, it does not make this level arbitrarily small. Noise only vanishes when the volume of intersection can be computed exactly, that is when the grid step size used in the discretization is zero. However, the averaging over resolution approach is always an underestimate of the true volume. Moreover, the additional computational cost is substantial in the sequential environment. We now describe the second approach, *direct noise level reduction*, which simply changes "resolution" to achieve certain level of accuracy. Figure 2.9 presents the resulting smoother function surfaces by reducing "resolution"

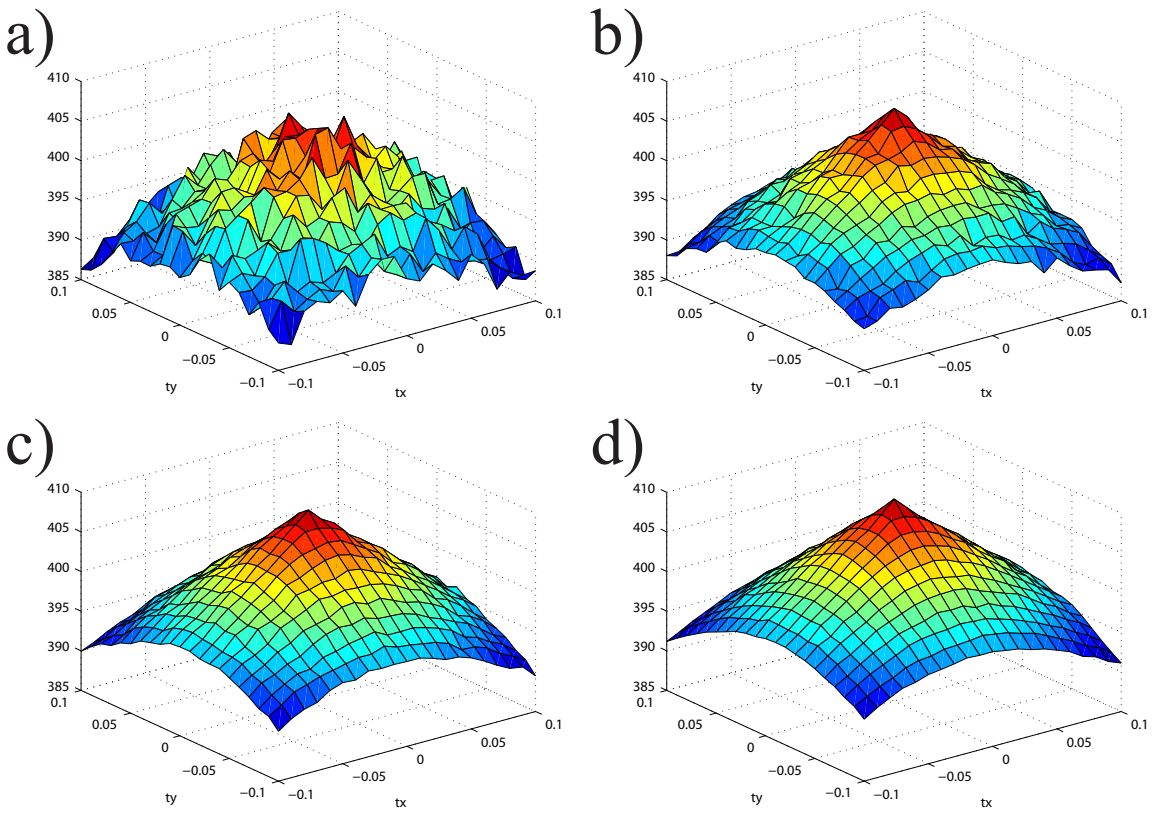


Figure 2.9: Direct noise level reduction. (a) Resolution: .5, time: 710 sec. (b) Resolution .45, time: 850 sec. (c) Resolution .35, time: 1510 sec. (d) Resolution .3, time: 2300 sec

from 0.5, 0.45, 0.35, to 0.3, and the corresponding runtimes.

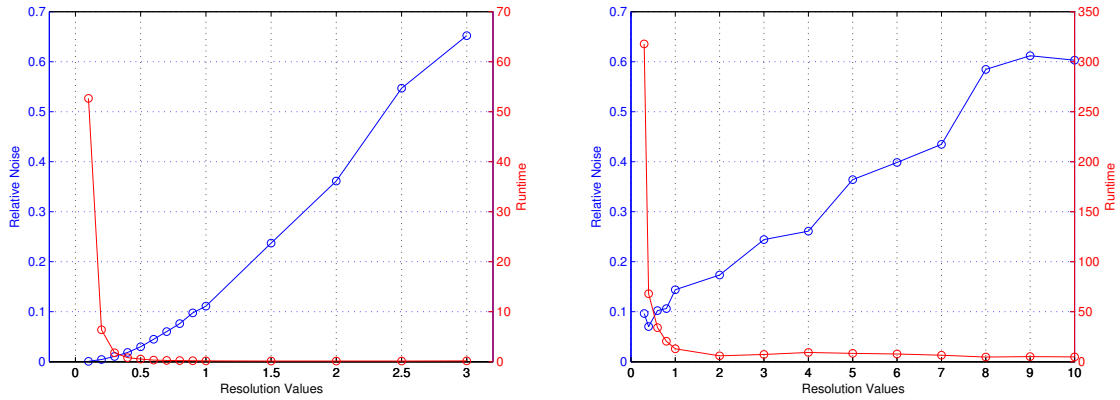
By comparing Figure 2.9 and Figure 2.8, one can observe that the trade-off in precision and computational cost in *direct noise level reduction* appears to be better than that of *averaging*. It can be observed from Figure 2.8 c) with Figure 2.9 d), there is clearly a lift of the surface in the later figure. It can be observed that, in Figure 2.9 d) where the volume is computed with 0.3 resolution , the volume is 392 at the point $[tx, ty] = [-0.1, 0.1]$, whereas the overlapping volume is only 385 at the same point when using an average of 11 distinct resolutions. Increasing the number of copies can indeed further smooth out the surface in 2.8 c), however, this difference in volume cannot be eliminated. Moreover, the increase in runtime is

substantial. While *averaging* takes 3250 seconds, *direct noise level reduction* only uses 2300 seconds. This result provides a justification for using *direct noise level reduction* as our major smoothing strategy.

2.3.2 Dynamic Adjustment of Accuracy

Since the estimates of the level of the noise can be computed, a dynamic strategy of adjusting the resolution parameter can be used. While the fast, low-accuracy function evaluations may be sufficient at the early stages of the algorithm, eventually the trust region radius (and hence the step size) becomes small, and so does the predicted reduction achieved by a trial step. Once the value of this reduction is comparable to the noise level, this step acceptance criterion is no longer reliable. In that case, noise level reduction becomes imperative to ensure progress. As higher accuracy evaluations take more time, we try to resort to them only when necessary. Hence, it is advantageous to increase the accuracy dynamically as the algorithm progresses.

Here we make use of the specific mechanism of our DFO algorithm. Because in our application we need to compute maximum volume alignment of many pairs or proteins, for all of whom the accuracy/time trade-offs are nearly the same, we precompute several estimates of the level of noise for different resolution values and apply them in the dynamic strategy of adjusting the resolution parameter. As the trust region radius (and hence the step size) gets smaller, so is the predicted reduction achieved by a trial step. Once the value of this reduction is comparable to the noise level, this step is no longer reliable. Only then, noise level reduction becomes imperative. Hence, we develop a dynamic accuracy increment strategy: at iteration k, given current relative noise level δ_l and a constant $\theta > 1$; if $m_k(x_k) - m_k(x_k^+) < \theta f(x_k) \cdot \delta_l$, we reduce the noise to the next level δ_{l+1}, and compute a new model in



(a) Ligand binding cavities.

(b) Electrostatic fields.

Figure 2.10: Trade-off between the relative noise and runtime in computing the volumes of ligand binding cavities and electrostatic fields.

$\mathcal{B}(x_k, \Delta_k)$. The algorithm can be formally described as in Algorithm 2.

To obtain the noise level estimates we assume that for a fixed resolution value, the noise level does not depend on the value of x (which is not true in the case of VASP, strictly speaking, but appears to produce reasonable results, since the level of noise is more affected by the resolution value more than by the change in rotation and translation). Given resolution (i.e, lattice cube size) r_v, for any x, the relative noise, is defined as

$$\delta_{r_v} = \frac{f^*(x) - f_{r_v}(x)}{f^*(x)},$$

where $f^*(x)$ represents the noise-free true function value and $f_{r_v}(x)$ is the computed function value by VASP with r_v as the resolution value. Letting r_v^* be the smallest resolution that is practically computable, the noise level can then be estimated as

$$\bar{\delta}_{r_v} = \frac{f_{r_v^*}(x) - f_{r_v}(x)}{f_{r_v^*}(x)}.$$

Figure 2.10 shows the relative noise estimation and related computational bur-

Algorithm 2 DFO-VASP: DFO ALGORITHM FOR PROTEIN ALIGNMENT PROBLEM

1: (Initialization) The noise levels $\{\delta_l\}_{0\leq l\leq lmax}$. An initial trust-region radius $\Delta_0 \in (0, \Delta_{max}]$, $\Delta_{max} > 0$. An initial poised interpolation set Y_0 of $2n+1$ points is constructed, that contains the starting point x_0. Y_0 defines an initial model m_0 (with gradient and possibly the Hessian at $s = 0$ given by g^{icb}_{k+1} and H^{icb}_{k+1} respectively). The parameters $\eta_0, \eta_1, \gamma_1, \gamma_2, \theta$, and gradient tolerance ϵ_c are given and satisfy the conditions $0 \leq \eta_0 < \eta_1 < 1$, $0 < \gamma_1 < 1 < \gamma_2, 0 < \theta < 1, \epsilon_c > 0$. Set $k = 0$ and $l = 0$.

2: (Step calculation) Compute a trial point $x^+_k = x_k + s_k$ by solving

$$\min_{s_k} m_k(x_k + s_k) \quad \text{s.t. } x_k + s_k \in B(x_k; \Delta_k).$$

3: (Noise estimation) If the current deterministic noise level is comparable to the predicted reduction achieved by a trial step, i.e.

$$m_k(x_k) - m_k(x^+_k) < \theta f(x_k) \cdot \delta_l,$$

reduce the relative noise level and increment l and k by one.
Resample Y_{k+1} and go to Step 2.

4: (Acceptance of the trial point) Compute the ratio

$$\rho_k = \frac{f(x_k) - f(x^+_k)}{m_k(x_k) - m_k(x^+_k)}.$$

Update the current iterate

$$x_{k+1} = \begin{cases} x^+_k & \text{if } \rho_k \geq \eta_0, \\ x_k & \text{otherwise.} \end{cases}$$

5: (Interpolation set update)

$$Y_{k+1} = \begin{cases} Y_k \cup \{x^+_k\} \setminus \{y_r\} & \text{if } \rho_k \geq \eta_0, \\ Y_k \cup \{x^+_k\} \setminus \{y_r\} & \text{if } \rho_k < \eta_0, \text{ but } x^+_k \text{ improves model,} \\ Y_k & \text{otherwise,} \end{cases}$$

where $y_r = \arg\max_{y_j \in Y_k} \|y_j - x_k\|_2$.

6: (Trust region update) Set trust region radius

$$\Delta_{k+1} = \begin{cases} \min\{\gamma_2 \Delta_k, \Delta_{\max}\} & \text{if } \rho_k \geq \eta_1 \\ \gamma_1 \Delta_k & \text{if } \rho_k < \eta_0 \text{ (and } m_k \text{ is fully linear on } B_k) \\ \Delta_k & \text{otherwise.} \end{cases}$$

Increment k by 1 and go to Step 2.

den with respect to r_v. We observed that as r_v decreases (approaching 0.08), the runtime increases superlinearly. This is natural, as the number of corner points that VASP needs to examine grows superlinearly as r_v decreases. We also note that the reduction in noise level decreases superlinearly to zero, while the noise level itself does not decrease to zero. Nevertheless, in our experiments, the smaller noise levels were sufficient to achieve solutions of acceptable accuracy. Lower values of r_v result in very costly function evaluations, but these levels were not necessary to obtain practical solutions for examining protein binding specificity. Similar trade-offs were observed between ligand binding cavities and whole-protein electrostatic isopotentials, demonstrating that the dynamic adjustment of accuracy is effective at absolute sizes and resolution levels.

Based on these observations, we select several noise levels based on the exchange between relative noise and runtime. We were able to choose resolution values that give a sufficient improvement in computation accuracy but avoid unnecessary calculations.

Noisy functions ofter lead to more "unsuccessful" steps and thus more trust region shrinkages than expansions. So it is necessary to reduce trust region at a much slower rate than that is usual for classical trust region settings. However, slow shrinking of the trust region may lead to slow progress towards satisfying the stopping criteria. Hence, we enforce another termination rule. That is, the smallest chosen noise level has to be reached to guarantee the solution quality. After this check, two criterions work together to stop the algorithm: either when the trust region size is smaller than a threshold value, or when the noise level is forced to be reduced again, the algorithm stops its execution. This stopping strategy worked reasonably well in our experiments, however a more aggressive strategy will be explored in future work to improve efficiency. This completes our description of Algorithm 2.

2.3.3 Warm-start v.s. Random-start

The DFO framework in [32] converges to a local stationary point. In practice this method tends to find "good" local optimal solutions, however, no guarantee of global solution can be provided. Hence different starting points may produce different final results if the optimization problem has multiple optima. Since the atomic and the maximum volume superpositions may be closely related, it is natural to use the atomic superposition as initial point for the optimization. We refer to results of this setting as the *warm-started* alignments. However, as discussed earlier, superstitions of corresponding atoms do not necessarily yield maximal overlapping volume between ligand binding cavities, and the lack of similarities in backbone structure may also make the atomic superposition a biased starting point. Therefore, as an alternative, we start our algorithm using randomly-generated alignments and investigate if we achieve further improvement.

In random-started tests the starting points are chosen by using Latin Hypercube Sampling (LHS) techniques [56], which has been successfully used in global derivative free optimization. It is a statistical method for generating a distribution of starting values of parameters from a multidimensional distribution. It selects m different values from each of n variables $X_1, \cdots, X_k$ in such a way that each sample is the only one in each axis-aligned hyperplane containing it. This LHS scheme ensures that the ensemble of random numbers is a reasonably good representative of the real variability, whereas the traditional random sampling (i.e. brute force) is just an ensemble of random numbers without any guarantees.

In our experiments, we start with 10 starting points (for the problem with seven variables). The range of each variable is divided into 10 equally probable intervals. In MATLAB, $X = lhsdesign(10, 7)$ generates a latin hypercube sample X containing 10

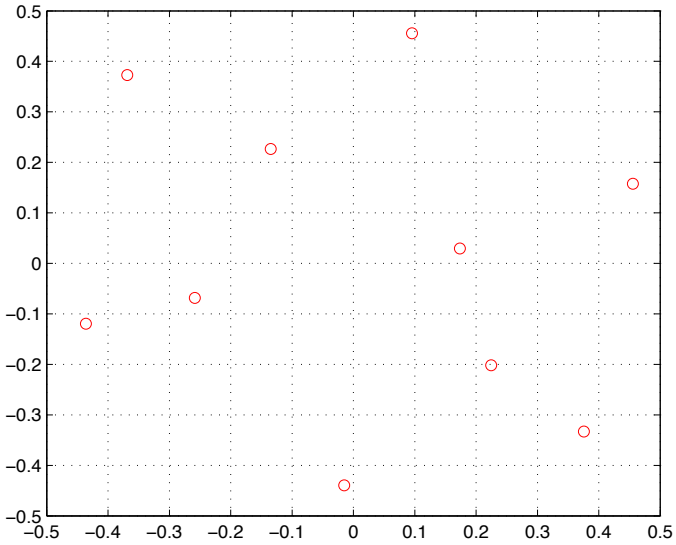


Figure 2.11: Latin Hypercube Sampling.

values of each of 7 variables. For each column, the 10 values are randomly distributed with one from each interval $(0, 1/10), (1/10, 2/10), ..., (1 - 1/10, 1)$, and they are randomly permuted. These intervals are shifted by 0.5 toward the negative axis, thus, resulting in ten sample points with each of the seven variables ranging from -0.5 to 0.5. After independently initiating DFO from these ten starting points, the solution with the largest intersection is returned.

Combined, these approaches enable us to detect cavity superpositions with a large overlapping volume without depending on atomic alignments. We refer to the combined approach as DFO-VASP. We demonstrate its capabilities below with applications to biological instances.

2.4 Computational Experiments

2.4.1 Data Set Construction

Protein Families

The serine protease and enolase superfamilies were selected for testing effectiveness of DFO-VASP in detecting binding preferences of proteins. Each superfamily contained at three subfamilies with distinct binding preferences that are achieved by well-known differences in binding site shape. Our experiments perform comparisons of the S1 subsites in serine proteasem which prefers to bind aromatic amino acids in chymotrypsins [61], basic amino acids in trypsins [40], and small hydrophobics in elastases [10]. Enolase superfamily binding sites differ because the enolase subfamily catalyzes the dehydration of 2-phospho-D-glycerate to phosphoenolpyruvate, [50], the mandelate racemase subfamily catalyzes the conversion of (R)-mandelate to and from (S)-mandelate [73], and muconate-lactonizing enzymes facilitate the reciprocal cycloisomerization of cis,cis-muconate and muconolactone [8].

Selection

This data set was also selected because each subfamily exhibits at least two sequentially non-redundant representatives. See Figure 2.12 for the number of individual structures, non-mutants and non-redundant representatives in each subfamily. This requirement ensures that the similarities discovered occur between nonidentical proteins and that differences discovered are observed in multiple examples. The protein structures used can be found in the protein data bank (PDB) [11], and are listed by the PDB code (Table 2.1). From each superfamily, we removed mutants, partially

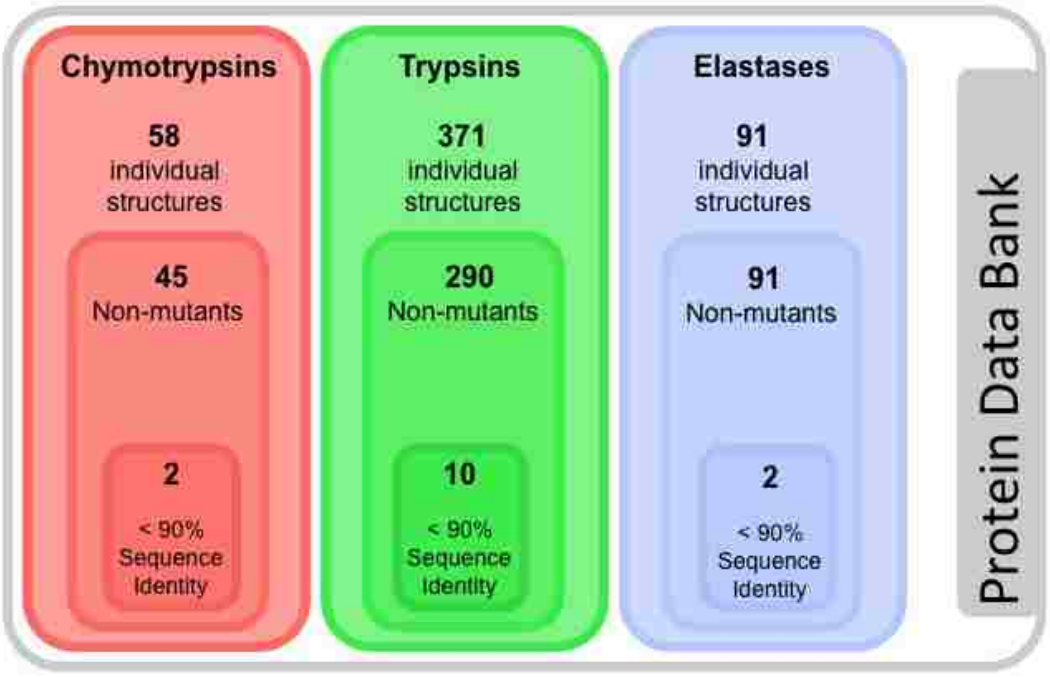


Figure 2.12: Protein data bank.

disordered structures, and structures in "closed" or otherwise inactive conformations. Structures with greater than 90% sequence identity were removed, with preference for those associated with publications, resulting in 14 serine protease and 10 enolase structures. Within these structures, ions, waters, and other non-protein atoms were removed. Since hydrogens were unavailable in all structures, all hydrogens were removed for uniformity. Atypical amino acids (e.g. selenomethionines) were not removed. Solid geometric representations of binding cavities were generated with a method described earlier [17]. These data formed the ligand binding sites compared in our study.

Electrostatic Isopotentials

We also compared electrostatic isopotentials from structures in the serine proteases. Beginning with the deprotonated structures described above, hydrogens were remod-

Table 2.1: PDB codes of structures used.

Superfamily	Subfamily	PDB Codes
Serine Protease	Trypsins	2f91, 1fn8, 2eek, 1h4w, 1bzx, 1aq7, 1ane, 1aks, 1trn, 1a0j
	Chymotrypsins	1eq9, 8gch
	Elastases	1elt, 1b0e
Enolase	Enolases	1e9i, 1iyx, 1pdy, 2pa6, 3otr, 1te6
	Mandelate Racemase	1mdr, 2ox4
	Muconate Lactonizing Enzyme	2pgw, 2zad

eled using reduce, from the MolProbity package [24]. The electrostatic potential field was computed using Delphi [76]. From the electrostatic potential field, isopotentials were generated at -10 kT/e. This threshold was selected because it is known that trypsins use a strong negative electrostatic field to select basic amino acids for binding. The negative threshold should therefore be an adequate test for evaluating if superpositions of trypsins and non-trypsins correctly exhibit these electrostatic differences.

Backbone Superposition

To compare DFO-VASP to an existing atom-based superposition method, we used Ska [86], an algorithm for whole-protein structure alignment. We superposed all pairs of serine protease structures and all pairs of enolase structures, generating an alternative superposition of all cavities. Superpositions were also generated with Dali [41] and CE [75], but since proteins in these datasets have identical folds, there were few differences.

2.4.2 Experimental Results

Validating DFO-VASP Superpositions.

DFO-VASP seeks to determine the superposition of two cavities that maximizes their overlapping volume, but this strategy does not inherently guarantee that superposing cavities from similar proteins will result in a biochemically relevant superposition. To test this hypothesis, we generated superpositions of all pairs of serine protease and all pairs of enolase cavities. Visually examining all 91 pairs of superposed serine protease cavities, we observed that all 91 cases, superposed cavities were logically oriented: Entrances to each cavity were oriented in exactly the same direction, and conserved cavity shapes were strongly superposed. An example of a superposition like this is Figure 2.13b. 33 of the 45 pairs of superposed enolase cavities were also superposed in logical orientations, with cavity entrances oriented in nearly identical directions, (e.g. Figure 2.13a). From the remaining 12, six pairs of enolase cavities were superposed with entrances at an angle of approximately 45 degrees, at an angle where ligand access to both cavities would have been difficult, and six more cavities were superposed at an angle of approximately 90 degrees, where ligand access to

Figure 2.13: Three superpositions by DFO-VASP. (a) Cavities from 1e9i (teal) and 1te6 (yellow, transparent). (b) Cavities from 1ane (teal) and 1a0j (yellow, transparent). (c) Cavities from 1e9i (teal) and 2pa6 (yellow, transparent). Black arrows indicate the entrance and direction of the cavity.

both cavities is impossible. Figure 2.13c is an example of this kind of erroneous superposition. In total, 124 out of the 136 superpositions produced cavities superposed in biochemically consistent orientations.

All superpositions observed here, however, differed in some respects from backbone superpositions. S1 cavities in serine proteases have different lengths, causing DFO-VASP to "center" smaller cavities along longer cavities. The entrance to these cavities is defined in part by backbone shape, and as a result, backbone superpositions generally superposed the cavity entrances more closely than the whole volume. Enolase cavities generally had similar depth, and the same effect did not occur.

Comparison to Backbone Superpositions.

To further evaluate DFO-VASP, we computed superpositions of each pair of cavities in both data sets. Optimal superpositions were computed using random starting positions, as described in Section 2.3.3, and also using warm-starting with backbone superposition. The volumes of intersection generated by these two methods were compared to the volume generated by backbone superposition.

Random-started superpositions and warm-started superpositions both exhibited greater volumes of superposition than backbone superpositions. This is apparent in Figure 2.14, which indicates that volumes of intersection for DFO-based superpositions is greater than backbone superpositions of the same pairs of cavities, because the differences are always greater than zero. Warm-started superpositions performed more dependably than random-started superpositions: The smallest difference between warm-started superpositions and backbone superpositions is higher than the smallest difference between random-started superpositions and backbone superpositions. However, the largest intersection volume differences between warm-started

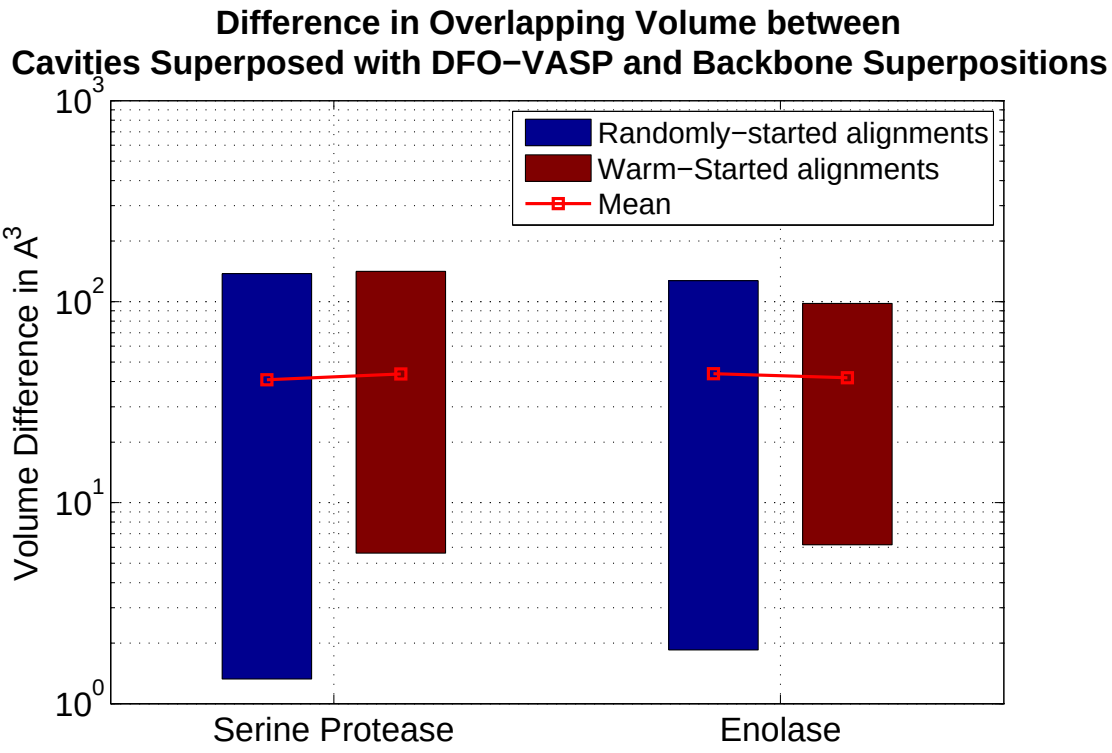


Figure 2.14: Comparison of Alignments.

superpositions and backbone superpositions were smaller or similar to the largest intersection volume differences between random-started and backbone superpositions. These suggest that random-starting may occasionally begin with unusual starting orientations that lead to a wider range of final intersection volumes.

Large-Scale Validation.

For any pair of aligned cavities, several regions inevitably exist where one cavity does not overlap the other. These regions, which we call *fragments*, are individual differences between two cavities. Cavities that are very similar exhibit small fragments, whereas cavities with large differences exhibit large fragments. Among binding cavities with similar binding preferences, we expect fragments to be very small, whereas large fragments might be expected between cavities with different binding prefer-

ences, because larger fragments could help accommodate different ligands.

To verify the predictive accuracy of our method, we constructed statistical models of fragment volumes between trypsin and enolase cavities [16]. These models estimate the probability of observing a fragment with a given volume, under the assumption that the two cavities that generated the fragment have the same binding preferences as trypsins or enolases. Using DFO-VASP, we computed all against all superpositions of serine protease cavities and enolase cavities with the latin hypercube starting strategy, and computed fragment volume for all superpositions. We used the statistical models, trained separately for trypsins and for enolases, to estimate the probability of observing all fragments. These test were performed with leave-one-out validation to avoid circular training.

Among superpositions of enolase cavities, 3236 out of the 3567 fragments generated between enolase cavities were statistically insignificant. Among the 210 superpositions of an enolase and a non-enolase cavity, 100% of the largest fragments were statistically significant. Enolase cavities superposed by DFO-VASP was volumetrically similar enough to be observed by random chance, while volumetric similarity between an enolase and a non-enolase cavity was unusual (p-value ≤ 0.05) in 210 out of 210 cases. Among superpositions of serine protease cavities, 97%, or 20424 out of the 20919 fragments generated between trypsin cavities were statistically insignificant. Among the 462 superpositions of a trypsin and a non-trypsin cavity, the largest fragment was statistically significant in 405 superpositions. Therefore, trypsin cavities superposed by DFO-VASP were volumetrically similar enough to be observed by random chance, while volumetric similarity between trypsin and non-trypsin cavities was unusual (p-value≤ 0.05) in 452 out of 462 superpositions. These results demonstrate that superpositions of ligand binding cavities with DFO-VASP can correctly identify cavities with different binding preferences.

2.5 Conclusions and Future Work

We have presented DFO-VASP for generating superpositions of ligand binding cavities by maximizing overlapping volume. It has been demonstrated [22] that a different representation of molecular shape, based on solid representations rather than atoms, can be used for generating meaningful superpositions of small molecule (ligand) binding cavities. This was achieved by optimizing the overlapping volume of two cavities, using Derivative Free Optimization (DFO) model-based method and the so-called, Volumetric Analysis of Surface Properties (VASP) software [20]. VASP is used as the black-box function, and it evaluates the volume of overlap between two cavities, based on an input superposition. Used together, DFO-VASP examine hundreds of possible superpositions in a systematic way in the search for an individual superposition with greatest overlapping volume.

Algorithmically, our method includes techniques that compensate for noisy, variable-time volume evaluations and methods for warm-starting the search for the optimum superposition. These techniques enable DFO-VASP to generate practical and accurate superpositions in a timely manner. Using VASP as a black-box function, with variable resolution, achieves tradeoffs in runtime, precision, and noise that yield unique optimization challenges. An analysis of noise in this calculation reveals that a dynamic approach to setting the resolution parameter points to reasonable tradeoffs between runtime and relative noise that are applicable for both binding cavities and electrostatic isopotentials. When comparing random-started and warm-started superpositions, we observed that final intersection volumes from random-started superpositions were more randomly distributed than warm-started superpositions, but both approaches to superposition yielded greater superposition volumes than backbone superposition. These results demonstrate that DFO-VASP is capable of gener-

ating superpositions independent of other protein structure data, creating a unique approach with significant potential applications in protein structure alignment.

From a biological point of view, we observed that derivative free optimization of the intersection volume of superposed binding cavities can be used to achieve biologically meaningful superpositions of ligand binding cavities. Visual examinations on two well-studied families of proteins (serine proteases and enolase superfamily) revealed that superposed cavities were almost always aligned in biologically relevant orientations: cavity entryways, for example, generally overlapped. We also compared the overlapping volume of cavities aligned by DFO-VASP and existing algorithms. In all cases, cavities aligned by DFO-VASP had similar or greater volumes of superposition. This result demonstrates that DFO-VASP can be a viable approach for binding site superposition, and that it exhibits novel capabilities. Finally, we assessed, at a large scale, the potential of DFO-VASP for generating superpositions of binding cavities that can be used to detect influences on binding specificity. On both data-sets, volumetric similarity were almost always unusual (p-value ≤ 0.05) between cavities with different binding preferences, and almost can always be observed by random chance between cavities with similar binding preferences.

Finally, we extended our experiments to electrostatic data where aligning the placements of charges between proteins might improve biological significance of our solutions. A second possibility with regard to electrostatic data is the simultaneous superposition of positive and negative isopotentials. Applying the same rotation and translation to both positive and negative isopotentials, it is possible that maximizing the superposition of the positive isopotential of protein A and the positive isopotential of protein B may not be as efficient as evaluating the total overlap of both the positive and negative isopotentials of protein A and protein B together. This approach has enhanced computational costs, but may also lead to superpositions with

fewer function evaluations.

Together, these results demonstrate that it is possible to align and compare ligand binding cavities when atomic similarities do not exist. More importantly, if we assume that DFO-VASP reliably identifies the optimal superposition, then a lack of cavity similarity in the optimal superposition indicates an unavoidable difference in ligand binding. This indication is a new capability unique to DFO-VASP that points to applications in protein engineering in discovering influences on ligand binding specificity.

In the future, we hope to continue utilizing VASP as a testing problem for derivative-free algorithms due to its controllable noise component. One possible future research that on can do is to further validate the usefulness of DFO-VASP by considering a new category of protein cavities that cannot be aligned with other test proteins by the backbone structure because of their fundamentally different structures. If we can find the right alignment that backbone alignments cannot find, the advantage of our method for obtaining alignment will be clearly showed.

Chapter 3

Randomized Search[1]

We propose STARS, a randomized derivative-free algorithm for unconstrained optimization when the function evaluations are contaminated with random noise. STARS takes dynamic, noise-adjusted smoothing stepsizes that minimize the least-squares error between the true directional derivative of a noisy function and its finite difference approximation. We provide a convergence rate analysis of STARS for solving convex problems with additive or multiplicative noise. Experimental results show that (1) STARS exhibits noise-invariant behavior with respect to different levels of stochastic noise; (2) the practical performance of STARS in terms of solution accuracy and convergence rate is significantly better than that indicated by the theoretical result; and (3) STARS outperforms a selection of randomized zero-order methods on both additive- and multiplicative-noisy functions.

[1]THIS CHAPTER IS BASED ON THE PAPER IN PROGRESS [23] COAUTHORED BY STEFAN M. WILD. WE ARE GRATEFUL TO KATYA SCHEINBERG FOR VALUABLE DISCUSSIONS.

3.1 Introduction

We propose STARS, a randomized derivative-free algorithm for unconstrained optimization when the function evaluations are contaminated with random noise. Formally, we address the stochastic optimization problem

$$\min_{x \in \mathbb{R}^n} f(x) = \mathbb{E}_\xi \left[\tilde{f}(x; \xi) \right], \tag{3.1}$$

where the objective $f(x)$ is assumed to be differentiable but is available only through noisy realizations $\tilde{f}(x; \xi)$. In particular, although our analysis will at times assume that the gradient of the objective function $f(x)$ exist and be Lipschitz continuous, we assume that direct evaluation of these derivatives is impossible. Of special interest to this work are situations when derivatives are unavailable or unreliable because of stochastic noise in the objective function evaluations. This type of noise introduces the dependence on the random variable ξ in (3.1) and may arise if random fluctuations or measurement errors occur in a simulation producing the objective f. In addition to stochastic and Monte Carlo simulations, this stochastic noise can also be used to model the variations in iterative or adaptive simulations resulting from finite-precision calculations and specification of internal tolerances [60].

Various methods have been designed for optimizing problems with noisy function evaluations. One such class of methods, dating back half a century, are *randomized search methods* [55]. Unlike classical, deterministic direct search methods [1, 2, 6, 53, 82, 83], randomized search methods attempt to accelerate the optimization by using random vectors as search directions. These randomized schemes share a simple basic framework, allow fast initialization, and have shown promise for solving large-scale derivative-free problems [38, 79]. Furthermore, optimization folklore and intuition suggest that these randomized steps should make the methods less

sensitive to modeling errors and "noise" in the general sense; we will systematically revisit such intuition in our computational experiments.

Recent works have addressed the special cases of zero-order minimization of convex functions with additive noise. For instance, Agarwahl et al. [3] utilize a bandit feedback model, but the regret bound depends on a term of order n^{16}. Recht et al. [44] consider a coordinate descent approach combined with an approximate line search that is robust to noise, but only theoretical bounds are provided. Moreover, the situation where the noise is nonstationary (for example, varying relative to the objective function) remains largely unstudied.

Our approach is inspired by the recent work of Nesterov [63], which established complexity bounds for convergence of random derivative-free methods for convex and nonconvex functions. Such methods work by iteratively moving along directions sampled from a normal distribution surrounding the current position. The conclusions are true for both the smooth and nonsmooth Lipschitz-continuous cases. Different improvements of these random search ideas appear in the latest literature. For instance, Stich et al. [79] give convergence rates for an algorithm where the search directions are uniformly distributed random vectors in a hypersphere and the stepsizes are determined by a line-search procedure. Incorporating the Gaussian smoothing technique of Nesterov [63], Ghadimi and Lan [38] present a randomized derivative-free method for stochastic optimization and show that the iteration complexity of their algorithm improves Nesterov's result by a factor of order n in the smooth, convex case. Although complexity bounds are readily available for these randomized algorithms, the practical usefulness of these algorithms and their potential for dealing with noisy functions have been relatively unexplored.

In this chapter, we address ways in which a randomized method can benefit from

careful choices of noise-adjusted smoothing stepsizes. We propose a new algorithm, STARS, short for STepsize Approximation in Random Search. The choice of stepsize work is greatly motivated by Moré and Wild's recent work on estimating computational noise [58] and derivatives of noisy simulations [59]. STARS takes dynamically changing smoothing stepsizes that minimize the least-squares error between the true directional derivative of a noisy function and its finite-difference approximation. We provide a convergence rate analysis of STARS for solving convex problems with both additive and multiplicative stochastic noise. With nonrestrictive assumptions about the noise, STARS enjoys a convergence rate for noisy convex functions identical to that of Nesterov's random search method for smooth convex functions.

The second contribution of our work is a numerical study of STARS. Our experimental results illustrate that (1) the performance of STARS exhibits little variability with respect to different levels of stochastic noise; (2) the practical performance of STARS in terms of solution accuracy and convergence rate is often significantly better than that indicated by the worst-case, theoretical bounds; and (3) STARS outperforms a selection of randomized zero-order methods on both additive- and multiplicative-noise problems.

The remainder of this chapter is organized as follows. In Section 3.2 we review basic assumptions about the noisy function setting and results on Gaussian smoothing. Section 3.3 presents the new STARS algorithm. In Sections 3.4 and 3.5, a convergence rate analysis is provided for solving convex problems with additive noise and multiplicative noise, respectively. Section 3.6 presents an empirical study of STARS on popular test problems by examining the performance relative to both the theoretical bounds and other randomized derivative-free solvers.

3.2 Randomized Optimization Preliminaries

One of the earliest randomized algorithms for the nonlinear, deterministic optimization problem

$$\min_{x \in \mathbb{R}^n} f(x), \tag{3.2}$$

where the objective function f is assumed to be differentiable but evaluations of the gradient ∇f are not employed by the algorithm, is attributed to Matyas [55]. Matyas introduced the random optimization approach that, at every iteration k, randomly samples a point x_+ from a Gaussian distribution centered on the current point x_k. The function is evaluated at $x_+ = x_k + u_k$, and the iterate is updated depending on whether decrease has been seen:

$$x_{k+1} = \begin{cases} x_+ & \text{if } f(x_+) < f(x_k) \\ x_k & \text{otherwise.} \end{cases}$$

Polyak [65] improved this scheme by describing stepsize rules for iterates of the form

$$x_{k+1} = x_k - h_k \frac{f(x_k + \mu_k u_k) - f(x_k)}{\mu_k} u_k, \tag{3.3}$$

where $h_k > 0$ is the stepsize, $\mu_k > 0$ is called the smoothing stepsize, and $u_k \in \mathbb{R}^n$ is a random direction.

Recently, Nesterov [63] has revived interest in Poljak-like schemes by showing that Gaussian directions $u \in \mathbb{R}^n$ allow one to benefit from properties of a Gaussian-smoothed version of the function f,

$$f_\mu(x) = \mathbb{E}_u[f(x + \mu u)], \tag{3.4}$$

where $\mu > 0$ is again the smoothing stepsize and where we have made explicit that the expectation is being taken with respect to the random vector u.

Before proceeding, we review additional notation and results concerning Gaussian smoothing.

3.2.1 Notation

We say that a function $f \in \mathcal{C}^{0,0}(\mathbb{R}^n)$ if $f : \mathbb{R}^n \mapsto \mathbb{R}$ is continuous and there exists a constant L_0 such that

$$|f(x) - f(y)| \leq L_0 \|x - y\|, \quad \forall x, y \in \mathbb{R}^n,$$

where $\|\cdot\|$ denotes the Euclidean norm. We say that $f \in \mathcal{C}^{1,1}(\mathbb{R}^n)$ if $f : \mathbb{R}^n \mapsto \mathbb{R}$ is continuously differentiable and there exists a constant L_1 such that

$$\|\nabla f(x) - \nabla f(y)\| \leq L_1 \|x - y\| \quad \forall x, y \in \mathbb{R}^n. \tag{3.5}$$

Equation (3.5) is equivalent to

$$|f(y) - f(x) - \langle \nabla f(x), y - x \rangle| \leq \frac{L_1}{2} \|x - y\|^2 \quad \forall x, y \in \mathbb{R}^n, \tag{3.6}$$

where $\langle \cdot, \cdot \rangle$ denotes the Euclidean inner product.

Similarly, if x^* is a global minimizer of $f \in \mathcal{C}^{1,1}(\mathbb{R}^n)$, then (3.6) implies that

$$\|\nabla f(x)\|^2 \leq 2L_1 (f(x) - f(x^*)) \quad \forall x \in \mathbb{R}^n. \tag{3.7}$$

We recall that a differentiable function f is convex if

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle \quad \forall x, y \in \mathbb{R}^n. \tag{3.8}$$

3.2.2 Gaussian Smoothing

We now examine properties of the Gaussian approximation of f in (3.4). For $\mu \neq 0$, we let $g_\mu(x)$ be the first-order-difference approximation of the derivative of $f(x)$ in the direction $u \in \mathbb{R}^n$,

$$g_\mu(x) = \frac{f(x + \mu u) - f(x)}{\mu} u,$$

where the nontrivial direction u is implicitly assumed. By $\nabla f_\mu(x)$ we denote the gradient (with respect to x) of the Gaussian approximation in (3.4). For standard (mean zero, covariance I_n) Gaussian random vectors u and a scalar $p \geq 0$, we define

$$M_p \equiv \mathbb{E}_u[\|u\|^p] = \frac{1}{(2\pi)^{\frac{n}{2}}} \int_{\mathbb{R}^n} \|u\|^p e^{-\frac{1}{2}\|u\|^2} du. \tag{3.9}$$

We summarize the relationships for Gaussian smoothing from [63] upon which we will rely in the following lemma.

Lemma 3.2.1. *Let $u \in \mathbb{R}^n$ be a normally distributed Gaussian vector. Then, the following are true.*

(a) For M_p defined in (3.9), we have

$$M_p \leq n^{p/2}, \quad \text{for } p \in [0, 2], \quad \text{and} \tag{3.10}$$

$$M_p \leq (n + p)^{p/2}, \quad \text{for } p > 2. \tag{3.11}$$

(b) If f is convex, then

$$f_\mu(x) \;\geq\; f(x) \quad \forall x \in \mathbb{R}^n. \tag{3.12}$$

(c) If f is convex and $f \in \mathcal{C}^{1,1}(\mathbb{R}^n)$, then

$$|f_\mu(x) - f(x)| \;\leq\; \frac{\mu^2}{2} L_1 n \quad \forall x \in \mathbb{R}^n. \tag{3.13}$$

(d) If f is differentiable at x, then

$$\mathbb{E}_u[g_\mu(x)] \;=\; \nabla f_\mu(x) \quad \forall x \in \mathbb{R}^n. \tag{3.14}$$

(e) If f is differentiable at x and $f \in \mathcal{C}^{1,1}(\mathbb{R}^n)$, then

$$\mathbb{E}_u[\|g_\mu(x)\|^2] \;\leq\; 2(n+4)\|\nabla f(x)\|^2 + \frac{\mu^2}{2} L_1^2 (n+6)^3 \quad \forall x \in \mathbb{R}^n. \tag{3.15}$$

3.3 The STARS Algorithm

The STARS algorithm for solving (3.1) while having access to the objective f only through its noisy version $\tilde{f}$ is summarized in Algorithm 3.

In general, the Gaussian directions used by Algorithm 3 can come from general Gaussian directions (e.g., with the covariance informed by knowledge about the scaling or curvature of f). For simplicity of exposition, however, we focus on standard Gaussian directions as formalized in Assumption 3.3.1. The general case can be recovered by a change of variables with an appropriate scaling of the Lipschitz constant(s).

Algorithm 3 (STARS: STep-size Approximation in Randomized Search)

1: Choose initial point x_1, iteration limit N, stepsizes $\{h_k\}_{k\geq 1}$. Evaluate the function at the initial point to obtain $\tilde{f}(x_1;\xi_0)$. Set $k \leftarrow 1$.
2: Generate a random Gaussian vector u_k, and compute the smoothing parameter μ_k.
3: Evaluate the function value $\tilde{f}(x_k + \mu_k u_k;\xi_k)$.
4: Call the stochastic gradient-free oracle

$$s_{\mu_k}(x_k;u_k,\xi_k,\xi_{k-1}) = \frac{\tilde{f}(x_k+\mu_k u_k;\xi_k) - \tilde{f}(x_k;\xi_{k-1})}{\mu_k} u_k. \tag{3.16}$$

5: Set $x_{k+1} = x_k - h_k s_{\mu_k}(x_k;u_k,\xi_k,\xi_{k-1})$.
6: Evaluate $\tilde{f}(x_{k+1};\xi_k)$, update $k \leftarrow k+1$, and return to Step 2.

Assumption 3.3.1 (Assumption about direction u)**.** *In each iteration k of Algorithm 3, u_k is a vector drawn from a multivariate normal distribution with mean* 0 *and covariance matrix I_n; equivalently, each element of u is independently and identically distributed (i.i.d.) from a standard normal distribution, $\mathcal{N}(0,1)$.*

What remains to be specified is the smoothing stepsize μ_k. It is computed by incorporating the noise information so that the approximation of the directional derivative has minimum error. We address two types of noise: *additive noise* (Section 3.4) and *multiplicative noise* (Section 3.5). These two forms of how $\tilde{f}$ depends on the random variable ξ correspond to two ways that noise often enters a system. The following sections provide near-optimal expressions for μ_k and a convergence rate analysis for both cases.

Importantly, we note Algorithm 3 allows the random variables ξ_k and ξ_{k-1} used in (3.16) to be different from one another. This generalization is in contrast to the stochastic optimization methods examined in [63], where it is assumed the same random variables are used in the smoothing calculation. This generalization does not affect the additive noise case, but will complicate the multiplicative noise case.

3.4 Additive Noise

We first consider an additive noise model for the stochastic objective function $\tilde{f}$:

$$\tilde{f}(x;\xi) = f(x) + \nu(x;\xi), \tag{3.17}$$

where $f : \mathbb{R}^n \mapsto \mathbb{R}$ is a smooth, deterministic function, $\xi \in \Xi$ is a random vector with probability distribution $P(\xi)$, and $\nu(x;\xi)$ is the stochastic noise component.

We make the following assumptions about f and ν.

Assumption 3.4.1 (Assumption about f)**.** *$f \in \mathcal{C}^{1,1}(\mathbb{R}^n)$ and f is convex.*

Assumption 3.4.2 (Assumption about additive ν)**.**

1. *For all $x \in \mathbb{R}^n$, ν is i.i.d. with bounded variance $\sigma_a^2 = Var(\nu(x;\xi)) > 0$.*

2. *For all $x \in \mathbb{R}^n$, the noise is unbiased; that is, $\mathbb{E}_\xi[\nu(x;\xi)] = 0$.*

We note that σ_a^2 is independent of x since $\nu(x;\xi)$ is identically distributed for all x. The second assumption is nonrestrictive, since if $\mathbb{E}_\xi[\nu(x;\xi)] \neq 0$, we could just redefine $f(x)$ to be $f(x) - \mathbb{E}_\xi[\nu(x;\xi)]$.

3.4.1 Noise and Finite Differences

Moré and Wild [59] introduce a way of computing the smoothing stepsize μ that mitigates the effects of the noise in $\tilde{f}$ when estimating a first-order directional directive. The method involves analyzing the expectation of the least-squared error between the forward-difference approximation, $\dfrac{\tilde{f}(x+\mu u;\xi_1) - \tilde{f}(x;\xi_2)}{\mu}$, and the directional derivative of the smooth function, $\langle \nabla f(x), u \rangle$. The authors show that a

near-optimal μ can be computed in such a way that the expected error has the tightest upper bound among all such values μ. Inspired by their approach, we consider the least-square error between $\dfrac{\tilde{f}(x+\mu u;\xi_1)-\tilde{f}(x;\xi_2)}{\mu}u$ and $\langle\nabla f(x),u\rangle u$. That is, our goal is to find μ^* that minimizes an upper bound on $\mathbb{E}[\mathcal{E}(\mu)]$, where

$$\mathcal{E}(\mu)\equiv\mathcal{E}(\mu;x,u,\xi_1,\xi_2)=\left\|\frac{\tilde{f}(x+\mu u;\xi_1)-\tilde{f}(x;\xi_2)}{\mu}u-\langle\nabla f(x),u\rangle u\right\|^2.$$

We recall that u, ξ_1, and ξ_2 are independent random variables.

Theorem 3.4.3. *Let Assumptions 3.3.1, 3.4.1, and 3.4.2 hold. If a smoothing step-size is chosen as*

$$\mu^*=\left[\frac{8\sigma_a^2 n}{L_1^2(n+6)^3}\right]^{\frac{1}{4}}, \tag{3.18}$$

then for any $x\in\mathbb{R}^n$, we have

$$\mathbb{E}_{u,\xi_1,\xi_2}[\mathcal{E}(\mu^*)]\leq\sqrt{2}L_1\sigma_a\sqrt{n(n+6)^3}. \tag{3.19}$$

Proof. Using (3.17) and (3.6), we derive

$$\begin{aligned}\mathcal{E}(\mu) &\leq \left\|\frac{\nu(x+\mu u;\xi_1)-\nu(x;\xi_2)}{\mu}u+\frac{\mu L_1}{2}\|u\|^2u\right\|^2\\ &\leq \left(\frac{\nu(x+\mu u;\xi_1)-\nu(x;\xi_2)}{\mu}+\frac{\mu L_1}{2}\|u\|^2\right)^2\|u\|^2.\end{aligned}$$

Let $X=\dfrac{\nu(x+\mu u;\xi_1)-\nu(x;\xi_2)}{\mu}+\dfrac{\mu L_1}{2}\|u\|^2$. By Assumption 3.4.2, the expectation of X with respect to ξ_1 and ξ_2 is $\mathbb{E}_{\xi_1,\xi_2}[X]=\dfrac{\mu L_1}{2}\|u\|^2$, and the corresponding variance is $\mathrm{Var}(X)=\dfrac{2\sigma_a^2}{\mu^2}$. It then follows that

$$\mathbb{E}_{\xi_1,\xi_2}[X^2]=(\mathbb{E}_{\xi_1,\xi_2}[X])^2+\mathrm{Var}(X)=\frac{\mu^2L_1^2}{4}\|u\|^4+\frac{2\sigma_a^2}{\mu^2}.$$

Hence, taking the expectation of $\mathcal{E}(\mu)$ with respect to u, ξ_1, and ξ_2 yields

$$\begin{aligned}\mathbb{E}_{u,\xi_1,\xi_2}[\mathcal{E}(\mu)] &\leq \mathbb{E}_u\left[\mathbb{E}_{\xi_1,\xi_2}[X^2\|u\|^2]\right] \\ &= \mathbb{E}_u\left[\frac{\mu^2 L_1^2}{4}\|u\|^6 + \frac{2\sigma_a^2}{\mu^2}\|u\|^2\right].\end{aligned}$$

Using (3.10) and (3.11), we can further derive

$$\mathbb{E}_{u,\xi_1,\xi_2}[\mathcal{E}(\mu)] \leq \frac{\mu^2 L_1^2}{4}(n+6)^3 + \frac{2\sigma_a^2}{\mu^2}n. \tag{3.20}$$

The right-hand side of (3.20) is uniformly convex in μ and has a global minimizer of

$$\mu^* = \left[\frac{8\sigma_a^2 n}{L_1^2(n+6)^3}\right]^{\frac{1}{4}},$$

with the corresponding minimum value yielding (3.19). □

Remarks:

- A key observation is that for a function $\tilde{f}(x;\xi)$ with additive noise, as long as the noise has a constant variance $\sigma_a > 0$, the optimal choice of the stepsize μ^* is independent of x.
- Since the proof of Theorem 3.4.3 does not rely on the convexity assumption about f, the error bound (3.19) for the finite-difference approximation also holds for the nonconvex case. The convergence rate analysis for STARS presented in the next section, however, will assume convexity of f; the nonconvex case is out of the scope of this chapter but is of interest for future research.

3.4.2 Convergence Rate Analysis

We now examine the convergence rate of Algorithm 3 applied to the additive noise case of (3.17) and with $\mu_k = \mu^*$ for all k. One of the main ideas behind this convergence proof relies on the fact that we can derive the improvement in f achieved by each step in terms of the change in x. Since the distance between the starting point and the optimal solution, denoted by $R = \|x_0 - x^*\|$, is finite, one can derive an upper bound for the "accumulative improvement in f," $\frac{1}{N+1}\sum_{k=0}^{N}(\mathbb{E}[f(x_k)] - f^*)$. Hence, we can show that increasing the number of iterations, N, of Algorithm 3 yields higher accuracy in the solution.

For simplicity, we denote by $\mathbb{E}[\cdot]$ the expectation over all random variables (i.e., $\mathbb{E}[\cdot] = \mathbb{E}_{u_k,\dots,u_1,\xi_k,\dots,\xi_0}[\cdot]$), unless otherwise specified. Similarly, we denote $s_{\mu_k}(x_k; u_k, \xi_k, \xi_{k-1})$ in (3.16) by s_{μ_k}. The following lemma directly follows from Theorem 3.4.3.

Lemma 3.4.4. *Let Assumptions 3.3.1, 3.4.1, and 3.4.2 hold. If the smoothing step-size μ_k is set to the constant μ^* from (3.18), then Algorithm 3 generates steps satisfying*

$$\mathbb{E}[\|s_{\mu_k}\|^2] \leq 2(n+4)\|\nabla f(x_k)\|^2 + C_2,$$

where $C_2 = 2\sqrt{2}L_1\sigma_a\sqrt{n(n+6)^3}$.

Proof. Let $g_0(x_k) = \langle \nabla f(x_k), u_k\rangle u_k$. Then (3.19) implies that

$$\mathbb{E}[\|s_{\mu_k}\|^2 - 2\langle s_{\mu_k}, g_0(x_k)\rangle + \|g_0(x_k)\|^2] \leq C_1, \tag{3.21}$$

where $C_1 = \sqrt{2}L_1\sigma_a\sqrt{n(n+6)^3}$.

The stochastic gradient-free oracle s_{μ_k} in (3.16) is a random approximation of the

gradient $\nabla f(x_k)$. Furthermore, the expectation of s_{μ_k} with respect to ξ_k and ξ_{k-1} yields the forward-difference approximation of the derivative of f in the direction u_k at x_k:

$$\mathbb{E}_{\xi_k,\xi_{k-1}}[s_{\mu_k}] = \frac{f(x_k + \mu_k u_k) - f(x_k)}{\mu_k} u_k = g_\mu(x_k). \tag{3.22}$$

Combining (3.21) and (3.22) yields

$$\begin{aligned}
\mathbb{E}\left[\|s_{\mu_k}\|^2\right] &\leq \mathbb{E}[2\langle s_{\mu_k}, g_0(x_k)\rangle - \|g_0(x_k)\|^2] + C_1 \\
&\overset{(3.22)}{=} \mathbb{E}_{u_k}[2\langle g_\mu(x_k), g_0(x_k)\rangle - \|g_0(x_k)\|^2] + C_1 \\
&= \mathbb{E}_{u_k}[-\|g_0(x_k) - g_\mu(x_k)\|^2 + \|g_\mu(x_k)\|^2] + C_1 \\
&\leq \mathbb{E}_{u_k}[\|g_\mu(x_k)\|^2] + C_1 \\
&\overset{(3.15)}{\leq} 2(n+4)\|\nabla f(x_k)\| + C_2,
\end{aligned}$$

where $C_2 = C_1 + \dfrac{\mu_k^2}{2} L_1^2 (n+6)^3 = 2\sqrt{2} L_1 \sigma_a \sqrt{n(n+6)^3}$. □

We are now ready to show convergence of the algorithm. Denote $x^* \in \mathbb{R}^n$ a minimizer associated with $f^* = f(x^*)$. Denote by $\mathcal{U}_k = \{u_1, \cdots, u_k\}$ the set of i.i.d. random variable realizations attached to each iteration of Algorithm 1. Similarly, let $\mathcal{P}_k = \{\xi_0, \cdots, \xi_k\}$. Define $\phi_0 = f(x_0)$ and $\phi_k = \mathbb{E}_{\mathcal{U}_{k-1},\mathcal{P}_{k-1}}[f(x_k)]$ for $k \geq 1$.

Theorem 3.4.5. *Let Assumptions 3.3.1, 3.4.1, and 3.4.2 hold. Let the sequence $\{x_k\}_{k\geq 0}$ be generated by Algorithm 1 with the smoothing stepsize μ_k set as μ^* in (3.18). If the fixed step length is $h_k = h = \dfrac{1}{4L_1(n+4)}$ for all k, then for any $N \geq 0$, we have*

$$\frac{1}{N+1}\sum_{k=0}^{N}(\phi_k - f^*) \leq \frac{4L_1(n+4)}{N+1}\|x_0 - x^*\|^2 + \frac{3\sqrt{2}}{5}\sigma_a(n+4).$$

Proof. We start with deriving the expectation of the change in x of each step, that is, $\mathbb{E}[r_{k+1}^2] - r_k^2$, where $r_k = \|x_k - x^*\|$. First,

$$\begin{aligned} r_{k+1}^2 &= \|x_k - h_k s_{\mu_k} - x^*\|^2 \\ &= r_k^2 - 2h_k\langle s_{\mu_k}, x_k - x^*\rangle + h_k^2\|s_{\mu_k}\|^2. \end{aligned}$$

$\mathbb{E}[s_{\mu_k}]$ can be derived by using (3.14) and (3.22). $\mathbb{E}[\|s_{\mu_k}\|^2]$ is derived in Lemma 3.4.4. Hence,

$$\mathbb{E}\left[r_{k+1}^2\right] \leq r_k^2 - 2h_k\langle \nabla f_\mu(x_k), x_k - x^*\rangle + h_k^2[2(n+4)\|\nabla f(x_k)\|^2 + C_2].$$

By using (3.8), (3.12), and (3.7), we derive

$$\mathbb{E}\left[r_{k+1}^2\right] \leq r_k^2 - 2h_k(f(x_k) - f_\mu(x^*)) + 4h_k^2 L_1(n+4)(f(x_k) - f(x^*)) + h_k^2 C_2.$$

Combining this expression with (3.13), which bounds the error between $f_\mu(x)$ and $f(x)$, we obtain

$$\mathbb{E}\left[r_{k+1}^2\right] \leq r_k^2 - 2h_k(1 - 2h_k L_1(n+4))(f(x_k) - f^*) + C_3,$$

where $C_3 = h_k^2 C_2 + 2h_k\dfrac{\mu_k^2}{2}L_1 n = h_k^2 C_2 + 2\sqrt{2}h_k\sigma_a\sqrt{\dfrac{n^3}{(n+6^3)}}$.

Let $h_k = h = \dfrac{1}{4L_1(n+4)}$. Then,

$$\mathbb{E}\left[r_{k+1}^2\right] \leq r_k^2 - \frac{f(x_k) - f^*}{4L_1(n+4)} + C_3, \tag{3.23}$$

where $C_3 = \dfrac{\sqrt{2}\sigma_a}{2L_1}g_1(n)$ and $g_1(n) = \dfrac{\sqrt{n(n+6)^3}}{4(n+4)^2} + \dfrac{1}{n+4}\sqrt{\dfrac{n^3}{(n+6)^3}}$. By showing that $g_1'(n) < 0$ for all $n \geq 10$ and $g_1'(n) > 0$ for all $n \leq 9$, we can prove that

$g_1(n) \leq \max\{g(9), g(10)\} = \max\{0.2936, 0.2934\} \leq 0.3$. Hence, $C_3 \leq \dfrac{3\sqrt{2}\sigma_a}{20L_1}$.

Taking the expectation in $\mathcal{U}_k$ and $\mathcal{P}_k$, we have

$$\mathbb{E}_{\mathcal{U}_k,\mathcal{P}_k}[r_{k+1}^2] \leq \mathbb{E}_{\mathcal{U}_{k-1},\mathcal{P}_{k-1}}[r_k^2] - \frac{\phi_k - f^*}{4L_1(n+4)} + \frac{3\sqrt{2}\sigma_a}{20L_1}.$$

Summing these inequalities over $k = 0, \cdots, N$ and dividing by $N+1$, we obtain the desired result. □

The bound in Theorem 3.4.5 is valid also for $\hat{\phi}_N = \mathbb{E}_{\mathcal{U}_{k-1},\mathcal{P}_{k-1}}[f(\hat{x}_N)]$, where $\hat{x}_N = \arg\min_x\{f(x) : x \in \{x_0, \cdots, x_N\}\}$. In this case,

$$\begin{aligned}\mathbb{E}_{\mathcal{U}_{k-1},\mathcal{P}_{k-1}}[f(\hat{x}_N)] - f^* &\leq \mathbb{E}_{\mathcal{U}_{k-1},\mathcal{P}_{k-1}}\left[\frac{1}{N+1}\sum_{k=0}^{N}(\phi_k - f^*)\right] \\ &\leq \frac{4L_1(n+4)}{N+1}\|x_0 - x^*\|^2 + \frac{3\sqrt{2}}{5}\sigma_a(n+4).\end{aligned}$$

Hence, in order to achieve a final accuracy of ϵ for $\hat{\phi}_N$ (that is, $\hat{\phi}_N - f^* \leq \epsilon$), the allowable absolute noise in the objective function has to satisfy $\sigma_a \leq \dfrac{5\epsilon}{6\sqrt{2}(n+4)}$. Furthermore, under this bound on the allowable noise, this ϵ accuracy can be ensured by STARS in

$$N = \frac{8(n+4)L_1R^2}{\epsilon} - 1 \sim \mathcal{O}\left(\frac{n}{\epsilon}L_1R^2\right) \tag{3.24}$$

iterations, where R^2 is an upper bound on the squared Euclidean distance between the starting point and the optimal solution: $\|x_0 - x^*\|^2 \leq R^2$. In other words, given an optimization problem that has bounded absolute noise of variance σ_a^2, the best

accuracy that can be ensured by STARS is

$$\epsilon_{\text{pred}} \geq \frac{6\sqrt{2}\sigma_a(n+4)}{5}, \tag{3.25}$$

and we can solve this noisy problem in $\mathcal{O}\left(\frac{n}{\epsilon_{\text{pred}}} L_1 R^2\right)$ iterations. Unsurprisingly, a price must be paid for having access only to noisy realizations, and this price is that arbitrary accuracy cannot be reached in the noisy setting.

3.5 Multiplicative Noise

A multiplicative noise model is described by

$$\tilde{f}(x;\xi) = f(x)[1+\nu(x;\xi)] = f(x) + f(x)\nu(x;\xi). \tag{3.26}$$

In practice, $|\nu|$ is bounded by something smaller (often much smaller) than 1. A canonical example is when f corresponds to a Monte Carlo integration, with the a stopping criterion based on the value $f(x)$. Similarly, if f is simple and computed in double precision, the relative errors are roughly 10^{-16}; in single precision, the errors are roughly 10^{-8} and in half precision we get errors of roughly 10^{-4}.

Formally, we make the following assumptions in our analysis of STARS for the problem (3.1) with multiplicative noise.

Assumption 3.5.1 (Assumption about f)**.** *f is continuously differentiable and convex and has Lipschitz constant L_0. ∇f has Lipschitz constant L_1.*

Assumption 3.5.2 (Assumption about multiplicative ν)**.**

1. *ν is i.i.d., with zero mean and bounded variance; that is, $\mathbb{E}[\nu] = 0$, $\sigma_r^2 = Var(\nu) > 0$.*

2. *The expectation of the signal-to-noise ratio is bounded; that is,* $\mathbb{E}[\frac{1}{1+\nu}] \leq b$.

3. *The support of* ν *(i.e., the range of values that* ν *can take with positive probability) is bounded by* $\pm a$*, where* $a < 1$.

The first part of Assumption 3.5.2 is analogous to that in Assumption 3.4.2 and guarantees that the distribution of ν is independent of x. Although not specifying a distributional form for ν (with respect to ξ), the final two parts of Assumption 3.5.2 are made to simplify the presentation and rule out cases where the noise completely corrupts the function.

3.5.1 Noise and Finite Differences

Analogous to Theorem 3.4.3, Theorem 3.5.3 shows how to compute the near-optimal stepsizes in the multiplicative noise setting.

Theorem 3.5.3. *Let Assumptions 3.5.1 and 3.5.2 hold. If a forward-difference parameter is chosen as*

$$\mu^* = C_4\sqrt{|f(x)|}, \quad \text{where } C_4 = \left[\frac{16\sigma_r^2 n}{L_1^2(1+3\sigma_r^2)(n+6)^3}\right]^{\frac{1}{4}},$$

then for any $x \in \mathbb{R}^n$ *we have*

$$\mathbb{E}_{u,\xi_1,\xi_2}[\mathcal{E}(\mu^*)] \leq 2L_1\sigma_r\sqrt{(1+3\sigma_r^2)n(n+6)^3}|f(x)| + 3L_0^2\sigma_r^2(n+4)^2. \tag{3.27}$$

Proof. By using (3.26) and (3.6), we derive

$$\begin{aligned}\mathcal{E}(\mu) &\leq \left\| \frac{f(x+\mu u)\nu(x+\mu u;\xi_1) - f(x)\nu(x;\xi_2)}{\mu} u + \frac{\mu L_1}{2}\|u\|^2 u \right\|^2 \\ &\leq \left(\frac{f(x+\mu u)\nu(x+\mu u;\xi_1) - f(x)\nu(x;\xi_2)}{\mu} + \frac{\mu L_1}{2}\|u\|^2 \right)^2 \|u\|^2.\end{aligned}$$

Again applying (3.6), we get $\mathcal{E}(\mu) \leq X^2\|u\|^2$, where

$$\begin{aligned}X &= \frac{f(x+\mu u)\nu(x+\mu u;\xi_1) - f(x)\nu(x;\xi_2)}{\mu} + \frac{\mu L_1}{2}\|u\|^2 \\ &\leq \left(\frac{f(x)}{\mu} + \nabla f(x)^T u + \frac{\mu L_1}{2}\|u\|^2 \right) \nu(x+\mu u;\xi_1) - \frac{f(x)}{\mu}\nu(x;\xi_2) + \frac{\mu L_1}{2}\|u\|^2.\end{aligned}$$

The expectation of X with respect to ξ_1 and ξ_2 is

$$E_{\xi_1,\xi_2}[X] = \frac{\mu L_1}{2}\|u\|^2$$

and the corresponding variance is

$$\begin{aligned}\mathrm{Var}(X) &= \left(\frac{f(x)}{\mu} + \nabla f(x)^T u + \frac{\mu L_1}{2}\|u\|^2 \right)^2 \sigma_r^2 + \frac{f^2(x)}{\mu^2}\sigma_r^2 \\ &\leq \left(\frac{3f^2(x)}{\mu^2} + 3(\nabla f(x)^T u)^2 + \frac{3\mu^2 L_1^2}{4}\|u\|^4 \right) \sigma_r^2 + \frac{f^2(x)}{\mu^2}\sigma_r^2 \\ &= \left(\frac{4f^2(x)}{\mu^2} + 3(\nabla f(x)^T u)^2 + \frac{3\mu^2 L_1^2}{4}\|u\|^4 \right) \sigma_r^2,\end{aligned}$$

where the inequality holds because $(a+b+c)^2 \leq 3a^2+3b^2+3c^2$ for any a, b, c. Since $\mathbb{E}[X^2] = \mathrm{Var}(X) + (\mathbb{E}[X])^2$, we have that

$$\begin{aligned}\mathbb{E}_{\xi_1,\xi_2}[X^2] &\leq \frac{\mu^2 L_1^2(1+3\sigma_r^2)}{4}\|u\|^4 + \frac{4\sigma_r^2}{\mu^2}f^2(x) + 3(\nabla f(x)^T u)^2\sigma_r^2 \\ &\leq \frac{\mu^2 L_1^2(1+3\sigma_r^2)}{4}\|u\|^4 + \frac{4\sigma_r^2}{\mu^2}f^2(x) + 3L_0^2\sigma_r^2\|u\|^2.\end{aligned}$$

Hence, we can derive

$$\begin{aligned}\mathbb{E}[\mathcal{E}(\mu)] &\leq \mathbb{E}_u[\mathbb{E}_{\xi_1,\xi_2}[X^2\|u\|^2]] \\ &= \mathbb{E}_u[\|u\|^2\mathbb{E}_{\xi_1,\xi_2}[X^2]] \\ &\leq \mathbb{E}_u\left[\frac{\mu^2 L_1^2(1+3\sigma_r^2)}{4}\|u\|^6 + \frac{4\sigma_r^2}{\mu^2}f^2(x)\|u\|^2 + 3L_0^2\sigma_r^2\|u\|^4\right].\end{aligned}$$

By using (3.10), (3.11), and this last expression, we get

$$\mathbb{E}[\mathcal{E}(\mu)] \leq \frac{\mu^2 L_1^2(1+3\sigma_r^2)}{4}(n+6)^3 + \frac{4\sigma_r^2 n}{\mu^2}f^2(x) + 3L_0^2\sigma_r^2(n+4)^2.$$

The right-hand side of this expression is uniformly convex in μ and attains its global minimum at $\mu^* = C_4\sqrt{|f(x)|}$; the corresponding expectation of the least-squares error is

$$\mathbb{E}_{u,\xi_1,\xi_2}[\mathcal{E}(\mu^*)] \leq 2L_1\sigma_r\sqrt{(1+3\sigma_r^2)n(n+6)^3}|f(x)| + 3L_0^2\sigma_r^2(n+4)^2.$$

□

Unlike for the absolute noise case of Section 3.4, the optimal μ value in Theorem 3.5.3 is not independent of x. Furthermore, letting $\mu_k = \mu^* = C_4\sqrt{|f(x)|}$ assumes that f is known. Unfortunately, we have access to f only through $\tilde{f}$. However, we can compute an estimate, $\tilde{\mu}$, of μ^* by substituting f with $\tilde{f}$ and still derive an error bound. To simplify the derivations, we introduce another random variable, ξ_3, independent of ξ_1 and ξ_2, to compute $\tilde{\mu} \equiv \tilde{\mu}(x;\xi_3)$. The goal is to obtain an upper

bound on $\mathbb{E}_{\xi_3}[\mathbb{E}_{\xi_1,\xi_2,u}[\mathcal{E}(\tilde{\mu})]]$, where

$$\mathcal{E}(\tilde{\mu}) \equiv \mathcal{E}(\tilde{\mu}, x; u, \xi_1, \xi_2, \xi_3) = \left\| \frac{\tilde{f}(x + \tilde{\mu}; \xi_1) - \tilde{f}(x; \xi_2)}{\tilde{\mu}} u - \langle \nabla f(x), u \rangle u \right\|^2 .$$

This then allows us to proceed with the usual derivations while requiring only an additional expectation over ξ_3.

Lemma 3.5.4. *Let Assumptions 3.5.1 and 3.5.2 hold. If a forward-difference parameter is chosen as*

$$\tilde{\mu} = C_4 \sqrt{|\tilde{f}(x; \xi_3)|}, \quad \textit{where } C_4 = \left[\frac{16\sigma_r^2 n}{L_1^2(1 + 3\sigma_r^2)(n+6)^3} \right]^{\frac{1}{4}}, \tag{3.28}$$

then for any $x \in \mathbb{R}^n$, we have

$$\mathbb{E}_{u,\xi_1,\xi_2,\xi_3}[\mathcal{E}(\tilde{\mu})] \leq (1+b) L_1 \sigma_r \sqrt{(1 + 3\sigma_r^2) n (n+6)^3} |f(x)| + 3L_0^2 \sigma_r^2 (n+4)^2. \tag{3.29}$$

Proof.

$$\begin{aligned}
\mathbb{E}[\mathcal{E}(\tilde{\mu})] &= \mathbb{E}_{\xi_3}[\mathbb{E}_{u,\xi_1,\xi_2}[\mathcal{E}(\tilde{\mu})]] \\
&\leq \mathbb{E}_{\xi_3}\left[\frac{\tilde{\mu}^2 L_1^2 (1 + 3\sigma_r^2)}{4} (n+6)^3 + \frac{4\sigma_r^2 n}{\tilde{\mu}^2} f^2(x) + 3L_0^2 \sigma_r^2 (n+4)^2 \right] \\
&= L_1 \sigma_r \sqrt{(1 + 3\sigma_r^2) n (n+6)^3} |f(x)| \mathbb{E}_{\xi_3}\left[1 + \nu(x; \xi_3) + \frac{1}{1 + \nu(x; \xi_3)} \right] + 3L_0^2 \sigma_r^2 (n+4)^2 \\
&\leq (1+b) L_1 \sigma_r \sqrt{(1 + 3\sigma_r^2) n (n+6)^3} |f(x)| + 3L_0^2 \sigma_r^2 (n+4)^2,
\end{aligned}$$

where the last inequality holds by Assumption 3.5.2 because the expectation of the signal-to-noise ratio is bounded by b. □

Remark: Similar to the additive noise case, Theorem 3.5.3 and Theorem 3.5.4 do not require f to be convex. Hence, (3.27) and (3.29) both hold in the nonconvex

case. However, the following convergence rate analysis applies only to the convex case, since Lemma 3.5.6 relies on a convexity assumption for f.

3.5.2 Convergence Rate Analysis

Let $\mu_k = \tilde{\mu} = C_4\sqrt{|\tilde{f}(x_k;\xi_{k'})|}$ in Algorithm 3. Before showing the convergence result, we derive $\mathbb{E}[\langle s_{\tilde{\mu}}, x_k - x^*\rangle]$ and $\mathbb{E}[\|s_{\tilde{\mu}}\|^2]$, where $s_{\tilde{\mu}}$ denotes $s_\mu(x_k; u_k, \xi_k, \xi_{k-1}, \xi_{k'})$ and $\mathbb{E}[\cdot]$ denotes the expectation over all random variables u_k, ξ_k, ξ_{k-1}, and $\xi_{k'}$(i.e., $\mathbb{E}[\cdot] = \mathbb{E}_{u_k,\xi_k,\xi_{k-1},\xi_{k'}}[\cdot]$), unless otherwise specified.

Lemma 3.5.5. *Let Assumptions 3.5.1 and 3.5.2 hold. If* $\mu_k = \tilde{\mu} = C_4\sqrt{|\tilde{f}(x_k;\xi_{k'})|}$, *then*

$$\mathbb{E}[\|s_{\tilde{\mu}}\|^2] \leq 2(n+4)\|\nabla f(x_k)\|^2 + C_5|f(x_k)| + C_6,$$

where $C_5 = \frac{1}{2}C_4^2L_1^2(n+6)^3 + (1+b)L_1\sigma_r\sqrt{(1+3\sigma_r^2)n(n+6)^3}$ *and* $C_6 = 3L_0^2\sigma_r^2(n+4)^2$.

Proof. Let $g_0(x_k) = \langle \nabla f(x_k), u_k\rangle u_k$. The bound (3.28) in Theorem 3.5.4 implies that

$$\mathbb{E}[\|s_{\tilde{\mu}} - g_0(x_k)\|^2] \;\leq\; (1+b)L_1\sigma_r\sqrt{(1+3\sigma_r^2)n(n+6)^3}|f(x)| + 3L_0^2\sigma_r^2(n+4)^2 \equiv \ell(x).$$

Hence,

$$\begin{aligned}
& \mathbb{E}\left[\|s_{\tilde{\mu}}\|^2\right] \\
\leq\ & \mathbb{E}_{\xi_{k'}}\left[\mathbb{E}_{u_k,\xi_k,\xi_{k-1}}[2\langle s_\mu, g_0(x_k)\rangle - \|g_0(x_k)\|^2]\right] + \ell(x) \\
\overset{(3.22)}{=}\ & \mathbb{E}_{\xi_{k'}}\left[\mathbb{E}_{u_k}[2\langle g_{\mu_k}(x_k), g_0(x_k)\rangle - \|g_0(x_k)\|^2]\right] + \ell(x) \\
\leq\ & \mathbb{E}_{\xi_{k'}}\left[\mathbb{E}_{u_k}[\|g_{\mu_k}(x_k)\|^2]\right] + \ell(x) \\
\overset{(3.15)}{\leq}\ & 2(n+4)\|\nabla f(x_k)\|^2 + \mathbb{E}_{\xi_{k'}}\left[\frac{\mu_k^2}{2}L_1^2(n+6^3)\right] + \ell(x) \\
=\ & 2(n+4)\|\nabla f(x_k)\|^2 + C_5|f(x_k)| + C_6,
\end{aligned}$$

where the last equality holds since $\mathbb{E}_{\xi_{k'}}[\mu_k^2] = \mathbb{E}_{\xi_{k'}}[C_4^2|f(x_k)|(1+\nu(x_k;\xi_{k'})] = C_4^2|f(x_k)|$.

□

Lemma 3.5.6. *Let Assumptions 3.5.1 and 3.5.2 hold. If* $\mu_k = \tilde{\mu} = C_4\sqrt{|\tilde{f}(x_k;\xi_{k'})|}$, *then*

$$\mathbb{E}[\langle s_{\tilde{\mu}}, x_k - x^*\rangle] \geq f(x_k) - f^* - \frac{C_4^2 L_1 n}{2}|f(x_k)|.$$

Proof. First, we have

$$\begin{aligned}
\mathbb{E}_{u_k,\xi_k,\xi_{k-1}}[s_{\tilde{\mu}}] &= \mathbb{E}_{u_k,\xi_k,\xi_{k-1}}\left[\frac{\tilde{f}(x_k+\mu_k u_k;\xi_k) - \tilde{f}(x_k;\xi_{k-1})}{\mu_k}u_k\right] \\
&= \mathbb{E}_{u_k,\xi_k,\xi_{k-1}}\left[\frac{f(x_k+\mu_k u_k)[1+\nu(x_k+\mu_k u_k;\xi_k)] - f(x_k)[1+\nu(x_k;\xi_{k-1})]}{\mu_k}u_k\right] \\
&= \mathbb{E}_{u_k}\left[\frac{f(x_k+\mu_k u_k) - f(x_k)}{\mu_k}u_k\right] \\
&= \mathbb{E}_{u_k}[g_{\mu_k}(x_k)] \\
&\overset{(3.14)}{=} \nabla f_{\mu_k}(x_k).
\end{aligned}$$

Then, we get

$$\begin{aligned}\mathbb{E}_{u_k,\xi_k,\xi_{k-1}}[\langle s_{\tilde{\mu}}, x_k - x^*\rangle] &= \langle \nabla f_{\mu_k}(x_k), x_k - x^*\rangle \\ &\overset{(3.8)}{\geq} f_{\mu_k}(x_k) - f_{\mu_k}(x^*) \\ &\overset{(3.12)}{\geq} f(x_k) - f_{\mu_k}(x^*) \\ &\overset{(3.13)}{\geq} f(x_k) - f^* - \frac{\mu_k}{2} L_1 n.\end{aligned}$$

Since $\mu_k = \tilde{\mu} = C_4\sqrt{|\tilde{f}(x_k;\xi_{k'})|}$, we have

$$\mathbb{E}[\langle s_{\tilde{\mu}}, x_k - x^*\rangle] = \mathbb{E}_{\xi_{k'}}[\mathbb{E}_{u_k,\xi_k,\xi_{k-1}}[\langle s_{\tilde{\mu}}, x_k - x^*\rangle]] \geq f(x_k) - f^* - \frac{C_4^2 L_1 n}{2}|f(x_k)|.$$

□

We are now ready to show the convergence of Algorithm 3, with $\mu_k = \tilde{\mu}$, for the minimization of a function (3.26) with bounded multiplicative noise.

Theorem 3.5.7. *Let Assumptions 3.5.1 and 3.5.2 hold. Let the sequence $\{x_k\}_{k\geq 0}$ be generated by Algorithm 3 with the smoothing parameter μ_k being*

$$\mu_k = \tilde{\mu} = C_4\sqrt{|\tilde{f}(x;\xi_{k'})|}$$

and the fixed step length set to $h_k = h = \dfrac{1}{4L_1(n+4)}$ for all k. Let M be an upper bound on the average of the historical absolute values of noise-free function evaluations; that is,

$$M \geq \frac{1}{N+1}\sum_{k=0}^{N}|\phi_k| = \frac{1}{N+1}\left(|f(x_0)| + \sum_{k=1}^{N}\mathbb{E}_{\mathcal{U}_{k-1},\mathcal{P}_{k-1}}\left[|f(x_k)|\right]\right).$$

Then, for any $N \geq 0$ we have

$$\frac{1}{N+1}\sum_{k=0}^{N}(\phi_k - f^*) \leq \frac{4L_1(n+4)}{N+1}\|x_0 - x^*\|^2 + 4L_1(n+4)\left(C_7 M + C_8\right), \quad (3.30)$$

where $C_7 = \dfrac{C_4^2 n}{4(n+4)} + \dfrac{C_5}{16L_1^2(n+4)^2}$ *and* $C_8 = \dfrac{C_6}{16L_1^2(n+4)^2}$.

Proof. Let $r_k = \|x_k - x^*\|$. First,

$$\begin{aligned} r_{k+1}^2 &= \|x_k - h_k s_{\tilde{\mu}} - x^*\|^2 \\ &= r_k^2 - 2h_k\langle s_{\tilde{\mu}}, x_k - x^*\rangle + h_k^2\|s_{\tilde{\mu}}\|^2. \end{aligned}$$

$\mathbb{E}[\langle s_{\tilde{\mu}}, x_k - x^*\rangle]$ and $\mathbb{E}[\|s_{\tilde{\mu}}\|^2]$ are derived in Lemma 3.5.6 and Lemma 3.5.5, respectively. Hence, incorporating (3.7), we derive

$$\begin{aligned} \mathbb{E}\left[r_{k+1}^2\right] &\leq r_k^2 - 2h_k(f(x_k) - f^* - \frac{C_4^2 L_1 n}{2}|f(x_k)|) + h_k^2[2(n+4)\|\nabla f(x_k)\|^2 + C_5|f(x_k)| + C_6] \\ &\leq r_k^2 - 2h_k(1 - 2h_k L_1(n+4))(f(x_k) - f^*) + (h_k C_4^2 L_1 n + h_k^2 C_5)|f(x_k)| + h_k^2 C_6. \end{aligned}$$

Let $h_k = \dfrac{1}{4L_1(n+4)}$. Then, taking the expectation with respect to $\mathcal{U}_k = \{u_1, \cdots, u_k\}$ and $\mathcal{P}_k = \{\xi_0, \xi_0', \xi_1, \xi_{1'}, \cdots, \xi_k\}$ yields

$$\mathbb{E}_{\mathcal{U}_k, \mathcal{P}_k}\left[r_{k+1}^2\right] \leq \mathbb{E}_{\mathcal{U}_{k-1}, \mathcal{P}_{k-1}}\left[r_k^2\right] - \frac{\phi_k - f^*}{4L_1(n+4)} + C_7|\phi_k| + C_8.$$

Summing these inequalities over $k = 0, \cdots, N$ and dividing by $N+1$, we get

$$\frac{1}{N+1}\sum_{k=0}^{N}(\phi_k - f^*) \leq \frac{4L_1(n+4)}{N+1}\|x_0 - x^*\|^2 + 4L_1(n+4)(C_7 M + C_8).$$

□

The bound (3.30) is valid also for $\hat{\phi}_N = \mathbb{E}_{\mathcal{U}_{k-1},\mathcal{P}_{k-1}}[f(\hat{x}_N)]$, where $\hat{x}_N = \arg\min_x\{f(x) : x \in \{x_0, \cdots, x_N\}\}$. In this case,

$$\begin{aligned}\mathbb{E}_{\mathcal{U}_{k-1},\mathcal{P}_{k-1}}[f(\hat{x}_N)] - f^* &\leq \mathbb{E}_{\mathcal{U}_{k-1},\mathcal{P}_{k-1}}\left[\frac{1}{N+1}\sum_{k=0}^{N}(\phi_k - f^*)\right] \\ &\leq \frac{4L_1(n+4)}{N+1}\|x_0 - x^*\|^2 + 4L_1(n+4)(C_7M + C_8) \quad (3.31)\end{aligned}$$

Let us collect and simplify the constants C_7 and C_8. First, $C_8 = \dfrac{C_6}{16L_1^2(n+4)^2} = \dfrac{3L_0^2\sigma_r^2}{16L_1^2}$. Second, since

$$\begin{aligned}C_5 &= \frac{1}{2}C_4^2L_1^2(n+6)^3 + (1+b)L_1\sigma_r\sqrt{(1+3\sigma_r^2)n(n+6)^3} \\ &= 2L_1\sigma_r\sqrt{\frac{1}{1+3\sigma_r^2}}\sqrt{n(n+6)^3} + (1+b)L_1\sigma_r\sqrt{(1+3\sigma_r^2)n(n+6)^3} \\ &\leq (b+3)L_1\sigma_r\sqrt{1+3\sigma_r^2}\sqrt{n(n+6)^3},\end{aligned}$$

where the last inequality holds because $\dfrac{1}{1+3\sigma_r^2} \leq 1 \leq 1+3\sigma_r^2$, we can derive

$$\begin{aligned}C_7 &= \frac{C_4^2 n}{4(n+4)} + \frac{C_5}{16L_1^2(n+4)^2} \\ &\leq \frac{1}{L_1}\sqrt{\frac{\sigma_r^2}{1+3\sigma_r^2}}\cdot\frac{n}{n+4}\sqrt{\frac{n}{(n+6)^3}} + \frac{(b+3)\sigma_r\sqrt{1+3\sigma_r^2}}{16L_1}\cdot\frac{\sqrt{n(n+6)^3}}{(n+4)^2} \\ &\leq \frac{\sigma_r\sqrt{1+3\sigma_r^2}}{L_1}\left[g_2(n) + (b+3)g_3(n)\right],\end{aligned}$$

where $g_2(n) = \dfrac{n}{n+4}\sqrt{\dfrac{n}{(n+6)^3}}$, $g_3 = \dfrac{\sqrt{n(n+6)^3}}{16(n+4)^2}$, and the last inequality again utilizes $\dfrac{1}{1+3\sigma_r^2} \leq 1 \leq 1+3\sigma_r^2$. It can be shown that $g_2'(n) < 0$ for all $n \geq 8$ and $g_2'(n) > 0$ for all $n \leq 7$, thus $g_2(n) \leq \max\{g(7), g(8)\} = \max\{0.0359, 0.0360\} \leq \dfrac{3}{64}$. Similarly, one can prove that $g_3'(12) = 0$, $g_3'(n) < 0$ for all $n > 12$, and $g_3'(n) > 0$ for

all $n < 12$, which indicates $g_3(n) \leq g_3(12) \approx 0.0646 \leq \frac{3}{32}$. Hence,

$$C_7 \leq \frac{3(2b+7)\sigma_r\sqrt{1+3\sigma_r^2}}{64L_1} \leq \frac{3\sqrt{3}(2b+7)(\sigma_r^2+\frac{1}{6})}{64L_1},$$

where the last inequality holds because $\sigma_r\sqrt{\frac{1}{3}+\sigma_r^2} \leq \sigma_r^2+\frac{1}{6}$.

With C_7 and C_8 simplified, (3.31) can be used to establish an accuracy ϵ for $\hat{\phi}_N$; that is, $\hat{\phi}_N - f^* \leq \epsilon$, can be achieved in $\mathcal{O}\left(\frac{n}{\epsilon}L_1R^2\right)$ iterations, provided the variance of the relative noise σ_r^2 satisfies

$$4L_1(n+4)(C_7M+C_8) \leq \frac{1}{2}C_9(\sigma_r^2+\frac{1}{6})(n+4) \leq \frac{\epsilon}{2},$$

where $C_9 = \frac{3\sqrt{3}}{8}(2b+7)M + \frac{3L_0^2}{2L_1}$, that is,

$$\sigma_r^2 \leq \frac{\epsilon}{C_9(n+4)} - \frac{1}{6}. \tag{3.32}$$

The bound in (3.32) may be cause for concern since the upper bound may only be positive for larger values of ϵ. Rearranging the terms explicitly shows that the additive term $\frac{1}{6}$ is a limiting factor for the best accuracy that can be ensured by this bound:

$$\epsilon_{\text{pred}} \geq C_9(\sigma_r^2+\frac{1}{6})(n+4). \tag{3.33}$$

3.6 Numerical Experiments

We perform three types of numerical studies. Since our convergence rate analysis guarantees only that the means converge, we first test how much variability the performance of STARS show from one run to another. Second, we study the convergence behavior of STARS in both the absolute noise and multiplicative noise cases and examine these results relative to the bounds established in our analysis. Then, we compare STARS with four other randomized zero-order methods to highlight what is gained by using an adaptive smoothing stepsize.

3.6.1 Performance Variability

We first examine the variability of the performance of STARS relative to that of Nesterov's RG algorithm [63], which is summarized in Algorithm 4. One can observe that RG and STARS have identical algorithmic updates except for the choice of the smoothing stepsize μ_k. Whereas STARS takes into account the noise level, RG calculates the smoothing stepsize based on the target accuracy ϵ in addition to the problem dimension and Lipschitz constant,

$$\mu = \frac{5}{3(n+4)}\sqrt{\frac{\epsilon}{2L_1}}. \tag{3.34}$$

MATLAB implementations of both RG and STARS are tested on a smooth convex function with random noise added in both additive and multiplicative forms. In our tests, we use uniform random noise, with ν generated uniformly from the interval $[-\sqrt{3}\sigma, \sqrt{3}\sigma]$ by using MATLAB's random number generator rand. This choice ensures that ν has zero mean and bounded variance σ^2 in both the additive ($\sigma_a = \sigma$)

Algorithm 4 (RG: Random Search for Smooth Optimization)

1: Choose initial point x_0 and iteration limit N. Fix step length $h_k = h = \frac{1}{4(n+4)L_1}$ and compute smoothing stepsize μ_k based on $\epsilon = 2^{-16}$. Set $k \leftarrow 1$.
2: Generate a random Gaussian vector u_k.
3: Evaluate the function values $\tilde{f}(x_k;\xi_k)$ and $\tilde{f}(x_k+\mu_k u_k;\xi_k)$.
4: Call the random stochastic gradient-free oracle

$$s_\mu(x_k;u_k,\xi_k) = \frac{\tilde{f}(x_k+\mu_k u_k;\xi_k) - \tilde{f}(x_k;\xi_k)}{\mu_k} u_k.$$

5: Set $x_{k+1} = x_k - h_k s_\mu(x_k;u_k,\xi_k)$, update $k \leftarrow k+1$, and return to Step 2.

and multiplicative cases ($\sigma_r = \sigma$) and that Assumptions 3.4.2 and 3.5.2 hold, provided that $\sigma < 3^{-1/2}$.

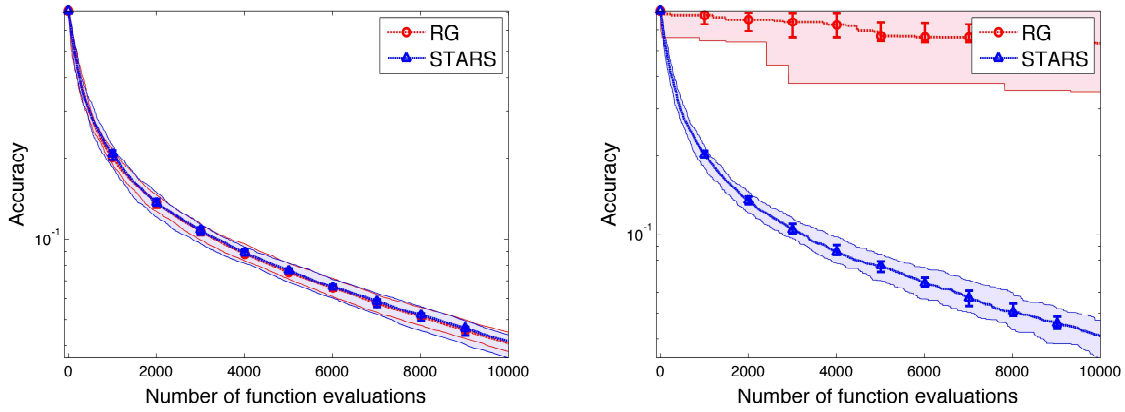
We use Nesterov's smooth function as introduced in [63]:

$$f_1(x) = \frac{1}{2}(x^{(1)})^2 + \frac{1}{2}\sum_{i=1}^{n-1}(x^{(i+1)} - x^i)^2 + \frac{1}{2}(x^{(n)})^2 - x^{(1)}, \tag{3.35}$$

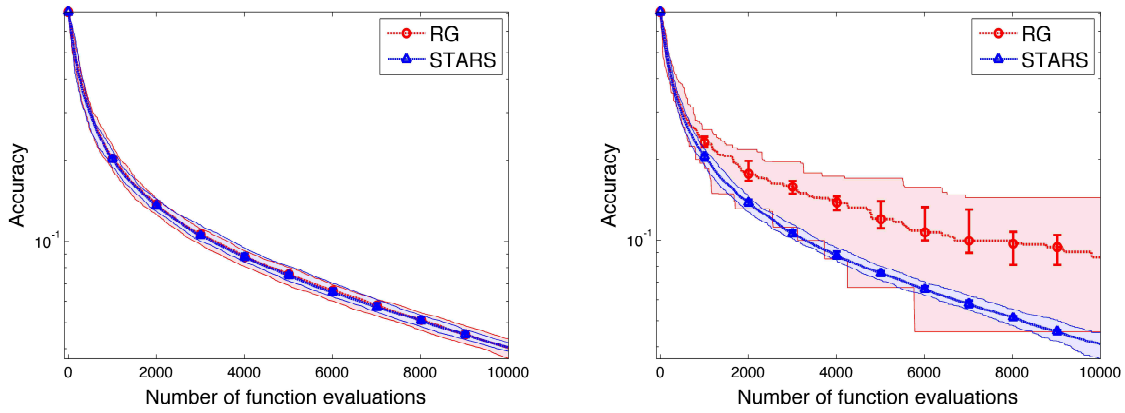
where $x^{(i)}$ denotes the ith component of the vector $x \in \mathbb{R}^n$. The starting point specified for this problem is the vector of zeros, $x_0 = \mathbf{0}$. The optimal solution is

$$x^{*(i)} = 1 - \frac{i}{n+1},\ i = 1,\cdots,n; \quad f(x^*) = -\frac{n}{2(n+1)}.$$

The analytical values for the parameters (corresponding to Lipschitz constant for the gradient and the squared Euclidean distance between the starting point and optimal solution) are: $L_1 \leq 4$ and $R^2 = \|x_0 - x^*\|^2 \leq \frac{n+1}{3}$. Both methods were given the same parameter value (4.0) for L_1, but the smoothing stepsizes differ. Whereas RG always uses fixed stepsizes of the form (3.34), STARS uses fixed stepsizes of the form (3.18) in the absolute noise case and uses dynamic stepsizes calculated as (3.28) in the multiplicative noise case. To observe convergence over many random trials, we



(a) $\sigma_a = 10^{-6}$ (b) $\sigma_a = 10^{-3}$



(c) $\sigma_r = 10^{-6}$ (d) $\sigma_r = 10^{-3}$

Figure 3.1: Median and quartile plots of achieved accuracy with respect to 20 random seeds when applying RG and STARS to the noisy f_1 function. Figure 3.1(a) and 3.1(b) show the additive noise case, while Figure 3.1(c) and 3.1(d) show the multiplicative noise case.

use a small problem dimension of $n = 8$; however, the behavior shown in Figure 3.1 is typical of the behavior that we observed in higher dimensions (but the $n = 8$ case requiring fewer function evaluations).

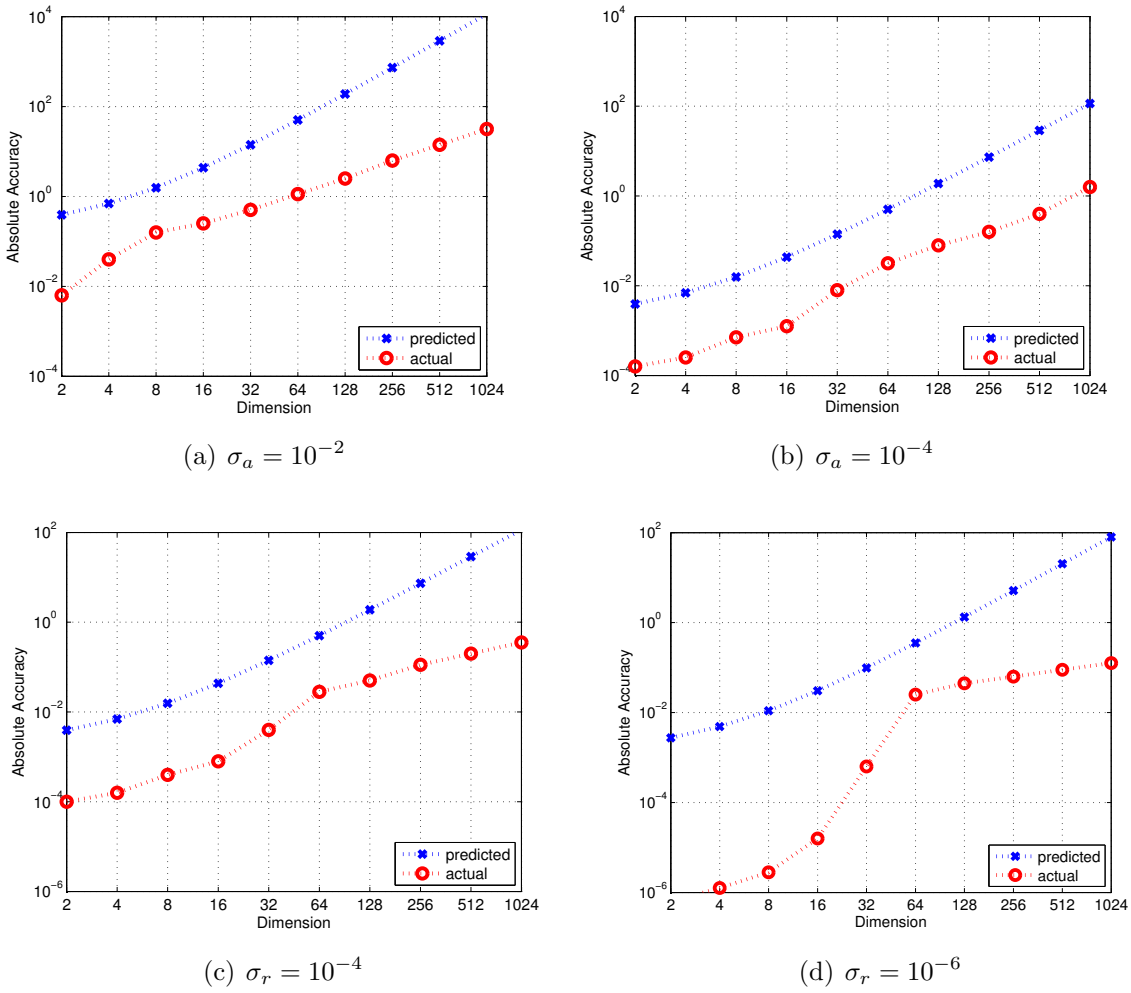
In Figure 3.1, we plot the accuracy achieved at each function evaluation, which is the true function value $f(x_k)$ minus the optimal function value $f(x^*)$. The median

across 20 trials is plotted as a line; the shaded region denotes the best and worst trials; and the 25% and 75% quartiles are plotted as error bars. We observe that when the function is relatively smooth, as in Figure 3.1(a) when the additive noise is 10^{-6}, the methods exhibit similar performance. As the function gets more noisy, however, as in Figure 3.1(b) when the additive noise becomes 10^{-3}, RG shows more fluctuations in performance resulting in large variance, whereas the performance STARS is almost the same as in the smoother case. The same noise-invariant behavior of STARS can be observed in the multiplicative case.

3.6.2 Convergence Behavior

We tested the convergence behavior of STARS with respect to dimension n and noise levels on the same smooth convex function f_1 with noise added in the same way as in Section 3.6.1. The results are summarized in Figure 3.2 , where (a) and (b) are for the additive case and (c) and (d) are for the multiplicative case. The horizontal axis marks the problem dimension and the vertical axis shows the absolute accuracy. Two types of absolute accuracy are plotted. First, ϵ_{pred} (in blue $\times$'s) is the best achievable accuracy given a certain noise level, computed by using (3.25) for the additive case and (3.32) for the multiplicative case. Second is the actual accuracy (in red circle) achieved by STARS after N iterations where N, calculated as in (3.24), is the number of iterations needed in theory to get ϵ_{pred}. Because of the stochastic nature of STARS, we perform 15 runs (each with a different random seed) of each test and report the averaged accuracy

$$\bar{\epsilon}_{\text{actual}} = \frac{1}{15}\sum_{i=1}^{15} \epsilon^i_{\text{actual}} = \frac{1}{15}\sum_{i=1}^{15} (f(x^i_N) - f^*). \tag{3.36}$$



(a) $\sigma_a = 10^{-2}$

(b) $\sigma_a = 10^{-4}$

(c) $\sigma_r = 10^{-4}$

(d) $\sigma_r = 10^{-6}$

Figure 3.2: Convergence behavior of STARS: absolute accuracy versus dimension n. Two absolute noise levels (a) and (b), and two relative noise levels (c) and (d) are presented.

We observe from Figure 3.2 that the solution obtained by STARS within the iteration limit N is more accurate than that predicted by the theoretical bounds. The difference between predicted and achieved accuracy is always over an order of magnitude and is relatively consistent for all dimensions we examined.

3.6.3 Illustrative Example

In this section, we provide a comparison between STARS and four other zero-order algorithms on noisy versions of (3.35) with $n = 8$. The methods we study all share a stochastic nature; that is, a random direction is generated at each iteration. Except for RP [79], which is designed for solving smooth convex functions, the rest are stochastic optimization algorithms. However, we still include RP in the comparison because of its similar algorithmic framework. The algorithms and their function-specific inputs are summarized in Table 3.1, where $\tilde{L}_1$ and $\tilde{\sigma}^2$ are, respectively, estimations of L_1 and σ^2 given a noisy function (details on how to estimate $\tilde{L}_1$ and $\tilde{\sigma}^2$ are discussed in Appendix). We now briefly introduce each of the tested algorithms; algorithmic and implementation details are given in the appendix of this chapter.

Table 3.1: Relevant function parameters for different methods.

Abbreviation	Method Name	Parameters
STARS	Stepsize Approximation in Random Search	L_1, σ^2
SS	Random Search for Stochastic Optimization [63]	L_0, R^2
RSGF	Random Stochastic Gradient Free method [38]	$\tilde{L}_1, \tilde{\sigma}^2$
RP	Random Pursuit [79]	-
ES	(1+1)-Evolution Strategy [74]	-

The first zero-order method we include, named SS (Random Search for Stochastic Optimization), is proposed in [63] for solving (3.1). It assumes that $f \in \mathcal{C}^{0,0}(\mathbb{R}^n)$ is convex. The SS algorithm, summarized in Algorithm 5, shares the same algorithmic framework as STARS except for the choice of smoothing stepsize μ_k and the step length h_k. It is shown that the quantities μ_k and h_k can be chosen so that a solution for (3.1) such that $f(x_N) - f^* \leq \epsilon$ can be ensured by SS in $\mathcal{O}(n^2/\epsilon^2)$ iterations.

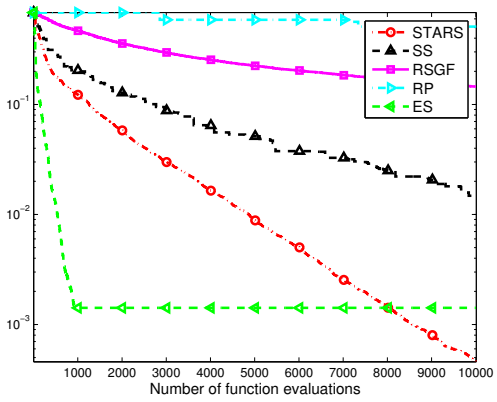
Another stochastic zero-order method that also shares an algorithmic framework

similar to STARS is RSGF [38], which is summarized in Algorithm 6. RSGF targets the stochastic optimization objective function in (3.1), but the authors relax the convexity assumption and allow f to be nonconvex. However, it is assumed that $\tilde{f}(\cdot, \xi) \in \mathcal{C}^{1,1}(\mathbb{R}^n)$ almost surely, which implies that $f \in \mathcal{C}^{1,1}(\mathbb{R}^n)$. The authors show that the iteration complexity for RSGF finding an ϵ-accurate solution, (i.e., a point $\bar{x}$ such that $\mathbb{E}[\|\nabla f(\bar{x})\|] \leq \epsilon$) can be bounded by $\mathcal{O}(n/\epsilon^2)$. Since such a solution $\bar{x}$ satisfies $f(\bar{x}) - f^* \leq \epsilon$ when f is convex, this bound improves Nesterov's result in [63] by a factor n for convex stochastic optimization problems.

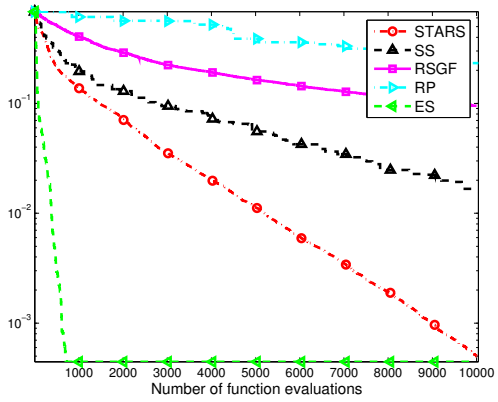
In contrast with the presented randomized approaches that work with a Gaussian vector u, we include an algorithm that samples from a uniform distribution on the unit hypersphere. Summarized in Algorithm 7, RP [79] is designed for unconstrained, smooth, convex optimization. It relaxes the requirement in [63] of approximating directional derivatives via a suitable oracle. Instead, the sampling directions are chosen uniformly at random on the unit hypersphere, and the step lengths are determined by a line search oracle. This randomized method also requires only zeroth-order information about the objective function, but it does not need any function-specific parametrization. It was shown that RP meets the convergence rates of the standard steepest descent method up to a factor n.

Experimental studies of variants of (1+1)-Evolution Strategy (ES), first proposed by Schumer and Steiglitz [74], have shown their effectiveness in practice and their robustness in noisy environment. However, provable convergence rates are derived only for the simplest forms of ES on unimodal objective functions [7, 43, 45], such as sphere or ellipsoidal functions. The implementation we study is summarized in Algorithm 8; however, different variants of this scheme have been studied in [13].

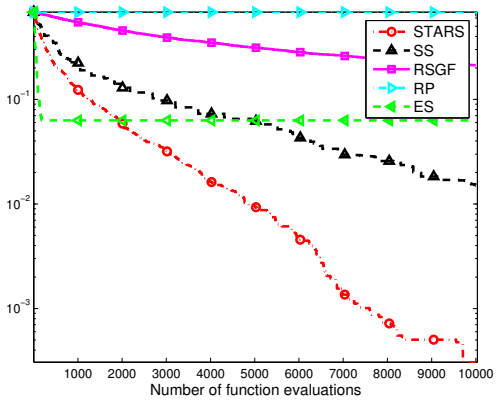
We observe from Figure 3.3 that STARS outperforms the other four algorithms



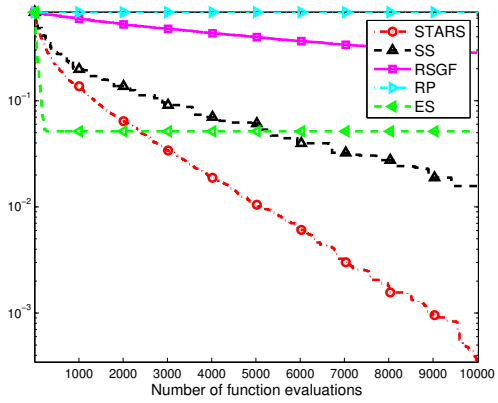
(a) $\sigma_a = 10^{-5}$



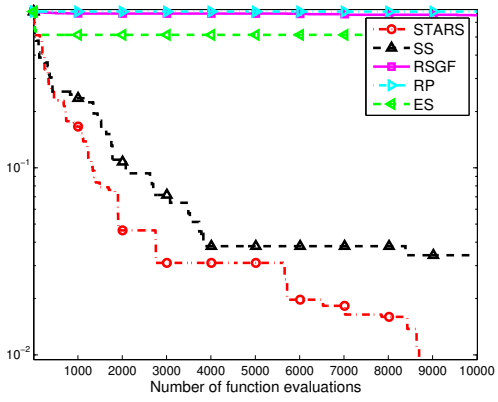
(b) $\sigma_r = 10^{-5}$



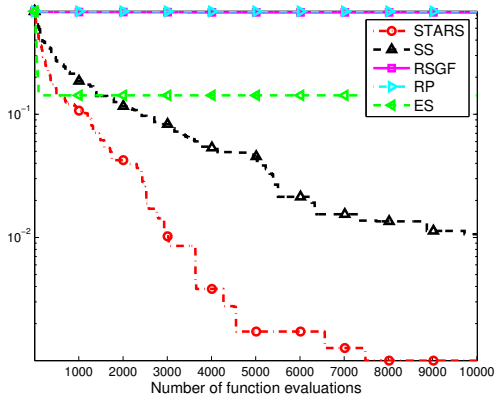
(c) $\sigma_a = 10^{-3}$



(d) $\sigma_r = 10^{-3}$



(e) $\sigma_a = 10^{-1}$



(f) $\sigma_r = 10^{-1}$

Figure 3.3: Trajectory plots of five zero-order methods in the additive and multiplicative noise settings. The vertical axis represents the true function value $f(x_k)$, and each line is the mean of 20 trials.

in terms of final accuracy in the solution. In both Figures 3.3(a) and 3.3(b), ES is the fastest algorithm among all in the beginning. However, ES stops progressing after a few iterations, whereas STARS keeps progressing to a more accurate solution. As the noise level increases from 10^{-5} to 10^{-1}, the performance of ES gradually worsens, similar to the other methods SS, RSGF, and RP. However, the noise-invariant property of STARS allows it to remain robust in these noisy environments.

3.7 Appendix

In this appendix we describe the implementation details of the four zero-order methods tested in Table 3.1 and Section 3.6.3.

Random Search for Stochastic Optimization

Algorithm 5 (SS: Random Search for Stochastic Optimization)

1: Choose initial point x_0 and iteration limit N. Fix step length $h_k = h = \dfrac{R}{(n+4)(N+1)^{1/2}L_0}$ and smoothing stepsize $\mu_k = \mu = \dfrac{\epsilon}{2L_0 n^{1/2}}$. Set $k \leftarrow 1$.
2: Generate a random Gaussian vector u_k.
3: Evaluate the function values $\tilde{f}(x_k;\xi_k)$ and $\tilde{f}(x_k+\mu_k u_k;\xi_k)$.
4: Call the random stochastic gradient-free oracle

$$s_\mu(x_k;u_k,\xi_k) = \frac{\tilde{f}(x_k+\mu_k u_k;\xi_k) - \tilde{f}(x_k;\xi_k)}{\mu_k} u_k.$$

5: Set $x_{k+1} = x_k - h_k s_\mu(x_k;u_k,\xi_k)$, update $k \leftarrow k+1$, and return to Step 2.

Algorithm 5 provides the SS (Random Search for Stochastic Optimization) algorithm from [63].

Remark: ϵ is suggested to be 2^{-16} in the experiments in [63]. Our experiments in Section 3.6.3, however, show that this choice of ϵ forces SS to take small steps and thus SS does not converge at all in the noisy environment. Hence, we increase ϵ (to $\epsilon = 0.1$) to show that optimistically, SS will work if the stepsize is big enough. Although in the additive noise case one can recover STARS by appropriately setting this ϵ in SS, it is not possible in the multiplicative case because STARS takes dynamically adjusted smoothing stepsizes in this case.

Randomized Stochastic Gradient-Free Method

Algorithm 6 (RSGF: Randomized Stochastic Gradient-Free Method)

1: Choose initial point x_0 and iteration limit N. Estimate L_1 and $\tilde{\sigma}^2$ of the noisy function $\tilde{f}$. Fix step length as

$$\gamma_k = \gamma = \frac{1}{\sqrt{n+4}} \min\left\{ \frac{1}{4L_1\sqrt{n+4}}, \frac{\tilde{D}}{\tilde{\sigma}\sqrt{N}} \right\},$$

where $\tilde{D} = (2f(x_0)/L_1)^{\frac{1}{2}}$. Fix $\mu_k = \mu = 0.0025$. Set $k \leftarrow 1$.

2: Generate a Gaussian vector u_k.

3: Evaluate the function values $\tilde{f}(x_k; \xi_k)$ and $\tilde{f}(x_k + \mu_k u_k; \xi_k)$.

4: Call the stochastic zero-order oracle

$$G_\mu(x_k; u_k, \xi_k) = \frac{\tilde{f}(x_k + \mu_k u_k; \xi_k) - \tilde{f}(x_k; \xi_k)}{\mu} u_k.$$

5: Set $x_{k+1} = x_k - \gamma_k G_\mu(x_k; u_k, \xi_k)$, update $k \leftarrow k + 1$, and return to Step 2.

Algorithm 6 provides the RSGF (Randomized Stochastic Gradient-Free Method) algorithm from [38].

Remark: Although the convergence analysis of RSGF is based on knowledge of the constants Ł$_1$ and σ^2, the discussion in [38] on how to implement RSGF does not reply on these inputs. Because the authors solved a support vector machine problem and an

inventory problem, both of which do not have known L_1 and σ^2 values, they provide details on how to estimate these parameters given a noisy function. Hence following [38], the parameter L_1 is estimated as the l_2 norm of the Hessian of the deterministic approximation of the noisy objective functions. This estimation is achieved by using a sample average approximation approach with 200 i.i.d. samples. Also, we compute the stochastic gradients of the objective functions at these randomly selected points and take the maximum variance of the stochastic gradients as an estimate of $\tilde{\sigma}^2$.

Random Pursuit

Algorithm 7 (RP: Random Pursuit)

1: Choose initial point x_0, iteration limit N, and line search accuracy $\mu = 0.0025$. Set $k \leftarrow 1$.
2: Choose a random Gaussian vector u_k.
3: Choose $x_{k+1} = x_k + \mathsf{LS}_{\mathsf{APPROX}_\mu}(x_k, u_k) \cdot u_k$, update $k \leftarrow k+1$, and return to Step 2.

Algorithm 7 provides the RP (Random Pursuit) algorithm from [79].

Remark: We follow the authors in [79] and use the built-in MATLAB routine fminunc.m as the approximate line search oracle.

$(1+1)$-Evolution Strategy

Algorithm 8 provides the ES ($(1+1)$-Evolution Strategy) algorithm from [74].

Remark: A problem-specific parameter required by Algorithm 8 is the initial stepsize σ_0, which is given in [79] for some of our test functions. The stepsize is multiplied by a factor $c_s = e^{1/3} > 1$ when the mutant's fitness is as good as the parent is and is

Algorithm 8 (ES: $(1+1)$-Evolution Strategy)

1: Choose initial point x_0, initial stepsize σ_0, iteration limit N, and probability of improvement $p = 0.27$. Set $c_s = e^{\frac{1}{3}} \approx 1.3956$ and $c_f = c_s \cdot e^{\frac{-p}{1-p}} \approx 0.8840$. Set $k \leftarrow 1$.
2: Generate a random Gaussian vector u_k.
3: Evaluate the function values $\tilde{f}(x_k; \xi_k)$ and $\tilde{f}(x_k + \sigma_k u_k; \xi_k)$.
4: If $\tilde{f}(x_k + \sigma_k u_k; \xi_k) \leq \tilde{f}(x_k; \xi_k)$, then set $x_{k+1} = x_k + \sigma_k u_k$ and $\sigma_{k+1} = c_s \sigma_k$; Otherwise, set $x_{k+1} = x_k$ and $\sigma_{k+1} = c_f \sigma_k$.
5: Update $k \leftarrow k + 1$ and return to Step 2.

otherwise multiplied by $c_s \cdot e^{\frac{-p}{1-p}} < 1$, where p is the probability of improvement set to the value 0.27 suggested by Schumer and Steiglitz [74].

3.8 Conclusions and Future Work

In this chapter, we proposed a randomized derivative-free method, STARS. Using noise-adjusted smoothing step sizes, our method is designed for solving general noisy problems with moderate stochastic noise. We derived a convergence proof showing that the convergence rate of our algorithm can achieve a theoretical bound of the same order of the original RG algorithm, in both additive noise and multiplicative noise case. Then we present numerical experiments that compare our proposed method with selected derivative-free algorithms. The computational results concerning short term behavior of the algorithms reveal that STARS clearly outperforms the original RG algorithm when the functions are associated with stochastic randomness. Moreover, STARS exhibits noise-invariant behavior with respect to different levels of stochastic noise.

In the future, we hope to conduct extensive tests in an effort to better delineate the types of functions on which we expect STARS to perform well. For larger scale problems, whether STARS can outperform the model-based methods remains an open question. Moreover, we intend to explore the performance of these algorithms on nonconvex problems, since for example, Nesterov's RS and Lan's RSGF both have theoretical convergence proofs on nonconvex problems, but supporting computational experiments are quite preliminary and not sufficient.

Chapter 4

Stochastic DFO using Probabilistic Models

In this chapter, we propose STORM, a trust-region model-based algorithm for solving unconstrained stochastic optimization problems. This chapter consists of three 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 iterate. It also relies on (random, noisy) estimates of the function values at the current 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. In the second part of the chapter we present some novel ideas about generating the sufficiently accurate random models by randomly sampling the objective function and constructing regression models based on these samples. Lastly we present some computational resulting showing the benefits of using random models and estimates, as well as the performance of STORM applied to the protein alignment problem described in Chapter 2.

4.1 Introduction

We aim to minimize a given function $f(x)$, $x \in \mathbb{R}^n$ which we assume 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 i.i.d with $\mathbb{E}[\varepsilon] = 0$ and $\mathrm{Var}(\varepsilon) = \sigma^2 < \infty$.

The overall goal is to solve

$$\min_x \ f(x) = \mathbb{E}[\tilde{f}(x)]. \tag{4.1}$$

Notations. Let $\|\cdot\|$ denote the Euclidean norm and $B(x, \Delta)$ denote the ball of radius Δ around x, i.e., $B(x, \Delta) : \{y : \|x - y\| \leq \Delta\}$. For convenience, we list here several constants that are used in the chapter 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".

4.2 The STORM Algorithm

We consider the trust-region class of methods for minimization of stochastic functions. They operate as follows: at each iteration k, given the current iterate x_k and a trust-region radius δ_k, build a model $m_k(x)$ which serves as an approximation of $f(x)$ in $B(x_k, \delta_k)$. Then $m_k(x)$ is minimized (approximately) in $B(x_k, \delta_k)$ to produce a step s_k and 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 at x_k. The trust-region radius δ_{k+1} is then chosen by updating δ_k according to some rules. The details of the algorithm is presented in Algorithm 9.

Algorithm 9 STORM:
STOCHASTIC TRUST-REGION-BASED OPTIMIZATION USING 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 $\gamma > 1$, $0 < \eta_1, \eta_2 < 1$; 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 a trial step

$$s_k = \arg \min_{s:\|s\| \le \delta_k} m_k(s)$$

(approximately) so that it satisfies condition (4.2).

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

$$\rho_k = \frac{f_k^0 - f_k^s}{m_k(x_k) - m_k(x_k + s_k)}.$$

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 fractional Cauchy decrease condition.

Assumption 4.2.1. *For every k, the step s_k is computed so that*

$$m_k(x_k) - m_k(x_k + s_k) \geq \frac{\kappa_{fcd}}{2} \|g_k\| \min \left\{ \frac{\|g_k\|}{\|H_k\|}, \delta_k \right\} \tag{4.2}$$

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

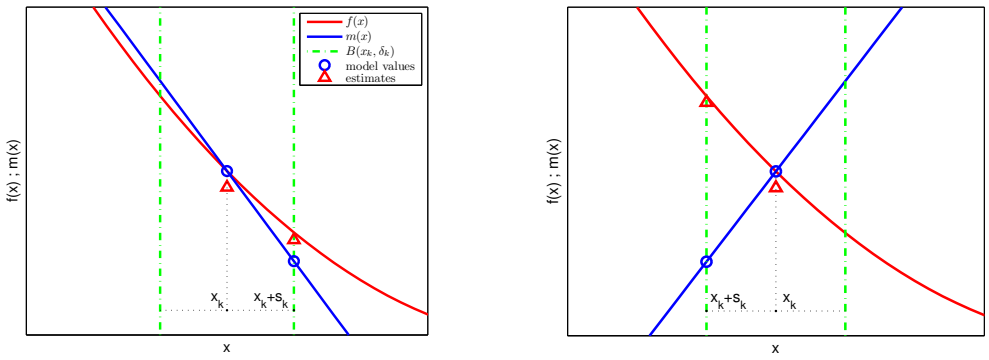
4.3 Probabilistic Models and Estimates

4.3.1 Motivation and Complications

Complication 1: False Successful Steps.

In Algorithm 9, if progress is achieved and a new iterate is accepted on the k-th step then we call this a successful step. Otherwise, the iteration is unsuccessful (and no step is taken). Hence a successful step occurs when $\rho_k \geq \eta_1$ and $\|g_k\| \geq \eta_2 \delta_k$. However, a successful step does not necessarily yields 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 the estimates, f_k^0 and f_k^s are not accurate enough, a successful step can result in the increase of the true function value.

Hence we consider two types of successful steps. Those where $f(x)$ is in fact decreased proportionally to $f_k^0 - f_k^s$ as seen in Figure 4.1(a), which we call *true* successful steps. All other successful steps, where the decrease of $f(x)$ can be arbitrary small



(a) Good model and good estimates yield true successful steps.

(b) Bad model and good estimates yield unsuccessful steps.

(c) Good model and bad estimates yield unsuccessful steps.

(d) Bad model and bad estimates yield false successful steps. f can increase.

Figure 4.1: Complications of using estimates.

or even negative, which we call *false* successful steps. Figure 4.1 gives an illustration of possible outcomes.

Suppose at a certain iteration, the model is fully linear. Minimizing the model over the trust region would yield a decrease in f. Ideally, if the estimates are accurate like in Figure 4.1(a), such a step would be correctly accepted. There are the *true* successful steps. Similarly, a bad trial step obtained from a bad model can also be correctly rejected if the estimates are accurate enough (See Figure 4.1(b)). However, if an inaccurate estimate of $f(x_k^+)$ is obtained as shown in Figure 4.1(c), such a good

step ensured by a fully linear model can be rejected. Similarly, we could accept bad steps because of bad estimates, which are referred as false successful steps. Figure 4.1(d) shows that on these steps $f(x)$ can increase, which is an outcome that does not happen in the deterministic case. As a result, $f(x)$ values may oscillate up and down indefinitely, by taking true and false successful steps alternatively. Hence we cannot state that all decrease obtained on successful steps is bounded.

Our setting and algorithmic framework does not allow us to determine which steps are true and which ones are false, however, we will be able to show that true steps happen sufficiently often for convergence, if the random estimates f_k^0 and f_k^s are sufficiently accurate.

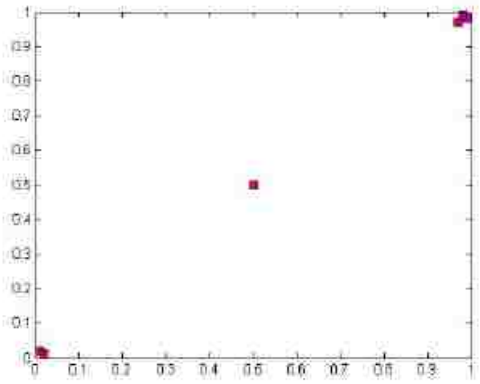
Complication 2: : Geometry/Poisedness of A Sample Set.

Constructing models based on well-poised sample sets produces good approximations of the objective function. Figure 4.2 [32] gives two examples where bad geometry of the sample set results in poor models. In Figure 4.2(a), six sample points align on a line. Thus, the resulting interpolation model (Figure 4.2(b)) only matches the function at these selected points, but not any where else in the trust region. Similarly, when the points are selected on a circle (Figure 4.2(c)), the resulting model also approximates the function poorly within the trust region.

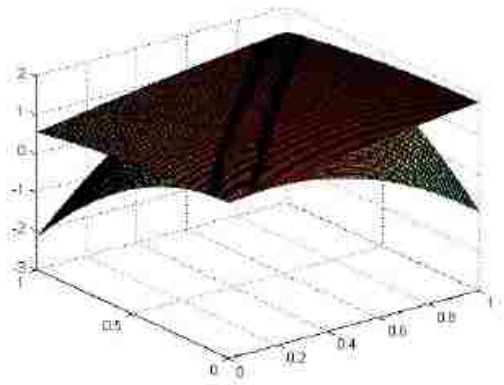
While there have been works on how to generate points with good geometry and the poisedness of a sample set has been well-defined in the literature, random sampling of points may also give well-poised sets without the cost of geometry correction. Figure 4.3 [32] gives such an example. A good model (Figure 4.3(b)) can be obtained from a carefully selected sample set (Figure 4.3(a)). An almost equally good model ((Figure 4.3(d))) can also be obtained a randomly generated set ((Figure 4.3(c))).

A trust region framework based on random models was introduced and analyzed in [9]. 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.

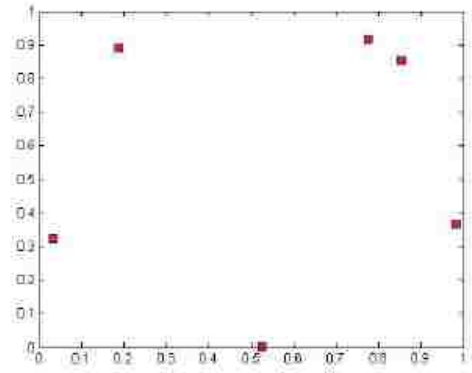
However, the randomness in the models in their setting arises from the the construction process, and not from the noisy objective function. It is assumed in [9] that the function values at the current iterate and the trial point can be computed explicitly and hence all successful iterations are true in that case. In our case, it is



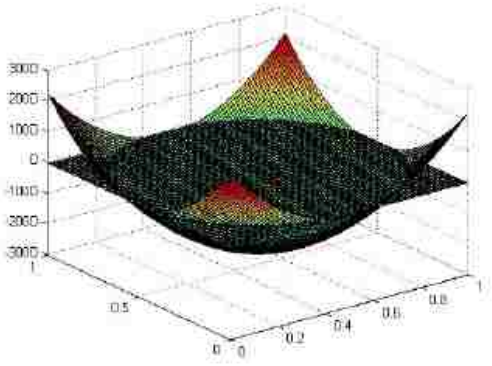
(a) Linear non-poised set.



(b) Linear non-poised Model.

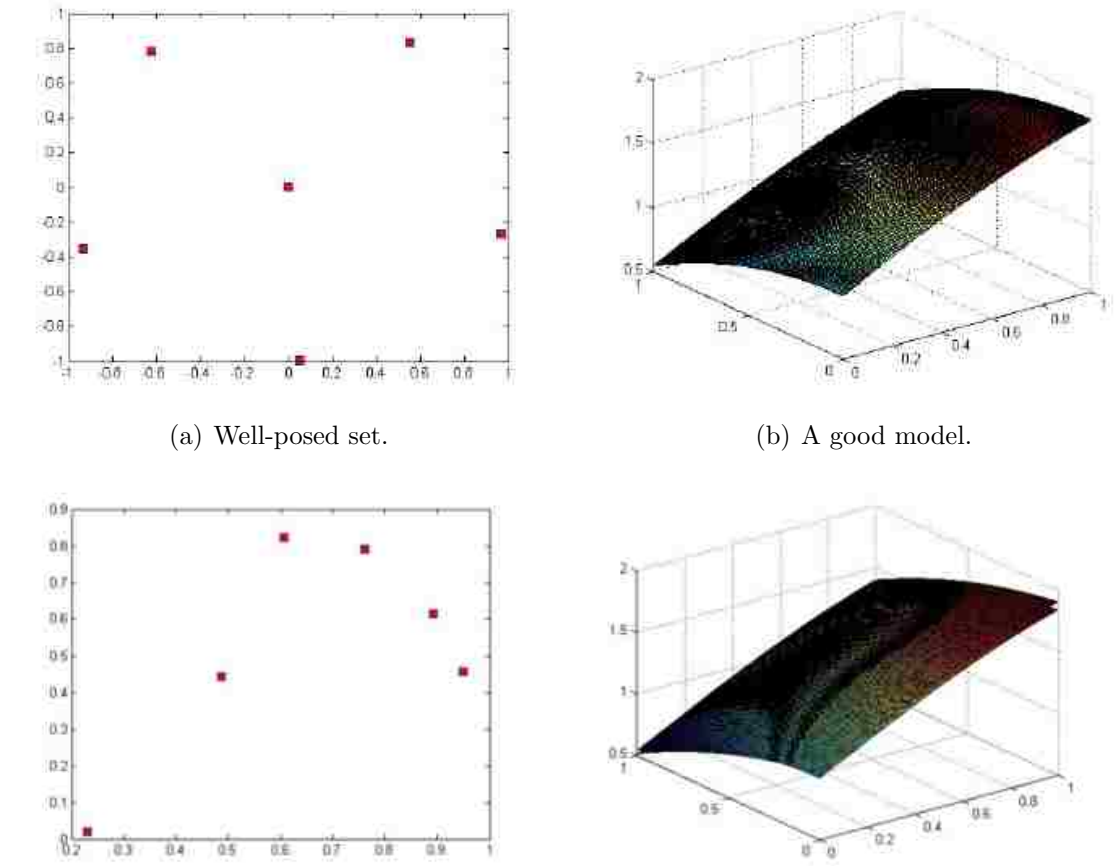


(c) Circle non-poised set.



(d) Circle non-poised model.

Figure 4.2: Two examples of non-poised set.



(a) Well-posed set.

(b) A good model.

(c) Randomly sampled set.

(d) Also a good model.

Figure 4.3: Random sampling may give well-poised sets.

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 [51].

4.3.2 Definitions

The models in this chapter, 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 on the k-th iteration, while we will use the notation $m_k = M_k(\omega_k)$ for its realizations, for a given sample $\omega \in \Omega$, where Ω denotes the probability sample space. As a consequence of using random models, the iterates X_k, the trust-region radius Δ_k and the step S_k are also random quantities, and thus $x_k = X_k(\omega_k)$, $\delta_k = \Delta_k(\omega_k)$, $s_k = S_k(\omega_k)$ 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 $f_k^0 = F_k^0(\omega_k)$ and $f_k^s = F_k^s(\omega_k)$. In other words, Algorithm 9 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 base desirable convergence properties with probability one.

We begin by recalling a measure for the accuracy of deterministic models introduced in [32] and [31] (with the exact notation introduced in [14]).

Definition 4.3.1. *∇f is Lipschitz continuous. A function m_k is a κ-fully linear model of f on $B(x_k, \delta_k)$ if, $\exists \kappa = (\kappa_{ef}, \kappa_{eg})$, s.t. $\forall y \in B$,*

$$\begin{aligned} \|\nabla f(y) - \nabla m_k(y)\| &\leq \kappa_{eg}\delta_k, \text{ and} \\ |f(y) - m_k(y)| &\leq \kappa_{ef}\delta_k^2. \end{aligned} \tag{4.3}$$

In this chapter we rely on the following concept of probabilistically fully-linear models which is proposed in [9].

Definition 4.3.2. *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*

$$I_k = \{M_k \text{ is a } \kappa\text{-fully linear model of } f \text{ on } B(X_k, \Delta_k)\} \tag{4.4}$$

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* $M_0, \cdots, M_{k-1}$.

This 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 some assumption on the expected value and the variance of the models is imposed.

In order to evaluate whether a step is successful, we require estimates of the function values $f(x_k)$, $f(x_k + s_k)$ that are sufficiently accurate. The following definitions of accurate and probabilistically accurate estimates are a modified version of those used in [51].

Definition 4.3.3. *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*

$$|f_k^0 - f(x_k)| \leq \epsilon_F \delta_k^2 \text{ and } |f_k^s - f(x_k + s_k)| \leq \epsilon_F \delta_k^2. \tag{4.5}$$

Now, let $\{F_k^0\}$ and $\{F_k^s\}$ denote the sequences of estimates for $\{f(x_k)\}$, $\{f(x_k + s_k)\}$ respectively.

Definition 4.3.4. *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

$$J_k = \{F_k^0, F_k^s \text{ are } \epsilon_F\text{-accurate estimates of } f(x_k) \text{ and } f(x_k+s_k), \text{ respectively, for } \Delta_k\} \tag{4.6}$$

satisfy the condition

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

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

As can be seen from the Definitions 4.3.2 and 4.3.4 the accuracy of the models and estimates with respect to $f(x)$ is required to be proportionate to δ_k^2. However, as will be evident from our analysis below, in the case of the fully-linear models, all that is required in the existence of fixed constants κ_{ef} and κ_{eg} such that (4.3) holds, while in the case of ϵ_F-accurate estimates it is required that (4.5) holds for some given, small enough, ϵ_F. The latter, by comparison, is a tighter requirement. However, we will see that the upper bound on ϵ_F that are sufficient for convergence is reasonably large. We will also assume, for simplicity of the analysis, that the models and the estimates are conditionally independent from each other, given the past. This is a reasonable assumption, which means that the estimates of the function value should not be computer using the current model. This assumption can be relaxed, by introducing additional constraints on dependency of models and estimates.

Procedures for obtaining probabilistically fully-linear models and probabilistically accurate estimates are described in Sections 4.5 and 4.6.2, respectively.

4.4 Convergence Analysis

4.4.1 Key Challenges

Typically a standard step in proving the convergence of a trust region method is showing that the trust-region radius δ_k goes to zero, that is, formally,

$$\lim_{k\to\infty} \delta_k = 0.$$

In the case where random models are used for optimizing deterministic functions, it is not certain whether the model is accurate or not at each iteration. Thus, in [9], the authors showed that δ_k is always driven to zero regardless of the realization of the model sequence $\{M_k\}$ of the algorithm, as long as the fraction of Cauchy decrease is achieved by each iteration.

Suppose that this property can be shown in the stochastic case, where one wants to minimize a deterministic smooth function f when having access only to noisy-corrupted values $\tilde{f}$. The convergence is then a natural extension of that in [9] using submartingale-like properties, as long as we assume that the model sequence $\{M_k\}$ is α-probabilistically $(\kappa_{ef}, \kappa_{eg})$ fully linear and the estimate sequence $\{F_k^0, F_k^s\}$ is θ-probabilistically ϵ_F-accurate, with $\alpha + \beta \geq 3/2$. The motivation is that, if $\|\nabla f(x)\|$ does not go to zero, then on some iterations where the model is fully linear, the model gradient would be sufficiently large relative to δ_k (because $\delta_k \to 0$). This results in a successful step and an increase in δ_k. One can then show that δ_k has to be oscillating above and below some positive constant b infinitely often with probability one, contradicting the fact that δ_k goes to zero.

However, in order to prove $\delta_k \to 0$, one needs to show that, at each iteration

where δ_k is increased, f is reduced by a constant. That is because, an increase in δ_k only happens when a trial step is successful, i.e, $\rho_k = \dfrac{f(x_k) - f(x_{k+1})}{m_k(x_k) - m_k(x_{k+1})} > \eta_1$, and we know that the model has achieved a fraction of Cauchy decrease. Hence, the function value decreases on all successful steps. Then, if one assumes that δ_k does not converge to zero, there must infinite number of iterations on which δ_k is not decreased, thus increased. However, since f is bounded from below, the number of such iterations cannot be infinite, and hence we obtain a contradiction.

Unfortunately, it is no longer true in the stochastic case that f always decreases on successful iterations. When the measurable function value $\tilde{f}$ is noisy, one can only compute

$$\tilde{\rho}_k = \frac{\tilde{f}(x_k) - \tilde{f}(x_{k+1})}{m_k(x_k) - m_k(x_{k+1})}.$$

Now a successful iteration indicates that $\tilde{f}(x_k) < \tilde{f}(x_{k+1})$, but the true function reduction is unknown. In fact, f might falsely increase.

The key is to show that δ_k still converges to zero even though false increase in f can happen. This is however complicated by the fact that the model and the estimates are both be inaccurate and various random outcomes become possible, as shown in Figure 4.1.

Our main motivation is that one should be able to upper bound the false increase in f using function Lipschitz continuity within the trust region. Moreover, false increase does not happen too often if our model and estimates are accurate with sufficiently high probability. For instance, if our model is fully linear with probability α and the estimates are accurate with probability β, the probability for the situation in Figure 4.1(d) to happen is only $(1 - \alpha)(1 - \beta)$. Hence, one can use the true decrease in f on true successful steps to counter-balance the false increase and use this to show $\delta_k \to 0$.

4.4.2 Convergence of Trust Region Radius

We now present convergence analysis for the general framework described in Algorithm 9. For the purpose of proving first-order convergence of the algorithm, we assume that the function f and its gradient are Lipschitz continuous in regions considered by the algorithm realizations. We follow the process in [31] to define this region.

Assumption 4.4.1 (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 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 4.4.2. *There exists a positive constant κ_{bhm} such that, for every k, the Hessian H_k of all realization 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 chapter, this bounds 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, however, as we will see in the convergence analysis, it will add restrictions on the trust region radius and some other algorithmic parameters.

We state the following result martingale literature [35] that will be useful later in our analysis.

Theorem 4.4.3. *Let G_k be a submartingale, i.e., a sequence of random variables which, for every k,*

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

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

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

$$P\left(\left\{\lim_{k\to 0} G_k < \infty\right\} \cap \left\{\limsup_{k\to 0} G_k = \infty\right\}\right) = 1. \tag{4.7}$$

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 relatively 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_k^0 and f_k^s are not accurate enough.

Lemma 4.4.4. *Suppose that a model is $(\kappa_{ef}, \kappa_{eg})$-fully linear on $B(x_k, \delta_k)$. If*

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

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

$$f(x_k + s_k) - f(x_k) \leq -\frac{\kappa_{fcd}}{4}\|g_k\|\delta_k. \tag{4.8}$$

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 $\delta_k \leq \dfrac{\kappa_{fcd}}{8\kappa_{ef}}\|g_k\|$. □

The next lemma shows that for δ_k small enough relatively 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 4.4.5. *Under Assumptions 4.4.1 and 4.4.2, suppose that a model is $(\kappa_{ef}, \kappa_{eg})$-fully linear on $B(x_k, \delta_k)$. If*

$$\delta_k \leq \min\left\{\frac{1}{\kappa_{bhm} + \kappa_{eg}}, \frac{1}{\frac{8\kappa_{ef}}{\kappa_{fcd}} + \kappa_{eg}}\right\}\|\nabla f(x_k)\|, \tag{4.9}$$

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

$$f(x_k + s_k) - f(x_k) \leq -C_1 \|\nabla f(x_k)\| \delta_k, \tag{4.10}$$

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

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

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

Since condition (4.9) implies that $\|\nabla f(x_k)\| \geq \max\left\{ \kappa_{bhm} + \kappa_{eg}, \dfrac{8\kappa_{ef}}{\kappa_{fcd}} + \kappa_{eg} \right\} \delta_k$, we have

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

Hence, the conditions of Lemma 4.4.4 hold and we have

$$f(x_k + s_k) - f(x_k) \leq -\frac{\kappa_{fcd}}{4} \|g_k\| \delta_k. \tag{4.11}$$

Since $\|g_k\| \geq \|\nabla f(x)\| - \kappa_{eg}\delta_k$ in which δ_k satisfies (4.9), we also have

$$\|g_k\| \geq \max\left\{ \frac{\kappa_{bhm}}{\kappa_{bhm} + \kappa_{eg}}, \frac{8\kappa_{ef}}{8\kappa_{ef} + \kappa_{fcd}\kappa_{eg}} \right\} \|\nabla f(x_k)\|. \tag{4.12}$$

Combining (4.11) and (4.12) yields (4.10). □

We now state and prove the lemma that states that, in the presence of sufficient accuracy of the estimates, if the model is fully-linear, the trust-region radius is sufficiently small relatively to the size of the model gradient, a successful step is guaranteed.

Lemma 4.4.6. *Under Assumptions 4.4.1 and 4.4.2, 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*

$$\delta_k \leq \min\left\{\frac{1}{\kappa_{bhm}}, \frac{1}{\eta_2}, \frac{\kappa_{fcd}(1-\eta_1)}{8\kappa_{ef}}\right\}\|g_k\|, \tag{4.13}$$

then the k-th iteration is successful.

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

$$m_k(x_k) - m_k(x_k + s_k) \geq \frac{\kappa_{fcd}}{2}\|g_k\| \min\left\{\frac{\|g_k\|}{\kappa_{bhm}}, \delta_k\right\} = \frac{\kappa_{fcd}}{2}\|g_k\|\delta_k. \tag{4.14}$$

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

$$|f(x_k) - m_k(x_k)| \leq \kappa_{ef}\delta_k^2, \text{ and} \tag{4.15}$$

$$|f(x_k + s_k) - m_k(x_k + s_k)| \leq \kappa_{ef}\delta_k^2. \tag{4.16}$$

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

$$|F_k^0 - f(x_k)| \leq \kappa_{ef}\delta_k^2, \text{ and } |F_k^s - f(x_k + s_k)| \leq \kappa_{ef}\delta_k^2. \tag{4.17}$$

Combining (4.14)-(4.17), 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 $\delta_k \leq \dfrac{\kappa_{fcd}(1-\eta_1)}{8\kappa_{ef}}\|g_k\|$ 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 ensures conditions on a successful step, which guarantee a decrease of the true objective function.

Lemma 4.4.7. *Under Assumptions 4.4.1 and 4.4.2, suppose that the estimates* $\{f_k^0, f_k^s\}$ *are* ϵ_F*-accurate with* $\epsilon_F < \dfrac{1}{4}\eta_1\eta_2\kappa_{fcd}\min\left\{\dfrac{\eta_2}{\kappa_{bhm}}, 1\right\}$. *If a trial step* s_k *is accepted as a successful step, then the improvement in* f *is bounded below as follows*

$$f(x_{k+1}) - f(x_k) \leq -C_2\delta_k^2, \tag{4.18}$$

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

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 9 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 4.4.8. *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 9 are, respectively, α-probabilistically $(\kappa_{ef}, \kappa_{eg})$- fully-linear and β-probabilistically ϵ_F-accurate. Moreover, events I_k and J_k are independent of each other, conditioned on $\mathcal{F}_{k-1}^{M\cdot F}$.*

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

Theorem 4.4.9. *Let Assumptions 4.4.1 and 4.4.2 be satisfied and assume that $\eta_2 > \kappa_{bhm}$ in Algorithm 9. Then $\alpha, \beta, \epsilon_F$ can be chosen so that, if Assumption 4.4.8 holds for these values, then the sequence of trust-region radii, $\{\Delta_k\}$, generated by Algorithm 9 satisfies*

$$\sum_{k=0}^{\infty} \Delta_k^2 < \infty \tag{4.19}$$

almost surely.

Remark 4.4.10. *The exact conditions on α, β and ϵ_F will be shown in Lemma 4.4.11.*

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 [51], but analysis itself is different. The overall goal is to show that there exists a constant $\sigma > 0$ such that for all k,

$$E[\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}^{M\cdot F}] \leq -\sigma\Delta_k^2 < 0. \tag{4.20}$$

Since f is bounded from below and $\Delta_K > 0$, then Φ_k is bounded from below for all k and hence if (4.20) holds on every iteration, then by summing (4.20) from 1 to ∞ and taking expectations on both sides we can conclude that (4.19) holds with probability 1. Hence, to prove the theorem we need to show that (4.20) 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 (4.20) 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 9. Note that on all successful iterations, $x_{k+1} = x_k + s_k$ and $\delta_{k+1} = \gamma\delta_k$ with $\gamma > 1$, hence

$$\phi_{k+1} - \phi_k = \nu(f(x_{k+1}) - f(x_k)) + (1-\nu)(\gamma^2 - 1)\delta_k^2. \tag{4.21}$$

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

$$\phi_{k+1} - \phi_k = (1-\nu)(\frac{1}{\gamma^2} - 1)\delta_k^2 \equiv b_1 < 0. \tag{4.22}$$

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 (4.4) and (4.6) 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 that the decrease on an unsuccessful iteration (which is equal to b_1, according to (4.22)). When the model and the estimates are both bad, an increase in ϕ_k may occur. This increase is bounded by $O(\delta_k^2)$ unless $\|\nabla f(x_k)\| \geq \zeta\delta_k$. In this last case the increase in ϕ_k may be proportional to $\|\nabla f(x_k)\|\delta_k$, however, the good model and good estimates guarantee a successful iteration, which in turns provides decrease in ϕ_k which is proportional to $\|\nabla f(x_k)\|\delta_k$. Hence by choosing sufficiently large values for α and β 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

$$\epsilon_F \leq \kappa_{ef}. \tag{4.23}$$

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 (4.9) in Lemma 4.4.5 holds. Therefore, the trial step s_k leads to a decrease in f as in (4.10).

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 (4.13) in Lemma 4.4.6 holds. Hence, iteration k is successful, i.e. $x_{k+1} = x_k + s_k$ and $\delta_{k+1} = \gamma\delta_k$.

Combining (4.10) and (4.21), we get

$$\phi_{k+1} - \phi_k \leq -\nu C_1\|\nabla f(x_k)\|\delta_k + (1-\nu)(\gamma^2 - 1)\delta_k^2 \equiv b_2, \tag{4.24}$$

with C_1 defined in Lemma 4.4.5. Since $\|\nabla f(x_k)\| \geq \zeta\delta_k$ we have

$$b_2 \leq [-\nu C_1\zeta + (1-\nu)(\gamma^2 - 1)]\delta_k^2 < 0, \tag{4.25}$$

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

$$\frac{\nu}{1-\nu} > \frac{\gamma^2 - 1}{\zeta C_1}. \tag{4.26}$$

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 4.4.5 still holds, that is s_k yields a sufficient decrease in f, hence, if the iteration is successful, we obtain (4.24) and (4.25). However, the step can be erroneously rejected, because of inaccurate probabilistic estimates, in which case we have an unsuccessful iteration and (4.22) holds. By choosing $\nu \in (0,1)$ large enough so that

$$\frac{\nu}{1-\nu} > \frac{\gamma^2 - 1/\gamma^2}{\zeta C_1}, \tag{4.27}$$

we ensure that the right hand side of (4.25) is strictly smaller than the right hand side of (4.22) and therefore, (4.22) 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

$$\epsilon_F < \frac{1}{8}\eta_1\eta_2\kappa_{fcd}. \tag{4.28}$$

In this case, iteration k can be either successful or unsuccessful. In the unsuccessful case (4.22) holds. When the iteration is successful, since the estimates are ϵ_F-accurate and (4.28) holds then by Lemma 4.4.7 (4.18) holds with $C_2 \geq \frac{1}{4}\eta_1\eta_2\kappa_{fcd}$. Hence, in this case we have

$$\phi_{k+1} - \phi_k \leq [-\nu C_2 + (1-\nu)(\gamma^2 - 1)]\delta_k^2. \tag{4.29}$$

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

$$\frac{\nu}{1-\nu} \geq \frac{\gamma^2 - 1/\gamma^2}{C_2} \tag{4.30}$$

we have that, as in case (b), (4.22) 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,

Table 4.1: A summary of the decrease in ϕ_k in four random outcomes in Case 1.

Scenario i	Probability P_i	Successful or unsuccessful	$D_i : \phi_{k+1} - \phi_k$
(a)	$\alpha\beta$	successful	$b_2 < 0$
(b)	$\alpha(1-\beta)$	both possible	$b_1 < 0$
(c)	$(1-\alpha)\beta$	both possible	$b_1 < 0$
(d)	$(1-\alpha)(1-\beta)$	both possible	$b_3 > 0$

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

$$\phi_{k+1} - \phi_k \leq \nu C_3\|\nabla f(x_k)\|\delta_k + (1-\nu)(\gamma^2 - 1)\delta_k^2 \equiv b_3. \tag{4.31}$$

Table 4.1 summarizes the analysis in Case 1 where $\|\nabla f(x_k)\| \geq \zeta\delta_k$. It is observed that only in scenario (a), a successful iteration is produced with probability one. In other random outcomes, both decisions can be randomly made, but the worse decrease in ϕ_k is upper bounded.

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 (4.24), 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 (4.22). Finally we note that $b_1 > b_2$ due to our choice of ν.

Hence, we can combine (4.22), (4.24), (4.29) and (4.31) 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

$$\frac{\alpha\beta}{(1-\alpha)(1-\beta)} \geq \frac{\frac{2(1-\nu)}{\nu} \cdot \frac{\gamma^2-1}{\zeta} + C_3}{C_1} \tag{4.32}$$

which implies

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

we have

$$E[\Phi_{k+1}-\Phi_k|\mathcal{F}_{k-1}^{M\cdot F}, \{\|\nabla f(X_k)\| \geq \zeta\Delta_k\}] \leq -[C_1\alpha\beta-(1-\alpha)(1-\beta)C_3]\nu\|\nabla f(X_k)\|\Delta_k. \tag{4.33}$$

and

$$E[\Phi_{k+1} - \Phi_k|\mathcal{F}_{k-1}^{M\cdot F}, \{\|\nabla f(X_k)\| \geq \zeta\Delta_k\}] \leq -(1-\nu)(\gamma^2-1)\Delta_k^2. \tag{4.34}$$

For the purposes of this lemma and the lim inf-type convergence result, which will follow, bound (4.34) is sufficient. We will use bound (4.33) 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 (4.22) 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, (4.22) 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 (4.22).

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

On successful iterations, (4.22) 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

$$\eta_2 \geq \frac{6\kappa_{ef}}{\kappa_{fcd}}. \tag{4.35}$$

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

$$\phi_{k+1} - \phi_k \leq [-\nu\kappa_{ef} + (1 - \nu)(\gamma^2 - 1)] \delta_k^2. \tag{4.36}$$

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

$$\frac{\nu}{1-\nu} > \frac{\gamma^2 - 1/\gamma^2}{\kappa_{ef}}, \tag{4.37}$$

we ensure that right hand side of (4.36) is strictly smaller than that of (4.22), hence (4.22) holds, whether the iteration is successful or not.

Remark: η_2 may need to be a relatively large constant to satisfy (4.35). 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 more complex. In this chapter we choose to stay with the simplest version, but keeping in mind that condition (4.35) 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 $\nabla f(x)$.

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

Table 4.2: A summary of the decrease in ϕ_k in four random outcomes in Case 2.

Scenario	Probability	Successful or not	$D_i : \phi_{k+1} - \phi_k$
a	$\alpha\beta$	both possible	$b_1 < 0$
b	$\alpha(1-\beta)$	both possible	$b_1 < 0$
c	$(1-\alpha)\beta$	both possible	$b_1 < 0$
d	$(1-\alpha)(1-\beta)$	both possible	$b_4 > 0$

Hence, the change in function ϕ is

$$\phi_{k+1} - \phi_k \leq [\nu C_3 \zeta + (1-\nu)(\gamma^2 - 1)]\delta_k^2 \equiv b_4. \tag{4.38}$$

Table 4.2 summarizes the four random scenarios in Case 2. We are now ready to bound the expectation of $\phi_{k+1} - \phi_k$ as we did in Case 1, except that in case Case 2 we simply combine (4.38), which holds with probability at most $(1-\alpha)(1-\beta)$ and (4.22) 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,

$$\frac{1 - (1-\alpha)(1-\beta)(1+\gamma^2)}{(1-\alpha)(1-\beta)} \geq \frac{2\nu\gamma^2 C_3 \zeta}{(1-\nu)(\gamma^2 - 1)}, \tag{4.39}$$

that is,

$$(1-\alpha)(1-\beta) \leq \frac{\gamma^2 - 1}{\gamma^4 - 1 + 2\gamma^2 C_3 \zeta \cdot \frac{\nu}{1-\nu}}, \tag{4.40}$$

then

$$E[\Phi_{k+1} - \Phi_k | \mathcal{F}_{k-1}^{M \cdot F}, \{\|\nabla f(X_k)\| < \zeta \Delta_k\}] \leq -\nu C_3 \zeta \Delta_k^2. \tag{4.41}$$

In conclusion, combining (4.34) and (4.41), 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 4.4.9 to ensure that the Lemma holds, we state the following additional lemma.

Lemma 4.4.11. *Let all assumptions of Theorem 4.4.9 hold. Then the statement of the theorem holds if the parameters are chosen to satisfy the following conditions: if the trust region acceptance parameters satisfy* $\eta_1 \geq \dfrac{1}{2}$,

$$\eta_2 \;\; \geq \;\; \max\left\{\kappa_{bhm}, \frac{8\kappa_{ef}}{\kappa_{fcd}(1-\eta_1)}\right\}, \tag{4.42}$$

and the accuracy parameter of the estimates satisfies

$$\epsilon_F \;\; \leq \;\; \min\left\{\kappa_{ef}, \frac{1}{8}\eta_1\eta_2\kappa_{fcd}\right\}, \tag{4.43}$$

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

these two conditions

$$\frac{\alpha\beta}{(1-\alpha)(1-\beta)} \geq \frac{1+\frac{8(\gamma^2-1)+3L_1}{2\zeta}}{C_1}, \text{ and} \tag{4.44}$$

$$(1-\alpha)(1-\beta) \leq \frac{\gamma^2-1}{\gamma^4-1+\gamma^4\left(3L_1+2\zeta\right)\cdot\max\left\{\frac{1}{\zeta C_1},\frac{1}{\kappa_{ef}},\frac{1}{2}\right\}}, \tag{4.45}$$

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 4.4.9. Condition (4.43) follows from conditions (4.23) and (4.28).

Condition (4.42) implies that $\eta_2 \geq \kappa_{bhm}$ and condition (4.35) hold, since $\eta_1 < 1$. Moreover, we now have

$$\zeta \geq \kappa_{eg} + \max\left\{\eta_2, \frac{8\kappa_{ef}}{\kappa_{fcd}(1-\eta_1)}\right\} \geq \kappa_{eg}+\eta_2. \tag{4.46}$$

Under condition (4.43) we have, using $\eta_1 \geq 1/2$,

$$C_2 = \frac{1}{2}\eta_1\eta_2\kappa_{fcd} - 2\epsilon_F \geq \frac{1}{4}\eta_1\eta_2\kappa_{fcd} \geq 2\eta_1\kappa_{ef} \geq \kappa_{ef}. \tag{4.47}$$

Hence, combining conditions (4.26), (4.27), (4.30) and (4.37) with (4.46) and (4.47) with $\gamma > 1$, we derive

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

Now we are ready to derive conditions on α and β which imply conditions (4.32)

and (4.40) 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 (4.44) 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 flowing from $\frac{1-\nu}{\nu} < 2$. Hence (4.32) holds.

Condition (4.40) with $C_3 = 1 + \frac{3L_1}{2\zeta}$ and

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

can be rewritten as

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

Hence, we obtain (4.45).

□

Clearly, choosing α and β sufficiently close to 1 will satisfy this condition. Let us examine some reasonable examples of constants and the corresponding choice of α and β.

Remark 4.4.12. *Notes that if $\beta = 0$, then $\Delta_k \to 0$ for any value of α, which is the case shown in [9].*

4.4.3 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 [9], the key difference being the assumption on the function estimates that are needed to ensure that a good step gets accepted by Algorithm 9.

Theorem 4.4.13. *Suppose that the model sequence $\{M_k\}$ is α-probabilistically $(\kappa_{ef}, \kappa_{eg})$ fully linear and the estimate sequence $\{F_k^0, F_k^s\}$ is β-probabilistically ϵ_F-accurate. Let assumptions of Lemmas 4.4.9 and 4.4.11 hold. Then for $\{X_k\}$ - the sequence of random iterates generated by Algorithm 9, almost surely,*

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

Proof. We prove this result by contradiction conditioned on the almost sure event

$$\Delta_k \to 0.$$

Let us, hence 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 $k \geq k_0$,

$$\delta_k < b := \min\left\{ \frac{\epsilon'}{2\kappa_{eg}}, \frac{\epsilon'}{2\kappa_{bhm}}, \frac{\kappa_{fcd}(1-\eta_1)\epsilon'}{16\kappa_{ef}}, \frac{\epsilon'}{2\eta_2}, \frac{\delta_{\max}}{\gamma} \right\}. \tag{4.48}$$

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 realization occurs 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 Definition 4.3.2 and Definition 4.3.4. 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 (4.48) holds we have exactly the same situation as in Case 1(a) in the proof of Lemma 4.4.9. In other words we can apply Lemmas 4.4.5 and 4.4.6 to conclude that the k-th iteration is successful. hence, the trust-region radius is increased. In particular, since $\delta_k \leq \dfrac{\delta_{\max}}{\gamma}$, $\delta_{k+1} = \gamma\delta_k$. Consequently, $r_{k+1} = r_k + 1$.

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

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

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 variable $\mathbf{1}_{I_k}$ 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.

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

and W_k is a submartingale since

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

where $\mathcal{F}_{k-1}^{S\cdot T} = \sigma(\mathbf{1}_{S_0},\cdots,\mathbf{1}_{S_{k-1}}) \cap \sigma(\mathbf{1}_{T_0},\cdots,\mathbf{1}_{T_{k-1}})$ and 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 4.4.3, 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.

□

4.4.4 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 [9], 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 [9] function values could only decrease or remain

unchanged.

Lemma 4.4.14. *Let $\{X_k\}$ and $\{\Delta_k\}$ be sequences of random iterates and random trust-region radii generated by Algorithm 9. 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 4.4.9 we know that

$$\sum \Delta_k^2 < \infty$$

and hence $\Delta_k \to 0$ almost surely. For each realization of Algorithm 9 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 in Theorem 4.4.9. 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 4.4.9 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 (4.33) 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 4.4.15. *Suppose that the model sequence $\{M_k\}$ is probabilistically $(\kappa_{eg}, \kappa_{ef})$-fully linear for some positive constants κ_{eg} and κ_{ef}. Let $\{X_k\}$ be a sequence of random iterates generated by Algorithm 9. 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 [9] hence we will not present the proof here. The key idea of the proof is to show that is the theorem does not hold, then with positive probability

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

with K_ϵ defined as in Lemma 4.4.14. 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 Lemma 4.4.14, we can conclude that the statement of the theorem holds.

4.5 Constructing Probabilistic Models [1]

In the previous section, we have proven the lim-type convergence of Algorithm 9 under fairly typical assumptions about the objective function and Assumption 4.4.8 on the sequence of models and estimates. The purpose of this section is to demonstrate how this latter assumption might be satisfied. In particular, we will use results of learning theory to obtain a provably α-probabilistically fully-linear sequence of models, for whatever α is sufficient by the result of Theorem 4.4.11.

Essentially, we need to produce a sequence of models $\{M_k\}$, each of which satisfies the fully linear definition in Definition 4.3.2, for the trust-region defined by the sequence of random variables $\{B(X_k, \Delta_k)\}$ with probability at least α. Moreover, this probability bound must hold for the k-th model M_k independently of the history $\{X_i\}, \{\Delta_i\}, \{M_i\}$, for $i = 0, 1, \ldots, k-1$.

Let us drop iteration subscripts for clarity of presentation and consider for some realization a fixed point x^0 and a trust region radius δ. Consider sampling points from $B(x^0, \delta)$ according to some distribution D and assume that D is continuous uniform on $B(x^0, \delta)$. Denote a single point being drawn from this distribution with the notation $x \sim D$.

Let us consider a class of models M intended to approximate the stochastic objective $\tilde{f}(x)$ on $B(x^0, \delta)$. Denote the distribution of the noise ϵ from the definition $\tilde{f}(x) = f(x) + \epsilon$ by $\mathcal{E}$. An intuitive way to choose a "best" approximating model from M is to consider the model $m^* \in M$ defined as

[1] THIS IS SUMMARIZED VERSION OF A SECTION IN THE TECHNICAL REPORT [21] COAUTHORED BY PROF. KATYA SCHEINBERG AND MATT MENICKELLY.

$$m^* := \arg\min_{m \in M} \tilde{L}(m), \tag{4.49}$$

where

$$\tilde{L}(m) := \mathbb{E}_{(x,\tilde{f}(x))\sim D\times\mathcal{E}}\left[(m(x) - \tilde{f}(x))^2\right].$$

$\tilde{L}$ decomposes into the sum of an expectation related to the deterministic $f(x)$ and the variance of the noise:

$$\begin{aligned}
\tilde{L}(m) &= \mathbb{E}_{(x,\tilde{f}(x))\sim D\times\mathcal{E}}\left[(m(x) - f(x) - \epsilon)^2\right] \\
&= \mathbb{E}_{(x,\tilde{f}(x))\sim D\times\mathcal{E}}\left[(m(x) - f(x))^2 - 2\epsilon(m(x) - f(x)) + \epsilon^2\right] \\
&= \mathbb{E}_{x\sim D}\left[(m(x) - f(x))^2\right] + \sigma^2 := L(m) + \sigma^2.
\end{aligned}$$

In the most general cases where we assume that noise is uncontrollable, i.e. σ^2 is a constant. Thus, minimizing $\tilde{L}(m)$ is equivalent to minimizing $L(m)$.

Generally, there is no tractable way to compute the solution to the optimization problem in (4.49), given the presence of the expectation operator and the unknown distribution $\mathcal{E}$. For this reason, we turn to results of learning theory [78] and consider a more tractable model $m^p(x)$, referred as the empirical risk minimizing model, defined as

$$m^p := \arg\min_{m \in M} \frac{1}{p}\sum_{i=1}^{p}(m(y^i) - \tilde{f}(y^i))^2, \tag{4.50}$$

where $y^1, \ldots, y^p$ are points sampled from $B(x^0, \delta)$ according to the distribution D.

Suppose M contains the first-order Taylor model $\hat{m}(x) = f(x^0) + \nabla f(x^0)(x - x^0)$. Then, for that $\hat{m}(x)$, there exists some constant κ so that $|\hat{m}(x) - f(x)| \leq \kappa\delta^2$ for all $x \in B(x^0, \delta)$. Thus,

$$\tilde{L}(\hat{m}) = L(\hat{m}) + \sigma^2 = \mathbb{E}_{x\sim D}\left[(\hat{m}(x) - f(x))^2\right] + \sigma^2 \leq \kappa^2\delta^4 + \sigma^2 \tag{4.51}$$

Since $\hat{m} \in M$, the optimal m^* must also satisfy the bound $\tilde{L}(m^*) \leq \kappa^2\delta^4 + \sigma^2$. By Jensen's inequality, (4.49) and (4.51) imply

$$\mathbb{E}_{(x,\tilde{f}(x))\sim D\times\mathcal{E}}\left[|m^*(x) - \tilde{f}(x)|\right] \leq \kappa\delta^2 + \sigma^2 \tag{4.52}$$

So, if the noise satisfy $\sigma^2 \in \mathcal{O}(\delta^4)$, then $m^*(x)$ is fully linear in expectation, that is:

$$\mathbb{E}_{x\sim D}\left[|m^*(x) - f(x)|\right] \in \mathcal{O}(\delta^2) \tag{4.53}$$

We are now able to show Theorem 4.5.1, proved in [21].

Theorem 4.5.1. *Assume that an estimate f^0 of $f(x^0)$ is available and $|f^0 - f(x^0)| \leq D'\delta$ for some positive constant D' (independent of x^0 and δ) with probability $1-\gamma'$. If $\sigma^2 \in \mathcal{O}(\delta^2)$ and the number of points randomly sampled from $B(x^0,\delta)$ satisfies $p \in \mathcal{O}(1/\delta^2)$, then there exists a positive constant κ_{Eef} such that, with probability $1-\gamma-\gamma'$,*

$$\mathbb{E}[(m^p(x) - f(x))^2] \leq \kappa_{Eef}\delta^4. \tag{4.54}$$

The main message is that, if the noise can be controlled so that $\sigma^2 \in \mathcal{O}(\delta^2)$, we can construct a model m^p that (4.54) by ensuring $p \in \mathcal{O}(1/\delta^2)$.

Obtaining Fully-linear Models

Now we use a Markov inequality argument to show that a model m^p that satisfies the sufficient condition of Theorem 4.5.1 also satisfies the definition of a fully-linear model in Definition 4.3.2.

Lemma 4.5.2. [2] *Consider a random model $m(x)$. If there exists a positive constant*

[2]This lemma is proven by me, originally in the appendix. Though it is not the main contribution

κ_{Eef} *such that for all* $x \in B(x^0, \delta)$,

$$\mathbb{E}[m(x) - f(x)] \leq \kappa_{Eef}\delta^2, \tag{4.55}$$

with probability $1 - \lambda$, *then* $m(x)$ *is* κ*-fully linear with probability* α, *where* α *and* $\kappa = \{\kappa_{ef}, \kappa_{eg}\}$ *are given by*

$$\begin{aligned} \alpha &= (1-\lambda)\prod_{i=0}^{n}\left(1 - \frac{\kappa_{Eef}}{\kappa_{ef_i}}\right), \\ \kappa_{ef} &= \sum_{i=1}^{n}\alpha_i(\kappa_{ef_i} + \kappa_{et_i}) + (n+1)\kappa_{ef_0} + \kappa_{et}, \\ \kappa_{eg} &= L_1 + 2\kappa_{ef} + \kappa_{etb}. \end{aligned}$$

Proof. **Part I:** We first show that there exist κ_{ef} such that

$$P_0 = P[|f(x) - m(x)| \leq \kappa_{ef}\delta^2, \forall x \in B(0, \delta)] \geq \alpha_{ef}.$$

Consider the first-order Taylor model $m_t(x) = \nabla f(x^0)^\top x + f(x^0)$ of $f(x)$ on $B(x^0, \delta)$. For an arbitrary point $x \in B(x^0, \delta)$, we can express

$$\begin{aligned} |m(x) - f(x)| &\leq |m(x) - m_t(x)| + |m_t(x) - f(x)| \\ &= |m(x) - m_t(x)| + \kappa_{et}\delta^2 \end{aligned} \tag{4.56}$$

for some fixed constant κ_{et} independent of δ.

Now let us bound $|m(x) - m_t(x)|$. Consider the n points on the surface of $B(x^0, \delta)$ defined by δe_i for $i \in 1 \ldots n$, where e_i is the ith elementary vector of $\mathbb{R}^n$. From a

of Matt's model constructing section, it seems appropriate to put this here. I will delete this footnote after your review.

simple application of Markov's Inequality to (4.55) on the ith point, we get that,

$$P(|m(\delta e_i) - f(\delta e_i)| < \kappa_{ef_i}\delta^2) > 1 - \frac{\kappa_{Eef}}{\kappa_{ef_i}},$$

for a constant κ_{ef_i} that can be made arbitrarily large and is independent of δ. Likewise, at the center of the ball $B(x^0, \delta)$, we can guarantee,

$$P(|m(x^0) - f(x^0)| < \kappa_{ef_0}\delta^2) > 1 - \frac{\kappa_{Eef}}{\kappa_{ef_0}}.$$

For each of the surface points δe_i, we have, also with probability at least $1 - \frac{\kappa_{Eef}}{\kappa_{ef_i}}$,

$$|m(\delta e_i) - m_t(\delta e_i)| \leq |m(\delta e_i) - f(\delta e_i)| + |f(\delta e_i) - m_t(\delta e_i)| < (\kappa_{ef_i} + \kappa_{et_i})\delta^2,$$

where the error term κ_{et_i} is independent of δ. Note that, via the reverse triangle equality and using the expanded notation of $m_t(x)$ and $m(x)$, we get

$$|(w - \nabla f(x^0))^\top \delta e_i| < (\kappa_{ef_i} + \kappa_{et_i})\delta^2 + |\beta - f(x^0)|.$$

Any arbitrary point $x \in B(x^0, \delta)$ is uniquely expressible as $\sum_{i=1}^{n} \alpha_i \delta e_i$ for some set of α_i. Using this linear combination, we can derive

$$\begin{aligned}
|m(x) - m_t(x)| &\leq \left|\sum_{i=1}^{n} \alpha_i (w - \nabla f(x^0))^\top \delta e_i\right| + |\beta - f(x^0)| \\
&\leq \sum_{i=1}^{n} \alpha_i |(w - \nabla f(x^0))^\top \delta e_i| + |\beta - f(x^0)| \\
&< \sum_{i=1}^{n} \alpha_i (\kappa_{ef_i} + \kappa_{et_i})\delta^2 + (n+1)|\beta - f(x^0)| \\
&< \sum_{i=1}^{n} \alpha_i (\kappa_{ef_i} + \kappa_{et_i})\delta^2 + (n+1)\kappa_{ef_0}\delta^2.
\end{aligned}$$

Since each of the bounds in the last inequality holds with a respective and independent probability, we get that this bound on $|m(x)-m_t(x)|$ holds with probability $\prod_{i=0}^{n}\left(1-\frac{\kappa_{Eef}}{\kappa_{ef_i}}\right)$, provided that $\mathbb{E}[|m(x)-f(x)|] \leq \kappa_{Eef}\delta^2$ happens with probability $1-\lambda$.

Hence, by Bayes' formula, we have

$$P_0 = P(|f(x)-m(x)| \leq \kappa_{ef}\delta^2,\, \forall x \in B(x^0,\delta)) \geq \alpha_{ef}, \tag{4.57}$$

where $\kappa_{ef} = \sum_{i=1}^{n}\alpha_i(\kappa_{ef_i}+\kappa_{et_i}) + (n+1)\kappa_{ef_0} + \kappa_{et}$, and $\alpha_{ef} = (1-\lambda)\prod_{i=0}^{n}\left(1-\frac{\kappa_{Eef}}{\kappa_{ef_i}}\right)$.

Part II: We next show that there exists κ_{eg} such that

$$P_1 = P(\|\nabla f(x)-\nabla m(x)\| \leq \kappa_{eg}\delta,\, \forall x \in B(x^0,\delta)) \geq \alpha_{eg}.$$

Using triangle inequality, we can obtain that, $\forall x \in B(x^0,B)$,

$$\|\nabla f(x)-\nabla m(x)\| \leq \|\nabla f(x)-\nabla f(x_0)\| + \|\nabla f(x_0)-\nabla m(x_0)\| + \|\nabla m(x_0)-\nabla m(x)\|.$$

With a linear model $m(x)$, the last term vanishes. The first term can be bounded using the gradient Lipschitz continuity assumption, i.e., $\|\nabla f(x)-\nabla f(x_0)\| \leq L_1(x-x^0) \leq L_1\delta$. Thus, what remains to bound is $\|\nabla f(x_0)-\nabla m(x_0)\|$.

Pick another surface point x_b so that $x_b - x^0 = \delta$. The Taylor expansions of the true function f and our model m of x_b at x^0 can be written as

$$\begin{aligned} f(x_b) &= f(x^0) + \nabla f(x^0)^\top(x_b-x^0) + O(\delta^2), \text{ and} \\ m(x_b) &= m(x^0) + \nabla m(x^0)^\top(x_b-x^0). \end{aligned} \tag{4.58}$$

Then, subtracting one from the other gives

$$\begin{aligned} f(x_b) - m(x_b) &= f(x_0) - m(x_0) + (\nabla f(x_0) - \nabla m(x_0))^\top (x_b - x^0) + \kappa_{etb}\delta^2 \\ &= f(x_0) - m(x_0) + (\nabla f(x_0) - \nabla m(x_0))^\top \dot{+} \kappa_{etb}\delta^2, \end{aligned}$$

where κ_{etb} is some positive constant representing the error from Taylor expansion of x_b at x^0. According to (4.57), we know that as long as the error in function value is bounded, the following naturally happen

$$|f(x_b) - m(x_b)| \leq \kappa_{ef}\delta^2, \text{ and } |f(x_0) - m(x_0)| \leq \kappa_{ef}\delta^2. \tag{4.59}$$

Combining (4.58) and (4.59), we get $\|\nabla f(x_0) - \nabla m(x_0)\| \leq (2\kappa_{ef} + \kappa_{etb})\delta$.Hence, conditioned on $|f(x) - m(x)| \leq \kappa_{ef}\delta^2$, $\forall x \in B(x^0, \delta)$

$$P_1 = P[\|\nabla f(x) - \nabla m(x)\| \leq \kappa_{eg}\delta, \, \forall x \in B(x^0, \delta)] = 1,$$

where $\kappa_{eg} = L_1 + 2\kappa_{ef} + \kappa_{etb}$.

Since these two events (model error and gradient error) are dependent, we can derive

$$P(m(x) \text{ is a } \kappa \text{ fully linear model of } f \text{ on } B(x^0, \delta)) \geq \alpha,$$

where $\alpha = \alpha_{ef} = (1-\lambda)\prod_{i=0}^{n}\left(1 - \frac{\kappa_{Eef}}{\kappa_{ef_i}}\right)$, $\kappa_{ef} = \sum_{i=1}^{n}\alpha_i(\kappa_{ef_i} + \kappa_{et_i}) + (n+1)\kappa_{ef_0} + \kappa_{et}$ and $\kappa_{eg} = L_1 + 2\kappa_{ef} + \kappa_{etb}$. □

This probabilistic bound has no dependence on the past history of Algorithm 9, and thus gives us exactly the α-probabilistically fully linear sequence $\{m_k\}$ that we need.

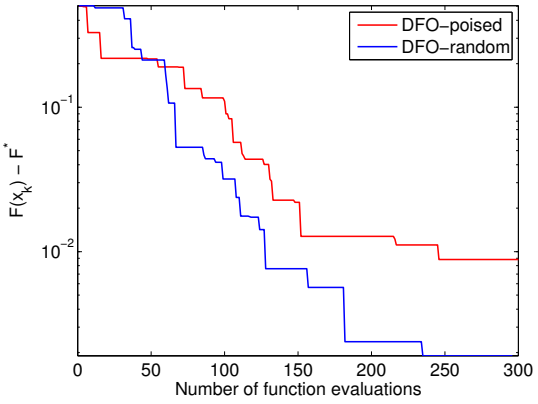
4.6 Computational Experiments

4.6.1 Results on Protein Alignment Problem

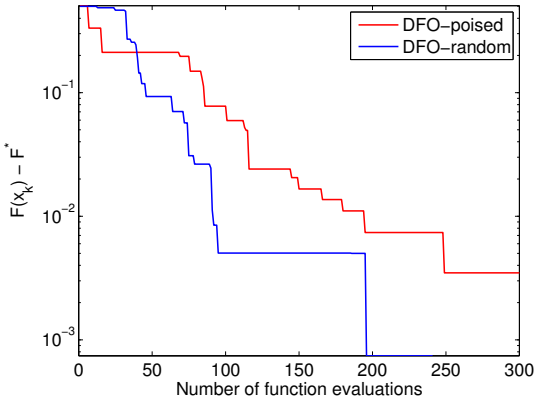
We test the performance of STORM on the protein alignment problem. An implementation of STORM, referred as DFO-random is compared with DFO-poised, a deterministic version of DFO used in [22, 36], where $2n + 1$ poised sample points to build quadratic models and compute the value at every point by averaging $\mathcal{O}(1/\delta_k^4)$ noisy values. The purpose of this comparison is to show the benefits of using a set of $O(1/\delta_k^2)$ many random points.

As different resolution values consumes different amount of runtime, this makes it different to make a fair comparison. This experiment considers a fixed resolution value, i.e., 0.5. To compare the relative performance of different versions of DFO, it might be hard to judge from the accuracy of the solution, as the function is noisy and the exact volume of intersection is unknown, expensive to evaluate in protein alignment. However, we can compare the runtime each algorithm takes to reach the state where the model reduction is too small comparing to the noise. As in general the accuracy is increased whenever this happens. If the runtime spent in each stage of increment is reduced, the overall runtime is thus improved.

Figure 4.4 is a preliminary test comparing DFO-poised and DFO-random, where DFO-random is merely a slightly randomized version of DFO-poised, executed in a sequential environment where parallelization is not permitted. The key difference between these two algorithms is that DFO-random uses least squares regression models and an initial random sample set, however, no estimates are computed, nor excess random points to evaluate at each iteration. Due to the randomness in the sample set and the noise, the algorithm is tested multiple times and the average performance



(a) 1a0j.SURF and 1b0e.SURF.



(b) 1aks.SURF and 1b0e.SURF.

Figure 4.4: Results on protein alignment problem. A plot showing the benefit of using a least squares regression model and an initial random sample set. Averaged over 10 trials. No parallel computation is needed.

is presented. It is demonstrated that using regression models yields more accurate solutions and faster convergence comparing to using interpolation models, on both pairs of protein alignments.

To specify a more detailed implementation of Algorithm 9, we need to define how the model function m_k and estimates f_k^0 and f_k^s are constructed, also how the trust region subproblem is solved to compute a trial step s_k. The following strategies are used.

- At iteration k, the model m_k is constructed using least squares regression on a random sample set of p_k points, where p_k is defined by

$$p_k = \min\{p_{\max}, \min\{2n+1, \left\lfloor \frac{1}{\delta_k^2} \right\rfloor\}\},$$

  and $p_{\max}$ is the maximum number of points allowed while building a model. These points are uniformly distributed over the interior of an n-dimensional hypersphere of radius δ_k with center at x_k.

- As the models are quadratic $m(x) = c + g^\top x + \frac{1}{2} x^\top H x$, we solve the empirical minimizing problem

$$\min_{c,g,H} \quad \sum_{i=1}^{p_k} (c + g^\top x_i + \frac{1}{2} x_i^\top H x_i - \tilde{f}(x_i))^2,$$

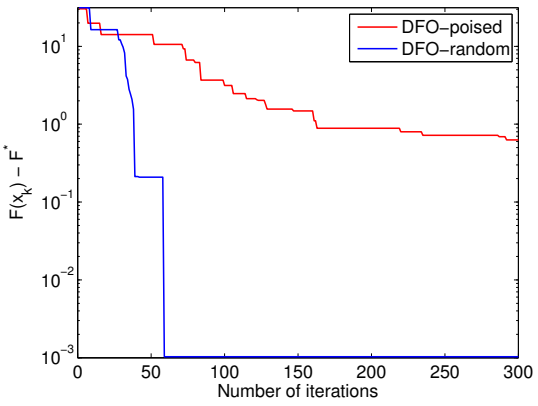
  Then, the model is minimized within the trust region to obtain s_k.

- The estimates f_k^0, f_k^s for $f(x_k)$, $f(x_k + s_k)$ respectively, are computed by repeatedly sampling $\tilde{f}$ at each point 10 times independently and computing the average.

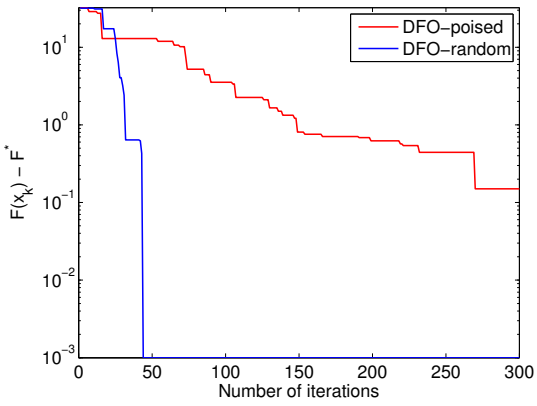
- Lastly, the parameter values used are: $p_{\max} = 100, \max Y = 100; \delta_0 = 1, \eta_1 = 0.75, \eta_2 = 0.5, \gamma_1 = 1.5, \gamma_2 = 0.8, \epsilon_\delta = 10^{-3}$.

While DFO-poised generates one point per iteration in a sequential way, DFO-random uses random points that are generated independently from each other. Thus it allows the use of parallel computation to evaluate multiple function values at once. Moreover, as the construction of probabilistic estimates is achieved by averaging, this step can also be performed in a similar way with parallel machines.

Figure 4.5 compares the relative performance of DFO-poised and DFO-random in a parallel environment. Although the prototype implementation above is inefficient in the sense that it does not reuse previous evaluated points and builds every model with an entirely new set of points, the runtime does not in fact increase dramatically thanks to the parallelization. For both algorithms, the runtime in each iteration is equivalent to that of one function evaluation. It can be observed in Figure 4.5 that DFO-random finds a solution with a 10^{-3} error within around 50 iterations on both biological instances, while DFO-poised proceeds in a much slower pace, producing solutions with higher errors ($10^{-1} \sim 1$) after 300 iterations. Hence, the flexibility of



(a) 1a0j.SURF and 1b0e.SURF.



(b) 1aks.SURF and 1b0e.SURF.

Figure 4.5: A plot showing the additional benefits of using more accurate estimates at $x_k, x_k + s_k$ and using a set of $O(1/\delta_k^2)$ many random points at each iteration to construct a regression model. Averaged over 10 trials. Multiple VASP function values computed at the same time in parallel.

using parallelization gives DFO-random the advantage to yield accurate solutions in a timely manner in the presence of noise.

4.6.2 Performance Profiles

Algorithms. We compare two implementations of STORM (DFO-random and DFO-control) with DFO-poised and two stochastic approximation (SA) methods:

- DFO-random: built least-squares regression models using $\mathcal{O}(1/\delta_k^2)$ random points.;
- DFO-control: control noise so that $\sigma \in \mathcal{O}(\delta_k^2)$ and built LS models using $\mathcal{O}(1/\delta_k^2)$ random points;
- DFO-poised: use $2n + 1$ poised sample points to build quadratic models and compute the value at every point by averaging $\mathcal{O}(1/\delta_k^4)$ noisy values (this averaging approach is also used in [36]);

- KW [48]: Finite Differences Stochastic Approximation (FDSA);
- SPSA [77]: Simultaneous Perturbation Stochastic Approximation.

Two SA algorithms follow the same updating strategy at each iteration,

$$x_{k+1} = x_k + \alpha_k \hat{g}_k(x_k),$$

where α_k denotes the step length while the gradient approximation $\hat{g}_k(x_k)$ is calculated differently:

$$\begin{aligned} \text{KW}: \quad (\hat{g}_k(x_k))_i &= \frac{\tilde{f}(x_k + c_k e_i) - \tilde{f}(x_k - c_k e_i)}{2c_k}, \\ \text{SPSA}: \quad (\hat{g}_k(x_k))_i &= \frac{\tilde{f}(x_k + c_k \Delta_k) - \tilde{f}(x_k - c_k \Delta_k)}{2c_k (\Delta_k)_i}. \end{aligned}$$

Notice that KW perturbs only one direction at a time and requires $2n$ evaluations of $\tilde{f}$ for each $\hat{g}_k$. Clearly, when n is large, this estimator loses efficiency. SPSA estimator, on the other hand, disturbs all directions at the same time, thus only 2 evaluations of $\tilde{f}$ for each $\hat{g}_k$ is needed, regardless of the dimension of the optimization problem. Since the numerator is identical in all n components, SPSA is n times fewer function evaluations than KW, which makes it a lot more efficient. Moreover, SPSA is a descent method capable of finding global minima.

Benchmarking Suit. The test set consists of 53 stochastic problems from [57]. The noise is additive and it follows a normal distribution with zero mean and 0.01 variance ($\sigma = 0.1$). The dimension of the problems ranges from 2 to 12. Each method was given at most 1500 function evaluations to solve each problem. Since problems are stochastic in nature, each method was given 10 attempts to solve each of the 53 problems, and the function values are averaged across these 10 runs.

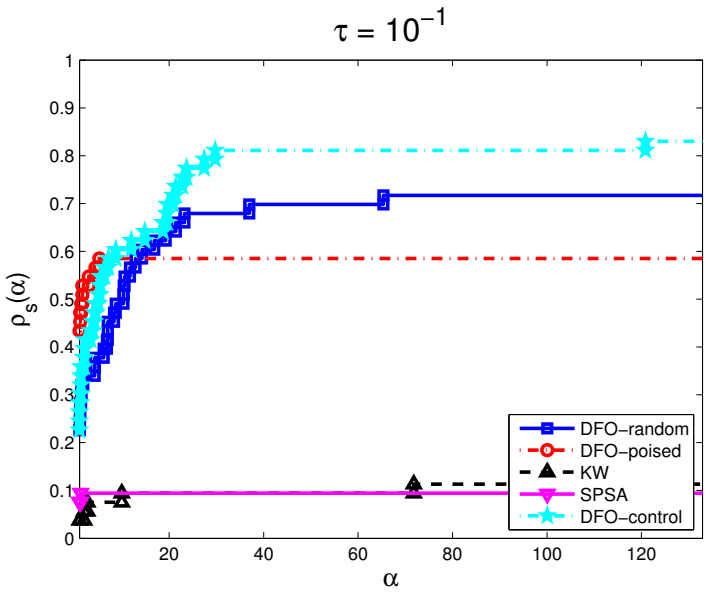


Figure 4.6: Preliminary Results on 53 Noisy Problems with $\sigma = 0.1$; performance profile threshold is $\tau = 10^{-1}$.

Results. Figure 4.6 shows that our prototypes of STORM all outperforms KW and SPSA. While DFO-poised is the fastest on almost 42% of the problems, it solving the least number of problems (only 60%) among other STORM variations. This is due to the presence of noise - although the set is poised, the model becomes poor when the noise gets relatively significant comparing to δ_k. DFO-control verifies our theory that if the noise can be controlled so that $\sigma \in O(1/\delta^2)$, probabilistically accurate models can be built with only $O(1/\delta^2)$ points. The problems that DFO random does not solve and DFO-control does are the problems where noise becomes an issue. One key observation is that DFO-random also solves more problems (70%) than DFO-poised. It shows that in practice, $O(1/\delta^2)$ many points might suffice without the control of noise.

4.7 Conclusions and Future Work

In this chapter we have proposed a stochastic DFO algorithm, STORM, for solving noisy unconstrained black-box optimization problems. This algorithm following a traditional trust region framework while contributing two new features. First, we consider probabilistic models that are constructed based on random sample points. Second, at each iteration, we use two random estimates of the true function values at the current point and a trial point. Theoretically, we prove that, if the model sequence is probabilistically fully-linear and the estimate sequence is probabilistically accurate, a lim inf-type and a lim-type convergence result can be shown. We also provide analysis on the number of random points that are needed to construct these probabilistic models. The theory indicates that, if the noise can be controlled so that $\sigma^2 \in \mathcal{O}(\delta^2)$, then we only need $p \in \mathcal{O}(1/\delta^2)$ many random points to construct probabilistically fully-linear models. However, our experiments suggest that $O(1/\delta^2)$ many points might suffice without the control of noise. This is true in an experiment on the protein alignment problem and a benchmarking test.

In the future, we hope to conduct more numerical experiments on the performance of STORM when solving noisy functions. An interesting question would be whether what types of noise or what levels of noise can be handled by STORM. Quantifying these factors will allow us to determine how useful the proposed algorithm is for solving real-world noisy problems. Another possible research direction one could take is to extend our results to the construction probabilistically quadratic models. How they would work in practice remains an open question of our study.

Chapter 5

Conclusions and Future Work

In this thesis, we have addressed general simulation-based optimization problems when the objective function is computed with noise. We have discussed and analyzed new noise-adapted algorithms from both randomized approaches and trust-region model-based approaches. Our algorithms make use of the noise estimates throughout the course of optimization in order to obtain better approximate solutions for the noisy functions than existing methods. Algorithmic advancements for derivative-free optimization can enhance the ability of DFO methods to handle general noisy problems.

In Chapter 2, we presented a review of existing derivative-free optimization methods of different types, particularly, randomized approaches and trust-region model-based approaches. It reveals the potential of improving both schemes to optimize noisy functions and this motivates our work in the following chapters. We also introduced the protein structure comparison problem as an ideal test function for developing robust DFO optimization methods for noisy problems, because the objective function of finding the maximal overlapping volume of proteins is computed

with stochastic noise and controllable deterministic noise.

In order to solve the proposed biological application problem, in Chapter 3 we introduced an adaptation of DFO and VASP (the black-box volume estimator), DFO-VASP, that is able to generate practical and accurate protein superpositions in a timely manner. DFO-VASP is designed to handle the presence of controllable deterministic noise. It employs several new algorithmic ideas, for instance, utilizing the noise estimates in the trust-region framework, dynamically adjusting the function accuracy and warm-starting the search for a more global optimizer. Numerical experiments were presented to illustrate the accuracy and computational efficiency of our method comparing to the original scheme. In this chapter, we are especially interested in evaluating how biologically meaningful our solutions are. Hence, a large scale validation was performed and it indicates the capability of DFO-VASP for finding superposition of binding cavities that can be used to detect influences on binding specificity.

In Chapter 3, we turned our attention to assess the potential of applying randomized derivative-free methods to noisy problems. The proposed new algorithm, STARS, demonstrates that careful choices of noise-adjusted smoothing step sizes can improve the practical competency of randomized methods when the objective function has stochastic noise. This theoretically-verified algorithm is a variant of Nesterov's randomized approach in [63] and is greatly motivated by More and Wild's recent work on estimating computational noise [58] and on estimating the derivatives of noisy simulations [59]. We provided convergence rate analysis of STARS in both additive and multiplicative noise settings. Our experiments show that STARS exhibits noise-invariant behavior with respect to different levels of stochastic noise and the empirical performance of STARS is superior than that indicated by our theoretical bounds and also comparing to several other randomized zero-order approaches.

Encouraged by the results in Chapter 2, we introduced a stochastic DFO algorithm, STORM, for solving noisy functions in Chapter 4. In order to deal with the randomness in the objective funtion evaluation, we employed random models and estimates that are sufficiently accurate with high probability. We proved $\liminf$-type and lim-type convergence results for our algorithm and provide analysis on how many random pointed are needed to construct these probabilistic models with sufficiently high probability. From an experimental point of view, preliminary results demonstrate the benefits of our proposed algorithm in solving noisy functions over previous versions of DFO methods, on both the protein alignment problem and a benchmarking suit.

There remain many open avenues for future work and we briefly mention some areas of particular interest to us. First, we hope to explore other models. The use of 2-norms in both terms of the objective function is natural as it leads to an optimization problem with a closed form solution. However, other regularized regression models, such as Lasso [80], or SVM regression [33], may have certain advantages depending on the properties of the noise. For instance the SVM regression models, which utilize hinge loss term $\sum_{i=0}^{p} \max\{|M(\bar{\Phi}, y_i)\alpha - \tilde{f}(y_i)|, \epsilon\}$, for a given ϵ, instead of the quadratic loss $\|M(\bar{\Phi}, Y)\alpha - \tilde{f}(Y)\|^2$, can be particularly advantageous in the case where the noise level is usually deterministic and can be estimated and controlled. In this case an additional computation burden introduced by solving the SVM regression problems has to be handled efficiently. Also, in this thesis we mostly considered serial optimization environment, except briefly in Section 4.6. However, distributed computing is increasingly prevalent and there remains a demand of developing more customized parallel variants of DFO methods.

Bibliography

[1] M. A. Abramson and C. Audet. Convergence of mesh adaptive direct search to second-order stationary points. *SIAM Journal on Optimization*, 17:606–619, 2006.

[2] M. A. Abramson, C. Audet, J. E. D. Jr., and S. L. Digabel. Orthomads: A deterministic mads instance with orthogonal directions. *SIAM Journal on Optimization*, 20(2):948–966, 2009.

[3] A. Agarwal, D. P. Foster, D. Hsu, S. M. Kakade, and A. Rakhlin. Stochastic convex optimization with bandit feedback. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. C. N. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1035–1043, 2011.

[4] E. J. Anderson and M. C. Ferris. A direct search algorithm for optimization with noisy function evaluations. *SIAM Journal on Optimization*, 11:837–857, 2001.

[5] C. Audet, V. Béchard, and S. Le Digabel. Nonsmooth optimization through mesh adaptive direct search and variable neighborhood search. *Journal of Global Optimization*, 41:299–318, 2008.

[6] C. Audet and J. E. Dennis Jr. Mesh adaptive direct search algorithms for constrained optimization. *SIAM Journal on Optimization*, 17:188–217, 2006.

[7] A. Auger. Convergence results for the $(1, \lambda)$-SA-ES using the theory of φ-irreducible markov chains. *Theoretical Computer Science*, 334(1–3):35–69, 2005.

[8] P. C. Babbitt, M. S. Hasson, J. E. Wedekind, D. R. Palmer, W. C. Barrett, G. H. Reed, I. Rayment, D. Ringe, G. L. Kenyon, and J. A. Gerlt. The enolase superfamily: a general strategy for enzyme-catalyzed abstraction of the alpha-protons of carboxylic acids. *Biochemistry*, 35(51):16489–501, 1996.

[9] A. S. 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.

[10] G. Berglund, A. Smalas, H. Outzen, and N. Willassen. Purification and characterization of pancreatic elastase from North Atlantic salmon (Salmo salar). *Molecular Marine Biology and Biotechnology*, 7(2):105–14, 1998.

[11] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein Data Bank. *Nucleic Acids Research*, 28(1):235–42, Jan. 2000.

[12] M. Berz, C. Bischof, G. F. Corliss, and A. Griewank, editors. *Computational Differentiation: Techniques, Applications, and Tools*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1996.

[13] H. Beyer and H. P. Schwefel. Evolution strategies–a comprehensive introduction. *Natural Computing*, 1(1):3–52, 2002.

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

[15] T. A. Binkowski, P. Freeman, and J. Liang. pvsoar: detecting similar surface patterns of pocket and void surfaces of amino acid residues on proteins. *Nucleic Acids Research*, 32(web server issue):W555–8, 2004.

[16] B. Chen and S. Bandyopadhyay. A Statistical Model of Overlapping Volume in Ligand Binding Cavities. In *Proceedings of the Computational Structural Bioinformatics Workshop (CSBW 2011)*, pages 424–31, 2011.

[17] B. Chen and S. Bandyopadhyay. VASP-S: A Volumetric Analysis and Statistical Model for Predicting Steric Influences on Protein-Ligand Binding Specificity. In *Proceedings of 2011 IEEE International Conference on Bioinformatics and Biomedicine*, pages 22–9, 2011.

[18] B. Y. Chen, V. Y. Fofanov, D. H. Bryant, B. D. Dodson, D. M. Kristensen, A. M. Lisewski, M. Kimmel, O. Lichtarge, and L. E. Kavraki. Geometric Sieving : Automated Distributed Optimization of 3D Motifs for Protein Function Prediction. In *Proceedings of The Tenth Annual International Conference on Computational Molecular Biology (RECOMB 2006)*, pages 500–515, 2006.

[19] B. Y. Chen, V. Y. Fofanov, D. H. Bryant, B. D. Dodson, D. M. Kristensen, A. M. Lisewski, M. Kimmel, O. Lichtarge, and L. E. Kavraki. The MASH pipeline for protein function prediction and an algorithm for the geometric refinement of 3D motifs. *Journal of Computational Biology*, 14(6):791–816, 2007.

[20] B. Y. Chen and B. Honig. Vasp: A volumetric analysis of surface properties yields insights into protein-ligand binding specificity. *PLoS Computational Biology*, 6(8):11, 2010.

[21] R. Chen, M. Menickelly, and K. Scheinberg. Stochastic derivative-free optimization using probabilistic models. Technical report, Lehigh University, 2014.

[22] R. Chen, K. Scheinberg, and B. Y. Chen. Aligning ligand binding cavities by optimizing superposed volume. In *2012 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 1–5. IEEE Computer Society, 2012.

[23] R. Chen and S. M. Wild. Randomized derivative-free optimization of noisy functions. Technical report, Argonne National Laboratory, 2015.

[24] V. Chen, W. Arendall, J. Headd, D. Keedy, R. Immorino, G. Kapral, L. Murray, J. Richardson, and D. Richardson. MolProbity: all-atom structure validation for macromolecular crystallography. *Acta Crystallographica Section D: Biological Crystallography*, 66(1):12–21, 2009.

[25] A. Conn, K. Scheinberg, and P. Toint. On the convergence of derivative-free methods for unconstrained optimization. *Approximation theory and optimization: tributes to MJD Powell*, pages 83–108, 1997.

[26] A. Conn, K. Scheinberg, and P. Toint. Recent progress in unconstrained nonlinear optimization without derivatives. *Mathematical Programming*, 79:397–414, 1997.

[27] A. Conn, K. Scheinberg, and L. Vicente. Geometry of interpolation sets in derivative free optimization. *Mathematical Programming*, 111:141–172, 2008.

[28] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-Region Methods*. MPS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, Philadelphia, 2000.

[29] A. R. Conn, K. Scheinberg, and P. L. Toint. A derivative free optimization algorithm in practice. Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, Missouri, September 2-4, 1998.

[30] A. R. Conn, K. Scheinberg, and L. N. Vicente. Geometry of sample sets in derivative-free optimization: polynomial regression and underdetermined interpolation. *IMA Journal of Numerical Analysis*, 28(4):721–748, 2008.

[31] 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 Journal on Optimization*, 20(1):387–415, Apr. 2009.

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

[33] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 1 edition, 2000.

[34] G. Deng and M. C. Ferris. Adaptation of the UOBQYA algorithm for noisy functions. In *Proceedings of the 2006 Winter Simulation Conference*, pages 312–319, 2006.

[35] R. Durrett. Probability: theory and examples. cambridge series in statistical and probabilistic mathematics. *Cambridge University Press, Cambridge*, 105:106, 2010.

[36] M. C. Ferris and G. Deng. Extension of the DIRECT optimization algorithm for noisy functions. In B. Biller, S. Henderson, M. Hsieh, and J. Shortle, editors, *Proceedings of the 2007 Winter Simulation Conference*, 2007.

[37] M. C. Ferris and G. Deng. Classification-based global search: An application to a simulation for breast cancer. In *Proceedings of 2008 NSF Engineering Research and Innovation Conference*, Knoxville, TN, 2008.

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

[39] P. Gilmore and C. T. Kelley. An implicit filtering algorithm for optimization of functions with many local minima. *SIAM Journal on Optimization*, 5:269–285, 1995.

[40] L. Gráf, a. Jancsó, L. Szilágyi, G. Hegyi, K. Pintér, G. Náray-Szabó, J. Hepp, K. Medzihradszky, and W. J. Rutter. Electrostatic complementarity within the substrate-binding pocket of trypsin. *Proceedings of the National Academy of Sciences of the United States of America*, 85(14):4961–5, July 1988.

[41] L. Holm and C. Sander. Mapping the protein universe. *Science*, 273(5275):595–603, 1996.

[42] W. Huyer and A. Neumaier. Snobfit – stable noisy optimization by branch and fit. *ACM Transactions on Mathematical Software*, 35(2):9:1–9:25, July 2008.

[43] J. Jägersküpper. How the (1+ 1) ES using isotropic mutations minimizes positive definite quadratic forms. *Theoretical Computer Science*, 361(1):38–56, 2006.

[44] K. G. Jamieson, R. D. Nowak, and B. Recht. Query complexity of derivative-free optimization. In P. L. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, and

K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 2*, pages 2681–2689, 2012.

[45] M. Jebalia, A. Auger, and N. Hansen. Log-linear convergence and divergence of the scale-invariant (1+1)-ES in noisy environments. *Algorithmica*, pages 1–36, 2010.

[46] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, Oct. 1993.

[47] C. T. Kelley. Detection and remediation of stagnation in the nelder-mead algorithm using a sufficient decrease condition. *SIAM Journal on Optimization*, 10:43–55, 1997.

[48] J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23(3):462–466, 09 1952.

[49] K. Kinoshita and H. Nakamura. Identification of the ligand binding sites on the molecular surface of proteins. *Protein Science*, 14:711–718, 2005.

[50] K. Kühnel and B. F. Luisi. Crystal structure of the Escherichia coli RNA degradosome component enolase. *Journal of Molecular Biology*, 313(3):583–92, Oct. 2001.

[51] J. Larson and S. C. Billups. Stochastic derivative-free optimization using a trust region framework. Submitted, 2013.

[52] J. Larson and S. M. Wild. Non-intrusive termination of noisy optimization. *Optimization Methods and Software*, 28(5):993–1011, 2013.

[53] R. M. Lewis, V. Torczon, and M. Trosset. Direct search methods: Then and now. *Journal of Computational and Applied Mathematics*, 124:191–207, 2000.

[54] W. E. Lorensen and H. E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In *Proceedings of the 14th annual conference on Computer Graphics and Interactive Techniques (SIGGRAPH '87)*, volume 21, pages 163–170, 1987.

[55] J. Matyas. Random optimization. *Automation and Remote Control*, 26(2):246–253, 1965.

[56] M. Mckay, W. Conover, and R. Beckman. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21:239–245, 1979.

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

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

[59] J. J. Moré and S. M. Wild. Estimating derivatives of noisy simulations. *ACM Transactions on Mathematical Software*, 38(3):19:1–19:21, Apr. 2012.

[60] J. J. Moré and S. M. Wild. Do you trust derivatives or differences? *Journal of Computational Physics*, 273:268–277, 2014.

[61] K. Morihara and H. Tsuzuki. Comparison of the specificities of various serine proteinases from microorganisms. *Archives of Biochemistry and Biophysics*, 129(2):620–634, 1969.

[62] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[63] Y. NESTEROV. Random gradient-free minimization of convex functions. CORE Discussion Papers 2011001, Université catholique de Louvain, Center for Operations Research and Econometrics (CORE), 2011.

[64] C. A. Oreng and W. R. Taylor. SSAP: Sequential Structure Alignment Program for Protein Structure Comparison. *Methods in Enzymology*, 266:617–635, 1996.

[65] B. Polyak. *Introduction to optimization*. Optimization Software, 1987.

[66] M. Powell. On trust region methods for unconstrained minimization without derivatives. *Mathematical programming*, 97(3):605–623, 2003.

[67] M. J. D. Powell. Trust region methods that employ quadratic interpolation to the objective function. Presentation at the 5th SIAM Conference on Optimization, Victoria, 1996.

[68] M. J. D. Powell. A quadratic model trust region method for unconstrained minimization without derivatives. Presentation at the International Conference on Nonlinear Programming and Variational Inequalities, Hong Kong, 1998.

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

[70] M. J. D. Powell. The NEWUOA software for unconstrained optimization without derivatives. In *Large-Scale Nonlinear Optimization*, volume 83, pages 255–297. Springer, US, 2006.

[71] M. J. D. Powell. Developments of NEWUOA for minimization without derivatives. *IMA Journal of Numerical Analysis*, 28(4):649–664, 2008.

[72] J. Schaer and M. Stone. Face traverses and a volume algorithm for polyhedra. *Lecture Notes in Computer Science*, 555:290–7, 1991.

[73] S. L. Schafer, W. C. Barrett, A. T. Kallarakal, B. Mitra, J. W. Kozarich, J. A. Gerlt, J. G. Clifton, G. A. Petsko, and G. L. Kenyon. Mechanism of the reaction catalyzed by mandelate racemase: structure and mechanistic properties of the D270N mutant. *Biochemistry*, 35(18):5662–9, May 1996.

[74] M. Schumer and K.Steiglitz. Adaptive step size random search. *IEEE Transactions on Automatic Control*, 13(3):270–276, 1968.

[75] I. N. Shindyalov and P. E. Bourne. Protein structure alignment by incremental combinatorial extension (CE) of the optimal path. *Protein Engineering*, 11(9):739–47, Sept. 1998.

[76] D. Sitkoff, K. Sharp, and B. Honig. Accurate calculation of hydration free energies using macroscopic solvent models. *The Journal of Physical Chemistry*, 98(7):1978–88, 1994.

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

[78] N. Srebro, K. Sridharan, and A. Tewari. Smoothness, low noise and fast rates. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Proceedings of Neural Information Processing Systems*, pages 2199–2207. Curran Associates, Inc., 2010.

[79] S. Stich, C. Müller, and B. Gärtner. Optimization of convex functions with random pursuit. *SIAM Journal on Optimization*, 23:1284–1309, 2013.

[80] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58(1):267–288, 1996.

[81] J. J. Tomick, S. F. Arnold, and R. R. Barton. Sample size selection for improved nelder-mead performance. In *Winter Simulation Conference*, pages 341–345, 1995.

[82] V. Torczon. On the convergence of the multidirectional search algorithm. *SIAM Journal on Optimization*, 1:123–145, 1991.

[83] V. Torczon. On the convergence of pattern search algorithms. *SIAM Journal on Optimization*, 7:1–25, 1997.

[84] R. Vijayagopal, P. Sharer, S. M. Wild, A. Rousseau, R. Chen, S. Bhide, G. Dongarkar, M. Zhang, and R. Meier. Using multi-objective optimization for HEV component sizing. In *Proceedings of the International Electric Vehicle Symposium and Exhibition*, number Paper # EVS28_0153, May 2015. Accepted, to appear.

[85] L. Xie and P. E. Bourne. Detecting evolutionary relationships across existing fold space, using sequence order-independent profile-profile alignments. *Proceedings of the National Academy of Sciences of the United States of America*, 105(14):5441–6, 2008.

[86] A.-S. Yang and B. Honig. An integrated approach to the analysis and modeling of protein sequences and structures. I. Protein structural alignment and a quantitative measure for protein structural distance. *Journal of Molecular Biology*, 301(3):665–78, Aug. 2000.

[87] Y. Ye and A. Godzik. Multiple flexible structure alignment using partial order graphs. *Bioinformatics*, 21(10):2362–9, 2005.

[88] A. X. Zheng and M. Bilenko. Lazy paired hyper-parameter tuning. In F. Rossi, editor, *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013)*, Beijing, China, 2013.

Biography

Ruobing Chen was born in July 1988 in Wuhan, Hubei, China. She received Bachelor of Science from Huazhong University of Science and Technology in Control Science in 2010. She began her Ph.D. studies in the Department of Industrial and Systems Engineering at Lehigh University in the Fall of 2010. She was awarded the Rossin Doctoral Fellowship in the Spring of 2013. She spent the summer of 2012 and 2013 at the Laboratory of Advanced Numerical Simulations at Argonne National Laboratory in Lemont, IL. In the summer of 2014, she worked as a Data Mining Intern at Bosch Research and Technology Center in Palo Alto, CA. Currently she is working towards the completion of her concurrent Master of Science and Ph.D. degrees in Optimization in January 2014 and will be joining the Data Mining Services and Solutions Center at Bosch in February 2014 as a Data Scientist.