ACCELERATING DERIVATIVE-FREE OPTIMIZATION WITH DIMENSION REDUCTION AND HYPERPARAMETER LEARNING WITH APPLICATIONS TO STOCHASTIC INVERSE PROBLEMS

by

JORDAN ROBERT HALL

B.S., Northeastern University, 2015

M.S., University of Colorado: Denver, 2019

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Applied Mathematics Program

2020

This thesis for the Doctor of Philosophy degree by

Jordan Robert Hall

has been approved for the

Applied Mathematics Program

by

Stephen Billups, Chair

Varis Carey, Co-Advisor

Troy Butler, Co-Advisor

Julien Langou

Scott Parker

Date: December 12, 2020

Hall, Jordan Robert (Ph.D., Applied Mathematics)

Accelerating Derivative-Free Optimization with Dimension Reduction and Hyperparameter Learning with Applications to Stochastic Inverse Problems

Thesis directed by Computational Scientist Varis Carey and Professor Troy Butler

ABSTRACT

In this dissertation, we propose and analyze methods for efficiently minimizing noisy convex functions without using gradients. We define a parameter-to-data map f and assume the parameter space is high-dimensional. The data space is low-dimensional, typically of dimension 1. In practice, evaluations of f are polluted with noise in the data collection process, modeled by a noisy signal. We assume f is computationally expensive to evaluate, and may be black-box, in the sense that we may lack the closed-form expression of f. We postulate the parameter space gradients ∇f exist, but may be inaccessible. When ∇f is accessible, minimization problems can be solved using gradient-based descent methods; however, when gradients are inaccessible, one may apply *Derivative-Free Optimization (DFO)* algorithms, which minimize f without using exact gradient information. Dimension reduction techniques search for a set of directions in parameter space accounting for the majority of variation of f. We demonstrate that the efficiency of certain DFO algorithms may be improved by applying dimension reduction in parameter space in order to perform algorithmic descent steps in a space of lesser dimension; we call this approach *Active STARS (ASTARS)*. Another challenge without gradients and with uncertain noise levels is estimating the values of certain hyperparameters to perform DFO. We show ASTARS will converge with schemes for estimating necessary hyperparameters and when inexact dimension reduction is performed, in a method we call *Fully Automated ASTARS (FAASTARS)*. We demonstrate in numerical examples that ASTARS and FAASTARS will, on average, perform more efficiently than other standard DFO methods when dimension reduction is possible in parameter space. We propose and investigate different strategies for enhancing the performance of ASTARS and other DFO algorithms, and as an application, we suggest a method which employs ASTARS to solve certain *Stochastic Inversion Problems (SIPs)* and analyze its performance in several examples.

The form and content of this abstract are approved. We recommend its publication.

Approved: Varis Carey and Troy Butler

This thesis is dedicated to my Mother, whose unrelenting support and endless love got me here.

ACKNOWLEDGEMENTS

There are many I wish to acknowledge. Firstly, I would like to thank my co-advisor who I worked most closely with, Varis. Between the classes I took with him and the countless hours we spent in our offices – and now on Zoom – he shared and imparted on me his knowledge and passion for math. Thanks most of all for taking genuine and deep interest in my research, and indeed, contributing so much. Any idea I had was carefully refined and, frankly, often improved by Varis.

Thanks to my other co-advisor, Troy, who taught me practically everything I know about PDE's and inversion. The applications in this thesis would not be possible without Troy. My committee chair, Steve Billups, also had a big influence on my trajectory. The final project I did in Steve's nonlinear optimization class planted a seed which grew into the research in this thesis. Thanks also to Julien Langou, whose expertise in numerical linear algebra kept me honest about the details, and Scott Parker, whose expertise in plasma physics kept me grounded in the complex realities of real-world applications and implications of my work.

Thanks to all of my colleagues in the math department at CU Denver – the professors, instructors, and staff. Audrey Hendricks was a great research mentor for me, outside of my committee, who kindly let me participate in her statistical genetics research group for the last two years, where I had the chance to apply my interests in an interdisciplinary setting. Mike Ferrara was a personal mentor, and the first person I ever talked to at CU Denver, when he called me more than five years ago now, to help me pick out my first classes. Gary Olson was my mentor in teaching and personally, pushing my pedagogical practices and my own mentorship skills, especially when I was TA Coach for a year. Our department's administrative staff, including Julie, Maria, and Susan all supported me behind the scenes, always with smiles.

My officemates and fellow computation researchers, Michael Pilosov, Dan, and Vince kept me sane – thanks guys! Others in my cohort and fellow students, like Megan, Emileigh, Michael Phillips, Tian Yu, Stetson, Nick, and Evan always brightened my day. My friends outside CU Denver, chief among them, Dennis, Carter, Chris, and Topher, have stuck with me through thick and thin for more than a decade. Thanks to my sweet, understanding, and encouraging girlfriend, Darian, who I hope to spend more time with after this process! I thank my family – the grandparents, aunts, uncles, and cousins who support and love me unconditionally. Finally, thanks to my immediate family: my Dad and Beth; my step-brothers David and Daniel; my sister Maddie; and to my Mom, who this thesis is dedicated to. My Mom never stopped pushing me in school. (Maybe now she can stop!)

TABLE OF CONTENTS

LIST OF TABLES

TABLE

LIST OF FIGURES

FIGURE

CHAPTER I

INTRODUCTION

1.1 Overview

This dissertation outlines methods for efficiently solving convex optimization problems with noisy objective functions using computers. Convex optimization is an important challenge in computational science. Convex optimization problems appear in applied mathematics problems in operations research, finance, statistical genetics, and stochastic inversion, among many others. In fact, a major focus of our proposed and investigated methodologies in this thesis is solving *Stochastic Inverse Problems (SIPs)*.

Obtaining the solution to SIPs is another important undertaking in computational science in its own right. By solving a SIP, one may update characterizations of uncertainty in parameter space by combining initial beliefs about parameters and observable data. Using a new approach detailed in Butler *et al.* (2018) – pioneered by my co-advisor Dr. Troy Butler, his collaborators, and his doctoral students – we may solve for a probability distribution in parameter space such that its pushforward (i.e., image) through a parameter-to-data map f exactly matches the observed probability density in data space; this *updated* distribution is the called the *data-consistent solution* to the SIP. Under certain conditions Butler *et al.* (2018); Tarantola (2005), an exact solution of the SIP can be obtained as the solution of a regularized, convex minimization problem.

1.1.1 Notation

We begin by providing Table 1.1 below, which summarizes the notation we shall use throughout this thesis. We may now begin a mathematically precise discussion by defining a parameter space Λ of dimension P, a parameter-to-data map or *model* f, and a data space $\mathcal{D}$; that is, $f : \Lambda \to \mathcal{D}$. Often, f will be a real-valued (or less often, vector-valued) differentiable, convex function of the parameters or variables $\lambda_1, \ldots, \lambda_P$. We assume that $\mathcal{D}$ has dimension D, typically with $D \leq P$. Points in $\mathcal{D}$ may be known values of $f(\lambda)$, $\lambda \in \Lambda$; as such, we may write $d = f(\lambda)$ to denote the particular datum corresponding to the evaluation of a point $\lambda \in \Lambda$ under f.

We allow for realizations of f to be noisy. Hence, we may model draws of $f(\lambda)$ with $\hat{f}(\lambda; \xi) = f(\lambda) + \epsilon(\xi)$, which is an additive noise model, or $\hat{f}(\lambda; \xi) = f(\lambda)(1 + \epsilon(\xi))$, a model for multiplicative noise. Here, $\epsilon(\xi)$ is a random variable with zero mean and positive but finite variance for all realizations $\xi \in \Xi$; if the variance is zero, we have a noise-less or stochastic-free function, or mathematically, $\hat{f} = f$.

Table 1.1: Summary of Notation

Symbol	Description
$\mathbb{R}$	set of real numbers
$\mathbb{N}$	set of natural numbers, beginning with 1
$\mathbb{Z}$	set of integers
Λ	vector space of parameters; often $\Lambda = \mathbb{R}^P$, $P \in \mathbb{N}$
P	$\dim \Lambda = P$, the dimension of Λ
λ	parameters, as scalar ($P = 1$) or vector values ($P > 1$)
λ_i	i-th component of the vector λ
u, x, y, z	also denote scalar or vector parameters
$\{x, y, z\}$	set with elements x, y, z
$\{x^{(i)}\}_{i=1}^n$	set with elements $x^{(1)}, x^{(2)}, \ldots, x^{(n)}$
$\{e^{(i)}\}_{i=1}^P$	standard basis in Λ
$\mathcal{D}$	vector space of data; often $\mathcal{D} = \mathbb{R}^D$, $D \in \mathbb{N}$
D	$\dim \mathcal{D} = D$, dimension of $\mathcal{D}$
d	observed data, as scalar ($D = 1$) or vector values ($D > 1$)
$\|\| \cdot \|\|_X$	(vector) norm in a normed linear space X
f	function $f : \Lambda \to \mathcal{D}$
$\nabla f(\lambda)$	for differentiable f, the $P \times 1$ Λ-gradient
$D_u f(\lambda)$	for differentiable f, the scalar directional derivative in the direction $u \in \Lambda$
L_1	for differentiable f, $\|\|\nabla f(\lambda^{(1)}) - \nabla f(\lambda^{(2)})\|\| \leq L_1 \|\|\lambda^{(1)} - \lambda^{(2)}\|\| \; \forall \lambda^{(1)}, \lambda^{(2)} \in \Lambda$
A, C, Q, V, W, X	denote matrices
A_{ij}	scalar occupying the i-th row and j-th column of matrix A
I	identity matrix
I_n	$n \times n$ identity matrix
A^{-1}	inverse of an invertible matrix A, where $AA^{-1} = A^{-1}A = I$
$A^\top$	transpose of A, where $A^\top_{i,j} = A_{j,i} \; \forall \, i, j$
1_n	$n \times 1$ vector with 1 in every component
$N\left(\mu, \sigma^2\right)$	Gaussian density with mean μ and variance σ^2
$\pi_\Lambda(\lambda)$	general probability density for values λ (scalar or vector) over a domain Λ

We assume that $\hat{f}$ is a black-box function in the sense that $\hat{f}$ lacks a closed form; that is, we only assume that for inputs $\lambda \in \Lambda$ there is a corresponding value $\hat{f}(\lambda)$. In our discussion, we postulate the stochastic-free f is differentiable, but that we lack $\nabla \hat{f} = \nabla f$ in closed form. $\hat{f}$ may represent the process of evaluating some QoI from the numerical solution of a PDE specified by parameters λ. We note that while the convexity of the underlying f may be unclear in this setting, we are often actually more interested in post-processed quantities including $\hat{f}$ evaluations; in particular, objective functions which are convex by construction and include evaluations of $\hat{f}$ only as part of a larger evaluation process. For example, we may be interested in minimizing a quadratic misfit function between observed data and modeled outcomes $\hat{f}(\lambda)$, which is the subject of a proceeding subsection in this chapter about optimization methods for solving SIPs. Regardless, stochastic noise in $\hat{f}$ with convex f pollutes the true signal, and can lead to

$\hat{f}$ violating convexity assumptions. These violations are often mild, and we are usually safe to treat $\hat{f}$ as convex in computational tasks.

1.1.2 Problem and Setting

Let a stochastic-free function f be convex and differentiable. In this thesis, we investigate methods for finding the approximate solution to the *Optimization Under Uncertainty (OUU)* problem

$$\min_{\lambda \in \Lambda} \quad \hat{f}(\lambda), \tag{1.1.1}$$

where we assume:

(I.) evaluations of $\hat{f}$ (possibly a black box) are noisy and expensive;

(II.) the dimension P of the parameter space Λ is large;

(III.) ∇f exists, but we do not have access to its closed form.

By these assumptions, we consider ourselves in an *Uncertainty Quantification (UQ) setting*. Ralph Smith, for example, in his recent textbook titled *Uncertainty Quantification* Smith (2013), considers gradient approximation schemes and many other problems arising in general mathematical modeling under assumptions similar to (I.)-(III.) Likewise, in *Randomized Derivative-Free Optimization of Noisy Convex Functions* Chen and Wild (2015), authors Chen and Wild consider problems under assumptions (I.)-(III.), considering the case of additive noise and, later in the paper, multiplicative noise. The UQ setting often arises in realistic mathematical modeling problems. That is, many mathematical models in real-world problems naturally involve some or all of the assumptions above.

1.1.2.1 An Illustrative Example

To understand how these theoretical concepts may be applied in the real world, let us briefly consider modeling the spread of an infectious disease. One must collect data on the population of interest; it may not be cheap to send researchers into the population to determine the infected, recovered, and susceptible sub-populations, and certainly these data come with some noise – hence, assumption (I.)

Next, one may use the data gathered to determine important ratios such as the proportion of the population which is infected with the disease. These values can be used to determine the solution of a system of *Differential Equations (DEs)* which model the spread of the disease. HIV spread, for example, has dynamics well understood using the ubiquitous *Susceptible, Infected, Recovered (SIR)* model Murray (1989); Smith (2013), which is a system of DEs that happens to have a closed-form solution. This means

that with some of the previously mentioned "important ratios" obtained from our data, one can obtain a function $I(t)$, which gives the number of infected members of the population at time t.

In practice, one can obtain a realistic model for I by using important ratios obtained from data, along with certain parameters that may be given by the problem at hand. Returning to the case of HIV, one may have, for instance, an a priori estimate for the rate at which pregnant mothers may pass HIV onto their newborns. Parameters like these, along with parameters obtained from data are responsible for landing one in assumption (II.)

Finally, assumption (III.) is met is many cases when the system of DEs governing the problem under consideration does not have analytic solutions, meaning that those solutions also lack closed-form gradients. We also note that even when analytical solutions in problems governed by DEs exist, when a *Quantity of Interest (QoI)* is some post-processed quantity involving the solution, extracting analytical gradients is nontrival, and a research area within UQ in its own right. As such, there are an abundance of methods for obtaining estimates to gradients in the UQ setting, though, they can be inexact and greatly distorted by noise in some settings.

It takes only a small extension of the example of disease spread above to understand why one may minimize a noisy signal, $\hat{f}$, in this setting. Let's suppose that the function $-I(t)$ is strictly convex for some known interval of time, $C \subset \mathbb{R}^{>0}$. If $\hat{f}(t) = -I(t) + \epsilon$, then one may be interested in finding

$$\min_{t \in C} \quad \hat{f}(t), \tag{1.1.2}$$

the maximum of the number of infected members of the population, and at what time t during the time interval C that maximum is achieved. (We obtain a maximum number of cases since we are minimizing the negative of I.)

We note that evaluations of I may come with additive noise ϵ in the problem above due to: uncertainty in parameters and data used to obtain the solution $I(t)$; uncertainty in the model $I(t)$ itself due to limits of using any math model to mimic reality. We emphasize that it is rare that a closed-form solution to a system of DEs such as I can be obtained. More often than not, we must solve DEs numerically, which introduces additional uncertainty into realizations of the solution. We have arrived at the problem which this work is devoted to studying: optimizing convex functions $\hat{f}$ in the UQ setting.

1.1.2.2 Addressing Challenges

The challenges associated with assumptions (I)-(III) make performing optimization nontrivial. Assumption (I) motivates the use of efficient methods, equipped to handle stochastic pollution of f in the

form of zero-mean noise. Assumption (II) increases the computational burden of most tasks; it is beneficial to avoid computations over the entire parameter space whenever possible by performing *Dimension Reduction*. Assumption (III) poses a direct challenge, as gradient information is commonly used in optimization. For example, with a closed-form gradient, one could perform any number of gradient-based descent methods in this setting Beck (2014); Bertsekas (2016). However, without ∇f, we consider *Derivative Free Optimization (DFO)*, which performs optimization without ∇f in closed form.

We avoid the challenge posed by assumption (III.) by relying on DFO, which does not require ∇f. Many DFO algorithms involve taking random steps in Λ and updating iterates using local estimates to directional derivatives (i.e., in the direction of the random step) to take approximate descent steps. It is: (1.) cheaper to approximate directional derivatives, requiring much less sampling than forming approximations to the full gradient; (2.) reasonable to trust directional derivatives locally for optimization, especially at certain – usually tiny – length scales. Since DFO requires evaluating $\hat{f}$ (often twice) at each iteration, we are able to store pairs $(\lambda, \hat{f}(\lambda))$, which we think of as samples of $\hat{f}$, albeit deterministic.

There is still another distinct challenge that arises in our setting, related to assumptions (I) and (II). Since we assume $\hat{f}$ is expensive to evaluate, it is of great interest to perform DFO as efficiently as possible, in order to reduce the number of times we must evaluate $\hat{f}$. As well, as $\dim \Lambda = P$ grows, so does the number of DFO iterations needed to optimize $\hat{f}$ – this well-known phenomenon is called the *curse of dimensionality*, and applies not only to optimizing functions with high-dimensional parameter spaces, but also to other computational tasks like sampling in high-dimensional spaces and integrating functions with high-dimensional domains.

To avoid this curse in their computations, many UQ researchers employ dimension reduction in parameter space in this setting whenever possible. Dimension reduction generally involves finding less than P directions in Λ which resolve or explain the behavior of f reasonably well. When we find f is sensitive to less directions than P, many computational tasks – including optimization – can be performed more efficiently. The major contribution of this thesis is a modification to the DFO algorithm STARS Chen and Wild (2015) which ensures iterates step in a space of lesser dimension than P whenever possible.

A brief example may be enlightening here: let $f(x, y) = x^2 + 0.01y^2$. We see f on average depends on x much more than it depends on y, since values of x are magnified 100 times more than y values. In other words, for many points $(x, y) \in \mathbb{R}^2$, $f(x, y) \approx x^2$, which could be interpreted as a cheaper model to replace the full f in computations, although, in this two-dimensional example, the computational winnings are small.

Since $\hat{f}$ may be interrogated as a black box it is not immediately clear how we find these directions. There are several methods to perform dimension reduction. We consider dimension reduction by us-

ing *active subspaces*, an approach developed by Paul Constantine Constantine (2015), which is equivalent to a method outlined in Russi (2010).

The active subspace of f refers to the collection of less than P directions explaining outputs $f(\lambda)$. Finding an active subspace does require evaluations of ∇f, so we encounter the challenge of assumption (III) once again. This challenge is often addressed by taking the gradient of a closed-form *surrogate* which approximates $\hat{f}$.

1.1.3 Background on Essential Computational Methods

We now give a brief overview of computational methods utilized heavily in this dissertation, including surrogate modeling, finite differences, and gradient descent. We give special consideration and provide some discussion of particular aspects of these methods that may present challenges in our setting.

1.1.3.1 Surrogate Modeling

A *surrogate model (surrogate)* is a smooth, closed-form function in Λ used as a cheaper replacement for the true model f. Surrogates can be used to approximately perform evaluations of f in computational processes such as integration and optimization, among many others. We denote surrogates with F, functions $F : \Lambda \to \mathcal{D}$ for which we hope $f(\lambda) \approx F(\lambda)$ for $\lambda \in \Lambda$. We heavily rely on Smith (2013) in the proceeding summary, but also refer to Hastie *et al.* (2001); Sauer (2011).

In practice, a surrogate is formed using M evaluations of the expensive and noisy $\hat{f}$ for various $\lambda \in \Lambda$, in the form $d_i = \hat{f}(\lambda^{(i)}; \xi_i) = f(\lambda^{(i)}) + \epsilon(\xi)$, $i = 1, \ldots, M$, when $\hat{f}$ has additive noise and $\dim \mathcal{D} = D = 1$, which we will assume in this discussion of surrogates. The goal is to model the true signal f, but surrogates will account for noise from both modeling error and observation error in their intercept terms. Surrogate modeling requires an assumption about the form of f, such as $f(\lambda)$ is approximately linear or quadratic in Λ, or possibly approximated by some other polynomial or function. Surrogate modeling often directly borrows techniques from statistical regression.

For instance, linear regression techniques involving *Ordinary Least Squares (OLS)* estimates for coefficients may be used to form a linear surrogate. We write $F(\lambda) = a_0 + \sum_{i=1}^{P} a_i \lambda_i$, for $a_i \in \mathbb{R}$, $i = 0, \ldots, P$ and seek coefficients such that $\sum_{i=1}^{M} (d_i - F(\lambda^{(i)}))^2$ is minimized.

Let $\mathbf{d} \in \mathcal{D}^M$ where $\mathbf{d}_i = d_i$, $i = 1, \ldots, M$, a vector storing the noisy observations of $\hat{f}$. We let $\mathbf{a}$ denote the $(P+1) \times 1$ vector storing the linear coefficients, for which $\mathbf{a}_i = a_i$, $i = 0, \ldots, P$, and let $X \in \mathbb{R}^{M \times (P+1)}$ be a *Van der Monde matrix* with every entry in its first column set to 1 and the remaining entries in a given i-th row as the M samples of $\lambda^{(i)}$ stored as row vectors; i.e., the i-th row of X is a 1 followed by $(\lambda^{(i)})^\top$. To capture the intercept term a_0, we also insert a column of 1's, or formally, the vector $1_{M \times 1}$, so that we can write $X\mathbf{a} = \mathbf{d}$. The OLS estimate to the linear coefficients, denoted $\mathbf{a}_{\text{OLS}}$ is given by

$$\mathbf{a}_{\text{OLS}} = \left(X^\top X\right)^{-1} X^\top \mathbf{d}. \tag{1.1.3}$$

Indeed, we verify (1.1.3) by noting that in our formulations, $X\mathbf{a} = \mathbf{d}$. Then, left-multiplying by $X^\top$ gives $X^\top X\mathbf{a} = X^\top \mathbf{d}$. Noting that $X^\top X$ is $P \times P$ and invertible as long as the columns of X are linear independent, we obtain (1.1.3) by left-multiplying both sides of $X^\top X\mathbf{a} = X^\top \mathbf{d}$ by $\left(X^\top X\right)^{-1}$. One can also show that this choice of coefficients minimizes $(X\mathbf{a} - \mathbf{d})^\top (X\mathbf{a} - \mathbf{d}) = \sum_{i=1}^{M} \left(F(\lambda^{(i)}) - d_i\right)^2$, which is why the solution in (1.1.3) is called the least-squares estimator. We need $M \geq P + 1$ to form the linear surrogate F, in order for the formulations above to be well-posed (i.e., we have enough data to form all $P + 1$ coefficients in $\mathbf{a}$).

By redefining F as $F(\lambda) = a_0 + \sum_{i=1}^{P} a_i \lambda_i + \sum_{i=1}^{P} a_{ii} \lambda_i^2 + \sum_{i=1}^{P} \sum_{j>i}^{P} a_{ij} \lambda_i \lambda_j$, we form a quadratic surrogate. We may use the OLS approach to obtain the coefficients in the quadratic surroagte. We apply (1.1.3) by redefining $\mathbf{a}$ and X, to account for the presence of additional coefficients from the quadratic model. We let $\mathbf{a}_i = a_i$ for $i = 0, \ldots, P$ as before, but now let $\mathbf{a}_{(P+1)+i} = a_{ii}$ for $i = 0, \ldots, P$ and $\mathbf{a}_{(2P+1)+i} = a_{ij}$ for $i = 1, \ldots, P$ and $i < j \leq P$. There are $P(P-1)/2$ coefficients a_{ij} corresponding to the cross-terms $\lambda_i \lambda_j$. Then we have $\mathbf{a} \in \mathbb{R}^{M_Q}$, where $M_Q := (P+1)(P+2)/2$, the number of coefficients required to fit the quadratic surrogate F, and, thus, the minimum number of required samples (i.e., $M \geq M_Q$). We analogously update our definition of the Vandermonde matrix in this quadratic case as the matrix $X \in \mathbb{R}^{M \times M_Q}$ where the i-th row is given by $[1 \;\; \lambda_1^{(i)} \;\; \cdots \;\; \lambda_P^{(i)} \;\; (\lambda_1^{(i)})^2 \;\; \cdots \;\; (\lambda_P^{(i)})^2 \;\; \lambda_1^{(i)}\lambda_2^{(i)} \;\; \cdots \;\; \lambda_{P-1}^{(i)}\lambda_P^{(i)}]$.

We may also use kriging models or *Radial Basis Functions (RBFs)*. We note that kriging models rely on expansions in a polynomial basis and RBFs rely on localized design. In both methods, OLS-like approaches are used to find coefficients which minimize the difference between data d_i and corresponding modeled values $F(\lambda^{(i)})$. For details on these methods, we refer to Smith (2013).

We briefly discuss the conditioning of the presented approaches. Recall the invertibility of $X^\top X$, appearing in (1.1.3) depends on whether X has linearly independent columns. The independence of columns in X will depend entirely on the quality of the data; samples randomly spread throughout Λ are ideal. The *condition number* of a matrix, relative to a chosen matrix norm $||\cdot||_X$ (induced by a chosen norm in Λ-space), is denoted $\kappa(X)$ and given by $\kappa(X) := ||X||_X \, ||X^{-1}||_X$ Trefethen and Bau (1997); Sauer (2011). When the matrix norm is induced by the Euclidean 2-norm in Λ, then we have $\kappa(X) = \sigma_1/\sigma_{M_Q}$, where σ_1 and σ_{M_Q} are the largest and smallest singular values of X from the *full* SVD of X. Now as columns of X approach dependence on one another, we will have singular values near zero, and we expect σ_{M_Q} will be small, meaning $\kappa(X)$ will be large, and we say X is *ill-conditioned* Trefethen and Bau (1997). A great deal of research and effort in the field of computational math is devoted to *preconditioning*, a method by

which X with a large condition number is multiplied by other matrices on the left and/or right to lower its condition number, which may improve the *stability* of the problem at hand (i.e., inverting the form $X^\top X$). Often, standard libraries in coding languages (e.g., numpy in Python) will apply preconditioning methods by default when X is ill-conditioned in OLS surrogate fitting routines.

Typically, when the number of samples M is larger – and the samples are randomly and sufficiently spaced out in Λ – the quality of surrogates improve. Since the purpose of obtaining surrogates in our setting is to perform dimension reduction in order to decrease the number of evaluations of $\hat{f}$, we cannot justify taking a large number of global samples to approximate ∇f, and again, as P grows large, the curse of dimensionality guarantees we would need a tremendous number of samples.

A major contribution of this thesis is approximating an active subspace from surrogates for ∇f trained on deterministic samples coming from optimizing f, or learning hyperparameters needed to optimize f via DFO. In this sense, we will often recycle the valuable information obtained from any and all evaluations of f. As such, we often need to violate a main assumption when computing active subspaces, which is that samples are taken randomly over the entire parameter space. We show: (1) in many cases the global active subspace formed from surrogates are reasonably approximated using deterministic samples; (2) we are often more interested in local approximations of the active subspace.

1.1.3.2 Finite Differences

We now discuss a ubiquitous method in numerical analysis and UQ for approximating derivatives, the *finite difference* method. In fact, there are numerous approaches for finite differencing in various contexts arising from applications in the literature, but we will only need to present a standard approximation to $f'(\lambda) := \frac{d}{d\lambda} f(\lambda)$ for $\lambda \in \Lambda = \mathbb{R}$ ($P = 1$), the first derivative of, in this case, a single-variable function $f(\lambda)$. We will, for now, consider f to be a noise-free function, and for the sake of discussion, assume that f is differentiable; so f' exists, even though we may not know f'.

We follow Chapter 5 of Sauer (2011) in summarizing finite differences in the one-dimensional case. Finite differences are motivated by the Calculus 1 concept that the derivative of f at a given $\lambda \in \mathbb{R}$ is well-approximated by the slope of the line between λ and a nearby point $\lambda' := \lambda + \mu$, $\mu > 0$ (made exact when $\mu \to 0$). Hence, we have reason to believe that

$$f'(\lambda) \approx \frac{f(\lambda') - f(\lambda)}{\mu} \tag{1.1.4}$$

for μ chosen reasonably small.

When $\Lambda = \mathbb{R}^P$, $P > 1$, we form the vector ∇f to contain derivative information, where $\nabla f(\lambda)_i = \frac{\partial f(\lambda)}{\partial \lambda_i}$, $i = 1, \ldots, P$. Now the task is not as simple as when $P = 1$, as with $P > 1$ we must now estimate

all P partial derivatives of f. We may do so by taking perturbations in each of the P coordinates individually, using, for instance, scaled standard basis elements in $\mathbb{R}^P$, denoted $e^{(i)}$ and given by a vector with zeros in all coordinates except for the i-th, which contains the value 1. That is, using the same discretization or step length μ, we perturb values of f at λ by calculating $f(\lambda + \mu e^{(i)})$ for $i = 1, \ldots, P$. We then use the single-variable finite difference scheme in (1.1.4) above to estimate the partial derivatives in turn.

DFO requires estimates to *directional derivatives*, that is, the rate of change of f in a particular direction in Λ. For a direction $u \in \Lambda$, the directional derivative of f in the direction of u, a scalar value denoted $D_u f(\lambda)$ and given by $\lim_{\mu \to 0} \frac{f(\lambda+\mu u)-f(\lambda)}{\mu}$. Using analogous logic to the finite difference schemes for estimating normal derivatives, we may once again use (1.1.4), but with $\lambda' = \lambda + \mu u$, in order to achieve a perturbation in the direction of u.

Since we mainly work with functions $\hat{f}$ with noise, considerations must be made for the ways in which stochasticity pollutes the true signal, f, before simply using (1.1.4) (with $\hat{f}$ replacing f). When the variance in the noise σ^2 is very small, there may not be a need for concern; however, as σ^2 grows, there is reason for concern. In practice μ is quite small, which ensures the accuracy of the finite difference. (Recall, as $\mu \to 0$, the approximation in (1.1.4) becomes exact.) Due to the small step size, the value of f may not change much at all between λ and λ', so noisy perturbations may effect the quality of the difference greatly, with both the value from the difference and σ^2 close in magnitude.

There are methods in the literature by which the numerator in (1.1.4) is de-noised, but these methods require heuristics and are not widely used. Also, when a particular realization of the noise is near zero, de-noising techniques could actually pollute the difference, essentially playing the role of noise.

We closely examine the necessary – and, in fact, optimal – value for μ to perform DFO in our setting in the next chapter. For now, it suffices to say that to ensure reasonable quality in finite differences, we need to discover the tradeoff between shrinking μ for quality in approximations from (1.1.4) and increasing μ so that a large enough step is taken in λ' for the difference to exceed the noise level (hopefully by at least an order of magnitude).

1.1.3.3 Gradient Descent

Given a nonlinear, convex, stochastic-free function f, the method of *gradient descent* (sometimes called *steepest descent* in the literature) is the most standard and widely-used computational method for unconstrained optimization. The main idea of gradient descent is taking iterative steps $\lambda^{(k)}$ in the direction of greatest descent (i.e., most significant decrease in f) – a direction which happens to be $-\nabla f(\lambda^{(k)})$ – controlled by a specific *step size*. We follow Beck (2014) Chapter 4 closely in our forthcoming summary, but also refer to Bertsekas (2016).

We first define a *descent direction* (sometimes called *descent step*) for a fixed point $\lambda \in \Lambda$ as a vector $d \in \Lambda$ for which the directional derivative $D_d f(\lambda) < 0$. A standard follow-up result is that for a descent direction d at the point λ, there exists a step size $h > 0$ such that $f(\lambda + hd) < f(\lambda)$ as in Lemma 4.2 in Beck (2014). Now, as long as we can find and prescribe a step size h_k along with a descent step $d^{(k)}$ for a given $\lambda^{(k)} \in \Lambda$, we could set $\lambda^{(k+1)} = \lambda^{(k)} + h_k d^{(k)}$ in an iterative fashion, a method Beck (2014) calls the *Schematic Descent Directions Method (SDDM)*.

Another theoretical follow-up is the fact we already briefly mentioned at the outset, which is that $d = -\nabla f(\lambda)$ is the direction of greatest descent, in the sense that $\min_{\{d:||d||_\Lambda=1\}} D_d f(\lambda) = -\frac{\nabla f(\lambda)}{||\nabla f(\lambda)||_\Lambda}$, proven in Lemma 4.5 of Beck (2014). (We want the directional derivative in the direction of d to be as small as possible so that we know the value of f is *decreasing* to the greatest extent.)

Combining SDDM with the fact that $-\nabla f(\lambda)$ is the direction of steepest descent among all possible descent steps yields the method of gradient descent. (This is why the method is also called the method of steepest descent.) That is, for a given iterate $\lambda^{(k)}$ and step size h_k (discussed more below), one sets the next iterate as

$$\lambda^{(k+1)} = \lambda^{(k)} - h_k \nabla f(\lambda^{(k)}), \tag{1.1.5}$$

until a stopping criteria is met, such as the value $||\nabla f(\lambda^{(k+1)})||_\Lambda < \tau$ for some tolerance $\tau > 0$.

Careful consideration must be given for the selection of step size h_k for a given step. Here, methods such as *exact line search*, *constant step size*, and *backtracking* are common Beck (2014). Since line searches and backtracking involve additional gradient evaluations, using a constant step size to perform DFO is standard.

1.2 Dimension Reduction in Parameter Space

1.2.1 Motivation

We consider functions which map a high-dimensional space Λ to a data space $\mathcal{D}$ of smaller dimension; i.e., $\dim \mathcal{D} = D << P = \dim \Lambda$. Many functions of interest actually represent post-processed quantities from the solution of potentially complex physical models. It is not often the case that every parameter has an equal impact on function values; usually some parameters matter more than others. In fact, some literature in this area indicate that some physical models – even those that may describe complex situations, such as plasma transport – depend on fewer parameters than what are traditionally called for in a given body of research or application. The notion of a lower-dimensional structure in physical applications can be concretely proven, such as in the low-dimensional behavior observed in the complex Ginzburg-Landau equation (in one spatial dimension), appearing in the studies of fluid dynamics and turbulence in

particle transport Doering *et al.* (1988). The authors show that whereas a Fourier expansion with dozens (or more) of Gaussian modes are used to approximate what is essentially treated as infinite dimensional signal in their application, a smaller amount of modes may be used, since the authors prove a bound on $P = \dim \Lambda$.

We find examples of overparameterization in mathematical and statistical models, especially in scenarios where the true model is unknown. When a model is unknown, an approximate model is typically formed based on: (1.) observed outputs (i.e., data); (2.) assumptions about inputs (i.e., parameters); (3.) assumptions about the structure of the map between these quantities (e.g., the map is approximately linear). For example, one could apply linear regression to learn the extent to which data depends linearly on certain parameters.

With some map f in hand, we wish to understand which parameters are more influential than others on the response of f, observed data. If we can discover which parameters are important (and those that are not), then we can reduce the dimension of Λ by entirely excluding the unimportant parameters. This general technique is aptly-named *dimension reduction*. We should note that in the case that $D > P$, dimension reduction can be applied in data space Shin *et al.* (2017). This is why we emphasize that we present and utilize parameter space dimension reduction methods, since in our setting we assume $D < P$.

If it is possible to mimic the response of f by processing fewer parameters, we expect computational savings on numerical tasks involving evaluations of f, such as integration, optimization, and sampling. For example, the task of fitting a surrogate for f becomes computationally less expensive Constantine *et al.* (2016). We may be able to fix certain unimportant variables, and explore the behavior of f with respect to only the most influential variables, requiring less sampling to resolve f than in full variables.

Recall, the fact that a greater number of samples are required to sufficiently explore a given vector space as the dimension of that space grows is known as the curse of dimensionality. In our case, for example, the volume of a hypercube in Λ with edge length ℓ is ℓ^P, meaning its volume grows exponentially with $\dim \Lambda = P$. Thus, even a hypercube with small volumes in lower dimensions, like the interval $[-1, 1]$, a unit square around the origin, $[-1, 1]^2$, or a unit cube around the origin, $[-1, 1]^3$, becomes much larger, with volume 2^P, as dimension grows. The rate at which the volume of even relatively small hypercubes grow as P increases demonstrates the challenge in exploring large parameter spaces with samples. Computational scientists rely heavily on dimension reduction methods in cases where P is extremely large, to avoid the curse of dimensionality as much as possible in their settings.

We take a similar approach, using dimension reduction whenever possible, in order to more efficiently solve optimization problems. In our setting, efficiency must include strong consideration for the number of times f is evaluated, since we assume those evaluations are expensive. In the proceeding sec-

tion, we present classical and current approaches for performing dimension reduction in our setting. We then introduce active subspaces, which we heavily utilize in proceeding chapters.

1.2.2 Background on Classical and Current Dimension Reduction Methods

In this section, we discuss methods for parameter space dimension reduction in our setting that we can roughly put into two categories: those which rely on surrogates for f and dimension reduction methods that rely on ∇f (sometimes also approximated via surrogates). If we decide to use a surrogate for f, via methods such as regression, we will make implicit assumptions about the form of f, such as that f is approximately linear, quadratic, or some other form. As we illustrate, since assumptions are made about f, some methods for dimension reduction do not require access to ∇f directly. Other dimension reduction methods use ∇f information – whether exact or approximate – to analyze the extent to which various parameters cause change in f. Because we postulate a lack of access to ∇f and that f itself is expensive to evaluate, we pay particular attention to methods that allow for gradient approximations with minimal map evaluations.

In regression analysis, it is possible to detect parameters that do not heavily influence observed data, sometimes called *latent parameters* in the literature. A naïve approach is to remove suspected latent parameters from the setup of the linear model and re-learn the linear relationship between the data and the non-latent parameters. If the newly-learned model (with suspected latent variables removed) fits the data (i.e., quantified by a correlation coefficient) as well as the original model (or better) then the suspected latent variables are indeed latent, and are left out of the model parameter space.

The main problem with removing parameters and re-fitting is that the approach is ad hoc, requiring some sort of intuition about which parameters may be latent. Of course, one could exhaustively check the correlation coefficients of models obtained by removing (and then re-incorporating) variables one at a time (or in groups), but this approach becomes computationally expensive as P increases.

There are more elegant approaches used in the literature to detect latent variables in the regression setting. In fact, the *Partial Least Squares (PLS)* method may be used to perform dimension reduction without access to ∇f at all. However, a major assumption in most PLS frameworks is that f is linear. There are nonlinear frameworks for performing PLS, but the main approaches in the literature either depend on a surrogate method or Gaussian-type kernels Rosipal and Krämer (2006); Papaioannou *et al.* (2019). All regressions assume some kind of structure of f (e.g., linear, polynomial, exponential, etc.)

We prefer methods that do not require assumptions about the form of f whenever possible since we treat f as a black-box in our setting. We find approximations to ∇f are essentially unavoidable to perform meaningful dimension reduction in our setting. While the methods we consider require forming surro-

gates for f to approximate ∇f, the underlying assumptions about f made when forming those surrogates may not greatly impact the quality of the dimension reduction tasks at hand.

Many parameter space dimension reduction methods require gradient information to determine the extent to which f changes for a given set of points in Λ. For instance, global sensitivity analysis, relies heavily on evaluations of ∇f for various points in Λ Smith (2013). Similarly, finding active subspaces requires either the exact form of ∇f or samples of $\nabla f(\lambda)$ for randomly drawn λ, where a greater number of samples will lead to more accurate dimension reduction Constantine (2015). In this and other methods, a surrogate F is typically formed to approximate f using global samples of $f(\lambda)$. Then, the gradient of the closed-form surrogate, ∇F, which is also in closed-form, is used in place of ∇f.

Gradient information – whether exact, or approximated via surrogate modeling – is utilized in various fashions in the literature. Active subspaces rely on the formation of a *sensitivity matrix*, defining a certain covariance structure over Λ with respect to ∇f. Hence, this matrix will contain information about the rate at which $f(\lambda)$ changes with respect to its various inputs λ.

Covariance matrices may be defined in Λ or $\mathcal{D}$. For now, we only consider covariance matrices in parameter space, Λ. Let $\lambda \sim \pi_\Lambda(\lambda)$ where π_Λ is a probability density over Λ. Now λ is a random vector. Formally, a covariance matrix C in Λ is a symmetric $P \times P$ matrix where $C_{i,j}$ equals the covariance between λ_i and λ_j, given π_Λ, our assumed density or distribution of λ.

Often, with a covariance matrix in hand, a statistician may perform *Principal Component Analysis (PCA)*. PCA, or, more specifically, *full* PCA is a method by which one decomposes a symmetric positive semi-definite covariance matrix C via the *Eigenvalue Decomposition (eigendecomposition)* to abstract a set of orthogonal directions (i.e., eigenvectors) in Λ corresponding to largest eigenvalues appearing in the eigendecomposition. Note that orthogonality of the eigenvectors is guaranteed by the symmetry of C. By computing the eigendecomposition of C, we obtain a set of linearly transformed (i.e., rotated and stretched) coordinates or directions accounting for the greatest amounts of variance, in descending order Trefethen and Bau (1997). In practice, depending on $\dim \Lambda = P$, given a set of samples of random λ, researchers might visualize those samples superimposing the first two eigenvectors from the eigendecomposition of C – which correspond to the largest two eigenvalues of C – to visually determine (in a two-dimensional plot) the extent to which the eigenvectors, called principal components, capture the observed variance of the samples. When P is large, one must investigate the obtained eigenvalues of C and analyze their magnitudes. Some researchers resort to ad hoc or heuristic-based decisions about how many principal components to use in a given analysis or application, although it is often the case that the first two principal components are sufficient for many tasks.

PCA – or a method like PCA – is useful in our setting to perform dimensions reduction. Since PCA provides the directions of greatest variance – say, one or two directions – in a covariance matrix, those handful of less than P directions may be used to more cheaply express the information encoded in the full $P \times P$ covariance structure. If we can arrive at a covariance matrix in Λ that expresses the relationship between points $\lambda \in \Lambda$ and the rate of change of $f(\lambda)$ – given by $\nabla f(\lambda)$, we may perform a PCA-like analysis to obtain the directions in Λ which account for the greatest amount of variation (change) in f. In the proceeding section, we shall see that this is a key idea in discovering active subspaces.

1.2.3 Active Subspace (AS) Methods

We consider *Active Subspace (AS)* methods described by Paul Constantine in Constantine (2015) and an equivalent method by T.M. Russi in Russi (2010). These techniques seek to explain outputs $f(\Lambda)$ in an AS $\mathcal{A} \subset \Lambda$ for which $\dim(\mathcal{A}) < P$. Here, we discuss the theoretical formulation of $\mathcal{A}$. The details of finding $\mathcal{A}$ algorithmically is discussed in the proceeding section.

We note that AS requires, at the very least, approximations to ∇f. We continue with the understanding that ∇f needs to be approximated in some fashion, the details of which will be discussed in the proceeding section. We assume that $\nabla f(\lambda)$ is square integrable in Λ with respect to a probability density $\pi_\Lambda(\lambda)$ that is positive everywhere in Λ and 0 otherwise.

In AS – and many other dimension reduction techniques Russi (2010) – we transform inputs λ to a bounded domain with some fixed variance, typically so that $\lambda \in [-1, 1]^P$ for all λ. Then, as in Constantine *et al.* (2015), we write the *sensitivity matrix*

$$W = \int_\Lambda \nabla f(\lambda) \nabla f(\lambda)^\top \pi_\Lambda(\lambda) d\lambda, \tag{1.2.1}$$

which is a $P \times P$ symmetric positive semi-definite matrix defining a certain covariance of ∇f in Λ. This interpretation of (1.2.1) suggests the computation of the eigendecomposition of W,

$$W = VQV^\top, \tag{1.2.2}$$

where V is $P \times P$ unitary with columns given by the orthonormal eigenvectors $v^{(i)}$, $i = 1, \ldots, P$ of W and Q is a diagonal matrix containing the ordered eigenvalues of W, $\{q_i\}_{i=1}^P$. To find the AS, we seek a drop-off in magnitude between some pair of eigenvalues, q_j and q_{j+1}, $1 \leq j \leq j+1 \leq P$, where $q_j >> q_{j+1}$. In this paper, we use at least 90% of the eigenvalues by weight, so that $q_1 + \cdots + q_j \geq 0.9(q_1 + \cdots + q_P)$. The AS of f is the span of $v^{(1)}, \ldots, v^{(j)}$, which we denote

$$\mathcal{A}(f) = \text{span}\{v^{(1)}, \ldots, v^{(j)}\}. \tag{1.2.3}$$

Likewise, we define the *inactive subspace* of f with

$$\mathcal{I}(f) = \text{span}\{v^{j+1}, \ldots, v^{(P)}\}. \tag{1.2.4}$$

The fact that $v^{(1)}, \ldots, v^{(j)}$ correspond to large eigenvalues is precisely why they account for the most amount of variance in function values. In fact, one can view an AS as a reasonable choice of principal components (i.e., eigenvectors) after a *full* PCA is performed in the gradient space W; for more details on this viewpoint, we refer the reader to Section 6.4 in Russi (2010).

For a point $\lambda \in \Lambda$, we define

$$\mathcal{P}_{\mathcal{A}}(\lambda) = \sum_{i=1}^{j} \left((v^{(i)})^T \lambda \right) v^{(i)} \in \mathcal{A}, \tag{1.2.5}$$

which is a projection of the point λ into the AS of f. We call this projection an *active variable*, which is a point in the AS $\mathcal{A}$. We define a submatrix of V with $V_{\mathcal{A}} := V_{1:P,1:j}$, the first j columns of V from the eigendecomposition of W in (1.2.2). Then (1.2.5) can be rewritten as $\mathcal{P}_{\mathcal{A}}(\lambda) = V_{\mathcal{A}} V_{\mathcal{A}}^{\top} \lambda$. We have arrived at the property that

$$f\left(\mathcal{P}_{\mathcal{A}}(\lambda)\right) \approx f(\lambda). \tag{1.2.6}$$

The above property gives the ability to save computational expense in many scenarios in UQ, including optimization, approximation, and solving inverse problems Constantine (2015). We analogously define a projection into the inactive variables with

$$\mathcal{P}_{\mathcal{I}}(\lambda) = \sum_{i=j+1}^{P} \left((v^{(i)})^T \lambda \right) v^{(i)} \in \mathcal{I}. \tag{1.2.7}$$

We define another submatrix of V with $V_{\mathcal{I}} := V_{1:P,j+1:P}$, the last $P - j$ columns of V from the eigendecomposition of W in (1.2.2). Then (1.2.7) can be rewritten as $\mathcal{P}_{\mathcal{I}}(\lambda) = V_{\mathcal{I}} V_{\mathcal{I}}^{\top} \lambda$.

We presented the theory of the AS method, which relies on the exact form of ∇f. In the next section, we present a method for approximating $\mathcal{A}$ using only evaluations of f – or, in our case, $\hat{f}$. The evaluations of $\hat{f}$ are used to obtain approximate gradient information, typically via the formation of a closed-form surrogate.

1.2.3.1 Active Subspace Learning

We present a method for computing an AS from samples. In practice, finding an AS of f – with evaluations of $\hat{f}$ – will require forming an approximation to W from (1.2.1) in a Monte Carlo fashion Con-

stantine *et al.* (2015) as well as gradient approximations. We choose to present a Monte Carlo approach that is simple to implement, found in Russi (2010), and equivalent to the method in Constantine *et al.* (2015). In short, for a draw $\lambda^{(i)} \in \Lambda$, we obtain an approximation to ∇f and store the row vector $\nabla f(\lambda^{(i)})^\top$ in a matrix $\tilde{W}$. The eigendecomposition of $\tilde{W} = \tilde{V}Q\tilde{V}^\top$ gives approximations to the eigenvectors and eigenvalues of W as in (1.2.2).

We assume we lack an analytic form of ∇f; if the analytic gradient is available, the proceeding Monte Carlo method remains valid, and all notations and methods corresponding to approximating ∇f may be dropped.

One initializes the method by performing S random draws of $\lambda^{(i)} \in \Lambda$. We then compute $\hat{f}(\lambda^{(i)})$ for all $i = 1, \ldots, S$ samples, which we note will require S evaluations of f; in a realistic setting, this would require S model solves. We define $D_S := \{s^{(i)}\}_{i=1}^S$, a set of S pairs of samples $\lambda^{(i)}$ and their function values.

Next, we need ∇f – which we assume that we do not have in closed form – evaluated at $\lambda^{(i)}$ for all $i = 1, \ldots, S$. Hence, we typically compute a closed-form surrogate F and use the closed-form ∇F in place of ∇f. Here we form a local linear, global linear, or global quadratic surrogate to f using D_S as in Constantine (2015). We also consider RBFs as an additional surrogate method.

The gradient of the closed-form surrogate is used to approximate ∇f. Using this approximation, we denote each estimation to $\nabla f(\lambda^{(i)})$ with $\hat{\nabla} f(\lambda^{(i)})$ and we define the $P \times S$ matrix $\tilde{W}$ (which is presented below as $\tilde{W}^\top$)

$$\tilde{W}^\top := \left[\widehat{\nabla} f(\lambda^{(1)}) \cdots \widehat{\nabla} f(\lambda^{(S)}) \right]. \tag{1.2.8}$$

Applying an eigendecomposition to $\tilde{W}^\top \tilde{W}$, which is $P \times P$, we obtain $\tilde{W}^\top \tilde{W} = \tilde{V}\tilde{Q}\tilde{V}^\top$ and search for a drop off in the magnitude of the numerical eigenvalues $\{\tilde{q}_i\}_{i=1}^P$. Assuming such a drop off occurs for an index $\tilde{j} : 1 \leq \tilde{j} \leq P$, we let

$$\tilde{\mathcal{A}}\left(\hat{f}; D_S\right) := \text{span}\{\tilde{v}^{(1)}, \ldots, \tilde{v}^{(\tilde{j})}\} \tag{1.2.9}$$

denote the AS of $\hat{f}$ with respect to the samples D_S. We choose this notation including D_S to emphasize the dependence of the approximated AS, $\tilde{\mathcal{A}}$, on the samples taken in Λ.

We now discuss analyzing the quality of the approximated AS $\tilde{\mathcal{A}}$. Recall $V_{\mathcal{A}} = V_{1:P,1:j}$, the first j columns of V from the eigendecomposition of W in (1.2.2) and let $\tilde{V}_1 := \tilde{V}_{1:P,1:j}$, where we are purposely

using j and not $\tilde{j}$ to truncate $\tilde{V}$. (We shall need to define and utilize a truncation of $\tilde{V}$ at $\tilde{j}$, but do so in the next section.)

When $j = \tilde{j}$, the quality of $\tilde{\mathcal{A}}$ is a matter of how closely the approximately active directions $q^{\tilde{(i)}}$ match the exact $q^{(i)}$'s. The extent to which $\tilde{\mathcal{A}}$ accurately approximates $\mathcal{A}$ is quantified by their *subspace distance,* denoted here with $d_S(\mathcal{A}, \tilde{\mathcal{A}})$ and defined, analogous to Constantine and Diaz (2017), as

$$d_S(\mathcal{A}, \tilde{\mathcal{A}}) := ||V_{\mathcal{A}} V_{\mathcal{A}}^\top - \tilde{V}_{\tilde{\mathcal{A}}} \tilde{V}_{\tilde{\mathcal{A}}}^\top||. \tag{1.2.10}$$

Here $||\cdot||$ denotes a matrix norm on the space of matrices $\Lambda^{P \times P}$; for instance, given vector norms $||\cdot||_\Lambda$ and $||\cdot||_{\mathcal{D}}$ on Λ and $\mathcal{D}$ respectively, we could use the matrix norm *induced* by those vector norms given by $||A||_{(P,D)} := \sup_{\{\lambda \in \Lambda \,:\, ||\lambda||_\Lambda = 1\}} ||A\lambda||_{\mathcal{D}}$, as in Trefethen and Bau (1997). The subspace distance between $\mathcal{A}$ and $\tilde{\mathcal{A}}$ essentially quantifies the overlap of these two spans. Indeed, when $d_S(\mathcal{A}.\tilde{\mathcal{A}}) = 0$, we have $V_{\mathcal{A}} = \tilde{V}_{\tilde{\mathcal{A}}}$, meaning that the columns of $\tilde{V}_{\mathcal{A}}$ precisely equal the columns of V_1; when $d_S(\mathcal{A}, \tilde{\mathcal{A}}) = \epsilon$, $\epsilon > 0$, we know that the columns of the associated $\tilde{V}_{\tilde{\mathcal{A}}}$ and $V_{\mathcal{A}}$ increasingly differ as ϵ grows.

In cases in which $\tilde{j} = 1$ or 2, we can use visualizations to check the extent to which the AS accounts for functions values $\hat{f}(\lambda)$ by checking for resolution in a *sufficient summary plot* Constantine (2015), where one plots active variables against function values. Interpolating these values results in a curve ($\tilde{j} = 1$) or surface ($\tilde{j} \geq 2$) called a *response surface.* (We shall discuss response surfaces in $\tilde{\mathcal{A}}$ in detail in a forthcoming section.) In these plots, we hope to see a pattern between the active variables versus their function values. For example, if f is quadratic in its active variables, then we expect to see a quadratic response surface. For $\tilde{j} > 3$ one may check the correlation coefficient between the response surface and true function values.

1.2.3.2 Surrogates from Active Subspaces

In Constantine *et al.* (2014), the authors propose the computation of surrogates or response surfaces in the lower-dimensional AS, with applications to kriging surfaces. Considerations are made for the case where $\mathcal{A}$ is known exactly, or where an approximate $\mathcal{A}$ (formed via the Monte Carlo methods described above) is used in place of $\mathcal{A}$.

Now let $\tilde{V}_{\tilde{\mathcal{A}}} := \tilde{V}_{1:P,1:\tilde{j}}$, the first j columns of $\tilde{V}$ from the eigendecomposition of $\tilde{W}$ in the Monte Carlo AS method. Then $\tilde{V}_{\tilde{\mathcal{A}}}$ is a matrix with columns spanning $\tilde{\mathcal{A}}$. Letting $y^{(i)} := \tilde{V}_{\tilde{\mathcal{A}}}^\top \lambda^{(i)}$ for points $\lambda^{(i)} \in \Lambda$, we obtain points $y^{(i)}$ in the rotated and reduced coordinates of $\tilde{\mathcal{A}}$.

Given $y^{(i)}$, $i = 1, \ldots, M$, a surrogate F is generated from a response surface $\mathcal{R}$ formed from all M samples of the rotated coordinates $y^{(i)}$. While any of the surrogate methods we have briefly discussed could be applied at this stage, the authors discuss and apply the computation of kriging surfaces.

The main advantage of computing the surrogate of $\hat{f}$ in the reduced space $\tilde{\mathcal{A}}$ is that certain computational tasks involving the surrogate will be cheaper and faster than analyses involving a surrogate in full variables. For example, in Constantine *et al.* (2014), a kriging response surface is formed using samples in the approximate AS of the map to a QoI in standard UQ test problem involving a PDE depending on 100 Gaussian random variables. In their application the authors find: (1.) the quality of the kriging surface formed over $\tilde{\mathcal{A}}$ is higher than a response surface formed using a standard method for sensitivity analysis; and (2.) for the same number of samples (i.e., evaluations) of the QoI map, the quality of the kriging surface in $\tilde{\mathcal{A}}$ is higher than a kringing surface formed in the full variables. Since $\dim \tilde{\mathcal{A}} < \dim \Lambda$ in the application, it is cheaper to more thoroughly sample $\tilde{\mathcal{A}}$ than Λ, and because thorough sampling (usually) leads to better surrogates, it is not surprising that surrogates formed over $\tilde{\mathcal{A}}$ will have a higher quality than a surrogate formed in full variables with the same number of samples.

Now, as with most applications of the AS method, one will only observe computational savings when: (1) an AS is clearly defined; and (2.) if and when the AS is approximated, the approximated $\tilde{\mathcal{A}}$ is close to $\mathcal{A}$, quantified by the subspace distance.

1.3 Derivative-Free Optimization (DFO)

1.3.1 Motivation

Many important physical systems possess turbulent or chaotic behavior. The physical state of a time-dependent system $u(t, \lambda)$ and the corresponding parameter-to-observable map may be modeled as a stochastic process, $\hat{f}(u(t, \lambda))$, a deterministic function with additive or multiplicative noise. In this setting, the efficient extraction of accurate gradients of $\hat{f}$ in parameter space is a challenging undertaking, as popular techniques based on linearization, including adjoint methods, are inaccurate Lea *et al.* (2000); Wang *et al.* (2014). And, as we saw in our brief discussion of finite difference, those approximations to ∇f involve $P = \dim(\Lambda)$ additional, usually nonlinear model solves for the physical system state $u(t, \lambda + \delta\lambda)$, and are greatly polluted by the noise in $\hat{f} = f + \epsilon$ or $\hat{f} = f(1 + \epsilon)$.

As a consequence of these difficulties, optimization in this setting often needs to be performed by algorithms which do not require full gradient approximations. Otherwise, we expect to surpass a reasonable computational budget since even one gradient approximation involves $\mathcal{O}(P)$ evaluations of $\hat{f}$ using, for instance, finite differencing.

Instead, many *Derivative-Free Optimization* (DFO) algorithms depend on finite difference-type information in only a single direction in Λ, avoiding approximations to ∇f, which requires perturbations in all P directions. For this reason, some authors in the literature emphasize that these methods may be more aptly called *Gradient-Free Optimization*, since the methods in this body of the research almost al-

ways rely on finite difference-type methods to approximation *directional* derivatives Nesterov and Spokoiny (2017). For our purposes, we shall still refer to these methods as derivative-free, which is more standard in the literature.

We now present a background on classical and modern approaches and methods to perform DFO in our setting. We then present the *STep-size Approximation in Randomized Search (STARS)* algorithm along with methods for learning STARS hyperparameters, which we utilize heavily in the methodologies presented in this dissertation Chen and Wild (2015).

1.3.2 Background on Classical and Current DFO Methods

As in Chen and Wild (2015) and many other papers in the literature Nesterov and Spokoiny (2017); Gorbunov *et al.* (2019), we will consider the additive noise OUU problem

$$\min_{\lambda \in \Lambda} \quad \mathbb{E}\left[f(\lambda) + \epsilon(\xi)\right], \tag{1.3.1}$$

and the multiplicative noise OUU problem

$$\min_{\lambda \in \Lambda} \quad \mathbb{E}\left[f(\lambda)(1 + \epsilon(\xi))\right], \tag{1.3.2}$$

where we assume:

(i.) $f : \Lambda = \mathbb{R}^P \to \mathbb{R} = \mathcal{D}$ is continuously differentiable, convex, and ∇f has a real-valued Lipschitz constant $L_1 > 0$;

(ii.) $\epsilon(\xi)$ is a random variable with probability density $\pi_\epsilon(\epsilon(\xi))$;

(iii.) for all λ and ξ the noise $\epsilon(\xi)$ is independent and identically distributed, has bounded variance σ^2, and is unbiased; i.e., $\mathbb{E}_\xi(\epsilon(\xi)) = 0$;

(iv.) for multiplicative OUU, the additional assumptions that: the *signal-to-noise ratio* is bounded – i.e., $\mathbb{E}((1 + \epsilon(\xi))^{-1}) < b$, $b > 0$ – and the support of $\epsilon(\xi)$ is bounded by $\pm a$, $a < 1$.

Here, L_1 denotes the first-degree or gradient Lipschitz constant of f, which we assume to exist. Note that due to stochastic noise, even though the underlying map f is assumed to be convex, we have no such guarantee for $\hat{f}$.

We consider DFO algorithms suited for additive and multiplicative noise. DFO algorithms typically require nothing more than the ability to evaluate the noisy model and randomly draw from a normal

(or other) probability distribution; ∇f is not required. We note the fairly strong caveat in directly employing certain DFO methods, which is their hyperparameters – namely, descent step size and discretization or *smoothing* used for finite differences – may depend on values generally unknown in our setting.

For example, in the DFO algorithm focused on in this dissertation, Chen and Wild (2015) and others Nesterov and Spokoiny (2017); Gorbunov *et al.* (2019), the hyperparameters depend on the variance in the noise σ^2 and L_1. We must control for noise, since we have seen that noise greatly pollutes finite difference approximations, which we shall utilize to approximation directional derivatives. We must control for L_1 in our setting as well. This value can be interpreted as a bound on all possible slopes (or derivatives) of f, and this bound will directly inform how large of a descent step is permissible, given the variation in f. We will discuss this problem in more detail in a forthcoming section.

As early as 1960, Howard Harry Rosenbrock considered the problem of automatically computing the minimum (or maximum) value of a stochastic-free convex (or concave) function $f(\lambda)$ of real variables, $\lambda \in \Lambda = \mathbb{R}^P$ Rosenbrock (1960). We continue with the assumption that f is convex and not concave, but note that our treatment is analogous for maximizing concave f.

Similarly to our setting, Rosenbrock postulates a lack of access to ∇f, due to the impracticality of obtaining ∇f in applications which motivated his research – "...it will be quite impracticable to differentiate the expressions [in our application] in order to find the gradient at any point: this must be done numerically" (Rosenbrock (1960), pp. 175).

Rosenbrock suggests a method by which one perturbs iterates $\lambda^{(k)}$ in P orthogonal unit directions $e^{(k)}$ to detect which directions to perform a descent step in; i.e., one finds which perturbation $e^{(k)}$ yields the least value of $f(\lambda^{(k)} + \mu e^{(k)})$, where μ controls the extent of the perturbation. Rosenbrock uses a simple heuristic to tune μ. Parallel descent steps are taken individually in the direction causing the greatest decrease in f, using numerically-approximated partial derivatives via finite difference, until steps no longer cause f to decrease. Then, the orthogonal direction causing the second-greatest decrease in f is stepped in, and so on. Once all directions are exhausted, perturbations are recomputed, and a new set of orthogonal directions are generated as well. This process of recomputing directions and stepping in them in turn will continue until no significant change can be made in the value of f.

There are obvious limitations to Rosenbrock's method in our setting; mainly, it is expensive. The f evaluations required to check the extent of minimization for P perturbations at the beginning of a recomputation, along with those used to tune the numerical step length μ for the perturbations at the outset of the problem will exceed numerical budgets even in the age of modern computing. Also, Rosenbrock does not give considerations for functions $\hat{f}$ with stochastic noise. Indeed, Rosenbrock states "It is most

unlikley that the process given here cannot be improved, and in work directed to this end the program may be useful as a standard comparison" (Rosenbrock (1960), pp. 183).

We shall use modern DFO algorithms for standard comparisons, however, Rosenbrock's classical approach is still useful for our consideration. In fact, we shall see that while his approach is more expensive, the main idea behind ASTARS (and some of its modifications) is not so different – both methods try to take steps in directions that minimize f most effectively until f can no longer be minimized in that direction.

Here, we highlight a major difference between the automatic DFO approach for optimizing and a standard method, like gradient descent. Rosenbrock points out that when one performs gradient descent of certain objective functions "In two dimensions, the existence of interaction corresponds to the presence of a long, narrow ridge. If this is not parallel to one of the two axes, progress can only be made by small steps in [those directions] in turn" (Rosenbrock (1960), pp. 176). Now when $P > 2$, a given step in gradient descent will only be orthogonal to the previous step – not all descent steps will be mutually orthogonal. Even still, Rosenbrock continues with discussing the poor performance of gradient descent in high-dimensional settings – "there is no reason to believe that the method of steepest descent will have much advantage: certainly it will not if the difficulty in a problem arises chiefly from the interaction of only two variables" (Rosenbrock (1960), pp. 176).

Indeed, if change in f depends mainly on only a handful of directions in Λ (i.e., not spanning all of Λ) which are not *axis-aligned*, then it may not make sense to continually change the descent direction, as in gradient descent. We will find it is more advantageous to descend as much as we can in directions that cause the greatest change in f, especially in our setting, where evaluations of f are expensive. Interestingly, we will motivate and justify a certain extension to ASTARS, called active subcylcing, much in the same way, where we will try to step in active directions (i.e., from performing AS analysis) as much as possible until we recompute an AS.

We now turn our attention to modern methods for DFO, with special consideration for those allowing for a signal $\hat{f}$ polluted with stochastic noise; we continue with the assumptions that the stochastic-free f is convex in Λ and ∇f is inaccessible, as usual. Given an initial iterate $\lambda^{(0)}$, many modern DFO algorithms find subsequent iterates $\lambda^{(k)}$ by first performing random walks or *ball walks* in Λ specified by draws from a P-dimensional Gaussian Chen and Wild (2015); Nesterov and Spokoiny (2017) or a surface, such as a unit hypersphere in Λ Gorbunov *et al.* (2019), forming a random direction $u^{(k)}$. The iterate $\lambda^{(k)}$ is perturbed in Λ-space by $u^{(k)}$ (scaled by a smoothing parameter μ). We obtain a point we denote here as $\lambda^{(k+1)} := \lambda^{(k)} + \mu e^{(k)}$, and we evaluate $\hat{f}$ at this perturbed point. Then, similar to Rosenbrock (1960), all methods rely on some type of routine, sometimes called a *gradient oracle*, which involve approximations to

the directional derivative of f in the direction of the random step $u^{(k)}$ (see 1.1.3.2 on finite differences for directional derivatives) to perform a descent step that will (approximately) decrease the value of f in that direction. Iterates are controlled by prescribed hyperparameters, namely the smoothing factor for finite differencing (which we continue to denote with μ to follow Chen and Wild (2015) and others) and step size for descent steps. Step sizes h are constant in the DFO methods we study and modify, and are taken to be inversely proportion to both L_1 and $P = \dim \Lambda$.

In Nesterov and Spokoiny (2017), the authors point out the double-edged sword mathematicians encounter in the study of DFO: the algorithms are simple to implement for any moderately-experience programmer, but are "difficult for theoretical investigation," certainly demonstrated in our theoretical results in the next chapter (Nesterov and Spokoiny (2017), pp. 528).

By the 1980's, the invention of *Automatic Differentiation (AD)* (and its many modifications and applications) allowed for the construction of full gradients using only (carefully-tracked-and-stored) function evaluations, and DFO fell out of style in the optimization community. Indeed, the authors continue: automatic differentiation "destroyed the last arguments for supporting the idea of [DFO in the 1980's]. During several decades, these methods were almost out of computational practice. However, in the last years, we can see a restoration of interest to this topic" (Nesterov and Spokoiny (2017), pp. 528).

Nesterov and Spokoiny outline the various ways in which DFO may make more sense to use in certain settings than automatic differentiation – even in 2020. In Nesterov and Spokoiny (2017), the authors cite multiple advantages of DFO, including challenges and computational expenses in modifying legacy software to store all intermediate calculations (required for AD). In other words, a major advantage of DFO in our setting – especially with a high-dimensional parameter space – is a much lighter computational burden with respect to storing certain calculations.

1.3.3 STep-size Approximation in Randomized Search (STARS)

In Chen and Wild (2015) the *STep-size Approximation in Randomized Search (STARS)* DFO algorithm is used to numerically solve the additive and multiplicative OUU problems (1.2.1) and (1.2.2) under assumptions (i.)-(iv.) Briefly, STARS uses small perturbations of iterates in Λ by the addition of a random vector with components drawn from a normal distribution, computes the noisy function value at the randomly perturbed point, and updates iterates using a Gaussian-smoothed finite-difference for approximate gradient information in a gradient-descent-type scheme. STARS only requires the ability to evaluate $\hat{f}$ and take random draws from a normal distribution. All in all, the algorithm is implemented in about 10 lines of code in any standard computing language. We used Python 3 for the results presented in this dissertation.

As mentioned above, STARS has two primary hyperparameters, a smoothing factor μ^* and step size h. The smoothing factor is used as the discretization necessary for performing a finite difference approximation to directional derivatives; the step size h is used in an *approximate* descent step STARS takes to update its iterates. We witness best performance in STARS when these two hyperparameters are near their optimal values. We discuss learning these important hyperparameters algorithmically in the proceeding subsection. We begin by presenting the precise relations between the needed STARS hyperparameters and the values L_1, σ^2, and P. We then present and discuss pseudocode for STARS Chen and Wild (2015).

STARS requires defining hyperparameters, denoted μ_k^* and h, which are the algorithm's *smoothing factor* and step size, respectively and $k = 1, \ldots, M$ denotes iterations 1 through M, the maximum number of iterations. The step length h will remain constant for all iterations regardless of the type of noise in $\hat{f}$. In the case of Additive OUU, μ_k^* will also be constant, i.e., $\mu_k^* = \mu^*$, a fixed value for all iterations $k = 1, \ldots, M$. However in the case of Multiplicative OUU, the smoothing factor μ_k^* will need to be adjusted at every iteration k.

For the additive noise OUU problem (1.3.1), the values for μ^* and h are

$$\mu^* := \left(\frac{8\sigma^2 P}{L_1^2(P+6)^3}\right)^{1/4} \quad h := (4L_1(P+4))^{-1}. \tag{1.3.3}$$

The authors in Chen and Wild (2015) show that given the constant step size h, the value μ^* is an optimal choice for minimizing error in the finite difference calculations used for estimating directional derivatives. (We investigate the optimality of smoothing parameters in the next chapter.)

In the multiplicative noise OUU problem (1.3.2), the step length h remains the same, exactly as in (1.3.3) above, held constant for each iteration. However, the smoothing factor must be updated for each iteration $k = 1, \ldots, M$. As shown in Chen and Wild (2015), the optimal smoothing factor for an iterate k is given by

$$\mu_k^* := \left(\frac{16\sigma^2 \hat{f}(\lambda^{(k)})^2 P}{L_1^2(1+3\sigma^2)(P+6)^3}\right)^{1/4}. \tag{1.3.4}$$

We present STARS suited for additive or multiplicative OUU as in (1.3.1) and (1.3.2) in the pseudocode below, using our notations. The algorithm is initiated by defining a maximum number of iterations to be taken (`maxit`), which we denote with M, an initial iterate $\lambda^{(0)} \in \Lambda$ and its corresponding noisy evaluation $\hat{f}(\lambda^{(0)})$. We also need to define h and μ^* (where μ^* is updated at every step in the case that the

noise is multiplicative); we now see in (1.3.3) and (1.3.4) that we shall need to know L_1 and σ^2 to do so. Then, for a given step, k, the algorithm will draw a random vector $u^{(k)}$, evaluate $\hat{f}$ at a perturbation of the previous iterate in that direction controlled by the smoothing factor, $\lambda^{(k-1)} + \mu_k^* \cdot u^{(k)}$, and use a finite-difference type calculation to approximate $D_{u^{(k)}} f$. Using this approximate derivative information, a *approximate* descent direction $d^{(k)}$ is formed and an *approximate* descent step is taken, controlled by the step size.

Algorithm 1: STARS Chen and Wild (2015)

Input: `maxit`=: M; $\lambda^{(0)}$; $f_0 := \hat{f}(\lambda^{(0)})$; h; $k = 1$
while $k \leq M$ **do**
  Form smoothing factor μ_k^* (can be performed at outset just once for additive noise)
  Draw $u^{(k)}$, where $u_p^{(k)} \sim N(0,1)$ for $p = 1, \ldots, P$;
  Evaluate $g_k := \hat{f}(\lambda^{(k-1)} + \mu_k^* \cdot u^{(k)})$;
  Set $d^{(k)} := \dfrac{g_k - f_{k-1}}{\mu_k^*} \cdot u^{(k)}$;
  Set $\lambda^{(k)} = \lambda^{(k-1)} - h \cdot d^{(k)}$;
  Evaluate $f_k := \hat{f}(\lambda^{(k)})$;
  Set $k = k + 1$;
end
Output: ($\lambda^{(M)}$, f_M)

Notice that STARS' gradient oracle is $\frac{g_k - f_{k-1}}{\mu_k^*} \cdot u^{(k)}$. Here, $\frac{g_k - f_{k-1}}{\mu_k^*} = \frac{\hat{f}(\lambda^{(k-1)} + \mu_k^* \cdot u^{(k)}) - \hat{f}(\lambda^{(k-1)})}{\mu_k^*}$ is used to obtain an approximation to the directional derivative of f in the direction u with a smoothing factor μ_k^*. (This is a finite difference approximation equivalent to what was outlined in 1.1.3.2.) Then, an *approximate* descent direction $d^{(k)} := \frac{g_k - f_{k-1}}{\mu_k^*} \cdot u^{(k)}$ is formed by scaling u by the approximated directional derivative in that direction. By negating $d^{(k)}$, and iterates are updated in that direction by the addition of $-h \cdot d^{(k)}$, where the constant step size h controls for taking too large of a step (possibly causing divergence).

STARS typically converges to a minimum when the hyperparameters μ_k^* and h are within an order of magnitude of their true values. The closer the user-defined μ_k^* and h values are to the truth, the faster STARS converges. If μ_k^* and h are underestimated, STARS will take very small and heavily smoothed steps, converging slowly; however, if the values are overestimated, the algorithm may cause divergence, in the sense that function evaluations will grow with each iterate.

It is then of high interest to tune the values μ_k^* and h so that function evaluations are not wasted. In the proceeding subsection, we discuss learning the values μ_k^* and h depend on – namely σ^2 and L_1.

1.3.4 Learning STARS Hyperparameters

STARS, like many DFO algorithms, exhibits optimal convergence if and only if its hyperparameters – namely the smoothing factor and step size – are properly tuned. For now, it is enough to recall that these hyperparameters depend on the bounded variance in the noise, σ^2, as well as the L_1 Lipschitz constant of the objective function f, and the known dimension of Λ, $P = \dim \Lambda$.

Tuning the hyperparameters in STARS is a matter of learning σ^2 and L_1. We examine and propose algorithmic methods of estimating STARS hyperparameters so that one need not specify the hyperparameters at all, fully automating the process of solving problems (1.2.1) and (1.2.2).

To estimate σ^2, we rely on the ECNoise algorithm Moré and Wild (2015) which even for P large requires few evaluations of $\hat{f}$ – often 6 to 10 evaluations will suffice. Briefly, ECNoise uses a set of nearby samples $s^{(i)} := (\lambda^{(i)}, \hat{f}(\lambda^{(i)}))$ of points $\lambda^{(i)}$ along a line in Λ. Forming a classical difference table of iterative residuals, the authors show that estimators $\hat{\sigma^2}$ to σ^2 may be formed using scaled averaging and selection criteria discussed more in the next section. Learning σ^2 is performed prior to STARS, and herein is viewed as a computational expense one must pay up front to ensure convergence of the algorithm.

Learning L_1 in our setting is a challenge mainly due to our postulated lack of access to ∇f. Given S pointwise estimates to the gradient, which we denote with $\hat{\nabla} f(\lambda^{(i)})$, $i = 1, \ldots, S$, and assuming $\lambda^{(i)} \neq \lambda^j$ for $i \neq j$ we could consider an estimator such as

$$\widehat{L_1} := \max_{i \neq j} \frac{\left|\left| \widehat{\nabla} f(\lambda^{(i)}) - \widehat{\nabla} f(\lambda^{(j)}) \right|\right| - 2\epsilon^*}{||\lambda^{(i)} - \lambda^{(j)}||}, \tag{1.3.5}$$

$\epsilon^* = \sup_\xi |\epsilon(\xi)|, i, j = 1, \ldots, S$, given in Calliess (2017). There are several problems with such an approach in this setting. Mainly, forming each $\hat{\nabla} f$ is expensive, requiring at least $P + 1$ evaluations of $\hat{f}$ for each approximation. Even if one uses only 3 samples $s^{(i)}$, $i = 1, 2, 3$, forming $\hat{L_1}$ requires $3(P + 1)$ f evaluations, which will often exceed the number of function evaluations needed for STARS to converge, assuming its hyperparameters are reasonably tuned.

Another challenge involves specifying ϵ^* in (1.2.3), which is subtracted from the estimator's numerator to control for noise in $\hat{f}$. To save computational expense, we propose forming an initial estimate to L_1 by re-using the samples from ECNoise to approximate *directional derivatives* in a finite-difference fashion, avoiding the naïve approach above (as well as approximating the full ∇f). Then, once STARS is initiated, $\hat{L_1}$ may be updated using information from approximate directional derivatives formed from iterates (and their corresponding function values). We shall see that the iterates (and intermediate iterates)

formed during STARS lend themselves naturally to finite differencing to estimate L_1 – if a larger (more pessimistic) value of L_1 is discovered, we replace or *update* $\hat{L_1}$ with this new value.

Here we present the methods used for producing $\hat{\sigma}^2$, which denotes the estimated bounded variance in the noise of $\hat{f}$, and $\hat{L_1}$, which is the estimator for the gradient-Lipschitz constant of f. To learn the noise in $\hat{f}$, we implement an algorithm based on ECNoise in Moré and Wild (2015), which will typically produce estimates to σ^2 within an order of magnitude of the truth using only 6-10 evaluations of $\hat{f}$. In our methodology, we use ECNoise prior to STARS or ASTARS. After discussing ECNoise, we present a process by which to recycle the samples created in ECNoise to form an initial estimate to L_1. Indeed, the evaluations of $\hat{f}$ created by ECNoise allows one to estimate L_1 in a variety of ways, including via finite difference approximations to directional derivatives, or via the more expensive, naïve estimator in (1.3.5).

We first discuss ECNoise. We first choose $v \in \Lambda$ such that $||v|| = 1$. Let $h_1 > 0$ and $\Delta^{h_1,0} f(\lambda^{(i)}) := \hat{f}(\lambda^{(i)})$. Then the following definitions provide one with classical k-level differences in $\hat{f}$ for samples $\{\lambda^{(i)}\}$ with a chosen discretization h_1:

$$\Delta^{h_1,k+1} f(\lambda^{(i)}) := \Delta^{h_1,k} f(\lambda^{(i+1)}) - \Delta^{h_1,k} f(\lambda^{(i)}). \tag{1.3.6}$$

Let S denote the maximum samples taken – i.e., $i = 0, 1, \ldots, S-1$. ECNoise uses S samples $\lambda^{(0)}, \ldots, \lambda^{(S-1)} \in \Lambda$ where only a *base point* $\lambda^{(0)}$ is specified, then letting $\lambda^{(i+1)} = \lambda^{(i)} + h_1 v$, $i = 0, \ldots, S-1$ provides $S-1$ additional points discretized by h_1. Then, the k-level differences are formed for all samples. For a given level $k < S-1$ we obtain $S-k-1$ differences or residuals. With sufficiently small h_1 and only a moderate choice of S (usually $S = 6$ or 7 sufficies), the differencing scheme in (1.3.4) leaves one with residuals which, when averaged, estimates σ^2, up to a constant. For $k = 1, \ldots, S-1$, and $h_1 > 0$ the authors define the *level k estimate to* σ^2,

$$\hat{\sigma_k}^2 := \frac{(k!)^2/(2k)!}{S+1-k} \sum_{i=1}^{S-k} \left(\Delta^{h_1,k} f(\lambda^{(i)}) \right)^2. \tag{1.3.7}$$

The authors in Moré and Wild (2015) offer sensible heuristics for choosing the estimation level k as well as chosing an appropriate value for the discretization h_1. Given $h_1 > 0$, one begins by computing $\Delta^{h_1,0} f$, which are simply function evaluations, producing S samples of form $(\lambda^{(i)}, \hat{f}(\lambda^{(i)}))$. Let f_{min} and f_{max} denote the minimum and maximum function values among the S samples, respectively. If

$$\frac{f_{max} - f_{min}}{\max\{|f_{max}|, |f_{min}|\}} < 0.1, \tag{1.3.8}$$

then h_1 is too large, and is multiplied by 10^{-2}. Next, the difference table is constructed using (1.3.6) with $k = 0, \ldots, S-1$. If the number of zero entries in the first column of the difference table (i.e., the first residuals) is $S/2$ or larger, then half or more than half of the sampled function values are equal; this means h_1 is too small, and is multiplied by 10^2.

Next, the $S-1$ estimators $\hat{\sigma_k}^2$ are formed according to (1.3.7). There are two main criteria to consider and balance when selecting which estimator to use: (1.) lower levels k are preferable since they are computed from more samples; (2.) a given level k is more preferable when the associated k-th column in the difference table contains residuals with at least 1 sign change (i.e., they are not all positive or negative). As well, the authors test for convergence of the sequence of estimators by checking that some subset of consecutive estimators are close in value. We implement ECNoise as an initial step in the fully-automated ASTARS algorithm.

We now discuss forming an *initial estimate* to L_1, denoted L_1^{init}. Given a direction $v \in \Lambda$, ECNoise uses samples of form $s^{(i)} = (\lambda^{(i)}, \hat{f}(\lambda^{(i)}))$, $i = 0, \ldots, S-1$, in which $\lambda^{(i)} - \lambda^{(i-1)} = h_1 v$ for all $i = 1, \ldots, S-1$. Thus, we may form a second-order finite difference approximation to the directional derivatives (in the direction v), denoted $D_v f(\lambda^{(i)}) \in \Lambda^P$, for each sample (excluding the endpoints). Approximate values $|D_v f(\lambda^{(i)})|$ give rough bounds on the rate f changes in the direction v (and nearby directions) at a scale determined by h_1. As such, estimates to these directional derivatives – which may be formed with samples that have already been taken – may be used as cheap and reasonable approximations to L_1.

Given a direction v and iterate $\lambda^{(i)}$, $i = 2, \ldots, S-2$, we approximate $|D_v f(\lambda^{(i)})|$ with the second-order form

$$\hat{D}_v^{(i)} := \frac{|\hat{f}(\lambda^{(i+1)} + h_1 v) - 2\hat{f}(\lambda^{(i)}) + \hat{f}(\lambda^{(i-1)} - h_1 v)|}{h_1^2}. \tag{1.3.9}$$

We take the maximum $\hat{D}_v^{(i)}$ among $i = 2, \ldots, S-2$ as our choice of L_1^{init} and we initiate ASTARS with $\hat{L_1} = L_1^{\text{init}}$.

Recall that iterates in STARS also take the form $(\lambda^{(k)}, \hat{f}(\lambda^{(k)}))$, and may be used as samples $s^{(k)}$. In addition, one will notice that there is an intermediate sample and corresponding function evaluation formed in every iteration of STARS which takes the form $(\lambda^{(k)} + \mu_k^* u^{(k)}, \hat{f}(\lambda^{(k)} + \mu_k^* u^{(k)}))$ where μ_k^* is the smoothing at iteration k and $u^{(k)}$ is the random vector at iteration k; these are also usable as samples. With additional samples formed during STARS, one may update $\hat{L_1}$ anytime we obtain an estimate L_1^{update} for which $\hat{L_1} < L_1^{\text{update}}$. Updates to $\hat{L_1}$ may be performed between iterations of STARS. The updated value $\hat{L_1} = L_1^{\text{update}}$ may be formed using an un-centered second-order finite difference scheme, which

is similar to how L_1^{init} is formed. One must use care to adjust the finite difference formulas to account for iterates and intermediate function evaluations, which are generally un-centered.

Both hyperparmeters (μ^* and h) must be adjusted if dimension reduction is performed because both values depend on $P = \dim \Lambda$. We note that both parameters in fact have an inverse proportionality with P. Hence, if Λ permits dimension reduction to j *active directions* where, say, $1 \leq j << P$, both hyperparemeters will be larger using j for $\dim \Lambda$ instead of P, which results in lighter smoothing and larger step sizes.

1.4 Stochastic Inverse Problems (SIPs)

1.4.1 Motivation and Background

SIPs arise from a natural question one may ask in our setting: *given a map and uncertain observed data, how can we update our initial understanding of uncertain parameters?* In particular, given observed – and thus, uncertain – data and an initial assumption on the uncertainty in parameter space, SIPs involve updating the probabilistic description of uncertainty in parameter space by incorporating information from an initial description of uncertainty in Λ, uncertain observations in $\mathcal{D}$, and a map $f : \Lambda \to \mathcal{D}$. For now, we will work with the stochastic-free f; considerations for inversion involving noisy $\hat{f}$ are made in Chapter IV.

We follow Butler *et al.* (2018); Stuart (2010); Tarantola (2005); Bardsley (2018); Smith (2013) to formally pose an inverse problem in our setting. First, we assume we have *prior* or *initial* knowledge of the parameter space, given by some density we denote with $\pi_\Lambda^0(\lambda)$ for this introductory discussion. In practical applications, an initial distribution may be given by application experts, but may also reflect the state of knowledge on Λ obtained by other statistical processes. However, when one lacks such a description, one must use an *uninformative initial* density, usually taken as a uniform density over a bounded space $E \subset \Lambda$, or an "unnormalized uniform, posed on the parameter support" (Smith (2013), pp. 100).

One may express uncertainty in $d = f(\lambda)$ (given uncertain λ) with the conditional density $\pi_{\mathcal{D}}(d|\lambda)$ in Λ, often called the *likelihood*, a key ingredient in for performing Bayesian inversion. Alternatively, in DCI, we assume that we have an *observed density*, $\pi_{\mathcal{D}}^{\text{obs}}(d)$, which represents our state of knowledge of observed data, and is uncertain due to, for example, measurement error. Although these densities are distinct objects, at times we may set $\pi_{\mathcal{D}}(d|\lambda) = \pi_{\mathcal{D}}(d)$ for comparisons between Bayesian inversion and DCI.

The inverse problem is the task of finding a density in Λ that combines the given initial, uncertain information in Λ and $\mathcal{D}$. Note that many SIPs are *ill-posed*, in the sense that solutions may not generally exist (without certain assumptions) and are non-unique (without the use of regularization).

Given an initial density π_Λ^0 and $\lambda \sim \pi_\Lambda^0$, the *forward Uncertainty Quantification (UQ) problem* is finding the probability distribution of $d = f(\lambda)$ (in $\mathcal{D}$ space) for a map $f : \Lambda \to \mathcal{D}$. The forward UQ problem is, in its own right, a nontrivial and important problem in uncertainty quantification. We call the solution to a forward UQ problem the *predicted density*, which is a *pushforward* of the initial density under the map f, as in Butler *et al.* (2018). In the context of DCI, forward problems reveal the extent to which a density in parameter space predicts observed data. (Mapping a given density in parameter space through f, we obtain the corresponding density in data space caused by the map, which can be compared to observed data.)

Briefly consider $\Lambda = \mathbb{R}$ and let all λ be independently and identically distributed (iid) as $\lambda \sim N(0,1)$, and let $f(\lambda) = 2\lambda$. Then we can use standard results from probability to conclude that given $\lambda \sim N(0,1)$, $f(\lambda) \sim N(0,4)$, the solution to this forward problem. We see that given the initial density we chose – a one-dimensional zero-mean and unit variance Gaussian – and the map we chose – which is linear – we obtain an exact, analytic solution to this forward problem.

Solving many forward problems, particularly those involving non-parametric initial densities and nonlinear maps f, involve *Kernel Density Estimation (KDE)*, a method by which samples from a given density are used to approximate its corresponding *Probability Distribution Function (PDF)* Smith (2013). Indeed, many nonlinear transformations of $N(0,1)$ – a standard parametric density – do not produce predicted densities with parametric forms.

Bayesian inversion involves equal consideration for the uncertainty in the data (as a likelihood) and π_Λ^0. In particular, we show that the mean of pushforwards of Bayesian solutions to SIPs will generally lie halfway (in data space) between the mean of the data likelihood and the mean of the pushforward of π_Λ^0 in $\mathcal{D}$. On the other hand, Data-Consistent Inversion (DCI) is a novel inversion method in which the solution is constructed so its pushforward exactly matches the observed data, $\pi_\mathcal{D}(d)$.

It is well-documented in the literature Wildey *et al.* (2018); Tarantola (2005); Constantine *et al.* (2016) that one may perform Bayesian inversion by posing an equivalent deterministic optimization problem, under the rather heavy assumptions that the model is linear in Λ and that $\pi_\Lambda^0(\lambda)$ and $\pi_\mathcal{D}(d|\lambda)$ are both Gaussian. We show that performing DCI is also equivalent to solving a certain optimization problem, again, with the assumptions that the map is linear and the given distributions $\pi_\Lambda^0(\lambda)$ and $\pi_\mathcal{D}^{\text{obs}}(d)$ are Gaussian.

In the remainder of this section, we briefly overview Bayesian inversion and DCI, reviewing the literature and importantly presenting the analytic solutions to both general problems. In the final subsection, we discuss optimization methods for solving certain SIPs both in the Bayesian and DCI frameworks and present two illustrative examples.

1.4.2 Bayesian Inversion

The Bayesian approach is the most commonly-used approach for solving inverse problems, involving only an initial description of uncertainty in Λ and a *chosen* likelihood $\pi_{\mathcal{D}}(d|\lambda)$. Throughout the literature on Bayesian inversion, the density π_Λ^0 is called the *prior density*, denoted here with $\pi_\Lambda^{\text{prior}}(\lambda)$. The density $\pi_{\mathcal{D}}(d|\lambda)$ is called a *likelihood*, and describes the probability of fixed observations d, given uncertain λ. While the likelihood could be estimated, as discussed Smith (2013), one often chooses a likelihood density based on their setting. In practice, one often chooses a likelihood to achieve *conjugacy*, which is the property that the prior and likelihood "have the same parametric form" (Smith (2013), pp. 103).

The Bayesian solution is called a conditional density called the *posterior*, which we denote with $\pi_\Lambda^{\text{post}}(\lambda|d)$. We present the Bayesian posterior in our notations, following Tarantola (2005); Smith (2013); Stuart (2010); Bardsley (2018). The posterior distribution is proportional to the product of the initial and observed densities,

$$\pi_\Lambda^{\text{post}}(\lambda|d) = \frac{\pi_\Lambda^{\text{prior}}(\lambda)\pi_{\mathcal{D}}(d|\lambda)}{c}, \tag{1.4.1}$$

where the constant $c = \int_\Lambda \pi_\Lambda^{\text{prior}}(\lambda)\pi_{\mathcal{D}}(d|\lambda)d\lambda$ is used to normalized the product of the prior and observed densities, ensuring $\int_\Lambda \pi_\Lambda^{\text{post}}(\lambda|d)d\lambda = 1$.

Notice (1.4.1) invokes Bayes' Rule in probability, which can be expressed $\mathbb{P}(A|B) \propto \mathbb{P}(A)\mathbb{P}(B|A)$ for independent events A and B. We note that c may be nontrivial to estimate in high-dimensional settings, and may require special quadrature techniques to lessen the computational burden Smith (2013).

1.4.3 Data-Consistent Inversion (DCI)

The authors in Butler *et al.* (2018) show that the *classical Bayesian* or *statistical Bayesian* solution to the inverse problem is not generally a *pull-back probability measure*, here meaning the pushforward of the updated distribution $\pi_\Lambda^{\text{post}}$ from the preceding section is not generally equal to the corresponding assumed data likelihood, $\pi_{\mathcal{D}}(d|\lambda)$, describing the probability that values $\lambda \sim \pi_\Lambda^{\text{prior}}$ predict observed data, $d = f(\lambda)$.

Data-consistent inversion (DCI) seeks an *updated solution*, denoted here as π_Λ^{up}, for which the pushforward exactly equals an observed data density $\pi_{\mathcal{D}}^{\text{obs}}$, hence *data-consistent*. To obtain such a solution, it is necessary to compute the pushforward of π_Λ^0, which in this setting is called the *initial density* and denoted with $\pi_\Lambda^{\text{init}}$.

We must form the pushforward of the initial density by solving the corresponding forward problem; recall, with certain assumptions on f, Λ, and $\mathcal{D}$ the forward problem may be straightforward, with

other assumptions, this problem is non-trivial. We denote the solution to the forward problem with $\pi_{\mathcal{D}}^{f(\Lambda)}$. Then, the *data-consistent* solution as in Butler *et al.* (2018) to the inverse problem is

$$\pi_{\Lambda}^{\text{up}}(\lambda) = \pi_{\Lambda}^{\text{init}}(\lambda)\frac{\pi_{\mathcal{D}}^{\text{obs}}(f(\lambda))}{\pi_{\mathcal{D}}^{f(\Lambda)}(f(\lambda))}. \tag{1.4.2}$$

Notice (1.4.2) involves a specified density (the pushforward of the initial) in its denominator, whereas in (1.4.1), the denominator is only a constant used for normalization. The presence of $\pi_{\mathcal{D}}^{f(\Lambda)}$ in (1.4.2) ensures that observed data is "canceled out" in the updated solution for any directions identically predicted by the pushforward of the initial. Hence, the updated solution is corrected in all directions for which the pushforward $\pi_{\mathcal{D}}^{f(\Lambda)}$ does not match the observed density $\pi_{\mathcal{D}}^{\text{obs}}$.

The updated solution in DCI has certain similarities to the posterior in the Bayesian framework, and is even identical in some cases. However, there are key differences. DCI and Bayesian inversion solve fundamentally different problems. Despite the fact that both methods will yield a new description of uncertainty in Λ based on descriptions of data and initial/prior beliefs about parameters, the Bayesian posterior density and the DCI updated density has fundamentally different properties due to fundamental differences in the methods. In particular, while DCI seeks an updated density which is data consistent – where its pushforward is exactly equal to given observed data – Bayesian inversion has no such guarantee. As we illustrate in the next section, the posterior in Bayesian inversion will equally balance the information from the prior and data likelihood densities, and will generally *not* be data-consistent.

The Bayesian posterior is classical and ubiquitous in the UQ and statistics literature. DCI is a novel approach to inversion, relying on deep results in measure theory, notably including the disintegration theorem Butler *et al.* (2018). While the underpinnings of DCI are outside the scope of this dissertation, we do note that the updated density is guaranteed to exist and be unique and stable with respect to perturbations in Λ, with key assumptions of compatibility and predictability discussed further in Butler *et al.* (2018).

1.4.4 Optimization Methods for Solving SIPs

As in Tarantola (2005) – depending on the model and given initial and observed desities – the Bayesian solution to the inverse problem can be obtained exactly or approximately by solving a corresponding deterministic convex optimization problem. In particular, the optimization approach is valid in the Bayesian setting when f is linear (i.e., when $f(\lambda) = A\lambda$ for a matrix $A \in \Lambda^{D\times P}$) and when the given prior and data likelihood are conjugate Gaussians. With analogous assumptions, one may alter the Bayesian or *classical* formulation of the deterministic optimization problem to ensure one obtains the (ex-

act or approximate) *data-consistent* solution to the inverse problem Wildey *et al.* (2018), which we discuss after presenting the Bayesian approach.

As in Tarantola (2005), given $d' \sim \pi_{\mathcal{D}}(d|\lambda)$ and $\lambda' \sim \pi_{\Lambda}^{\text{prior}}$, we define the regularized *misfit function*

$$S(\lambda) = \frac{1}{2}\left(||A\lambda - d'||_{\mathcal{D}}^2 + ||\lambda - \lambda'||_{\Lambda}^2\right). \tag{1.4.3}$$

We express the Λ and $\mathcal{D}$ norms appearing in S above as Euclidean norms involving covariance matrices. We assume $\pi_{\Lambda}^{\text{prior}}$ is a known Gaussian density with mean $\bar{\lambda}$ and covariance matrix C_{Λ}; likewise, the likelihood $\pi_{\mathcal{D}}(d|\lambda)$ is a Gaussian with mean $\bar{d}$ and covariance matrix $C_{\mathcal{D}}$. As in Tarantola (2005), we use the covariance matrices C_{Λ} and $C_{\mathcal{D}}$, rewriting (1.4.3) as

$$S(\lambda) = \frac{1}{2}\left(\left|\left|C_{\mathcal{D}}^{-1/2}(A\lambda - \bar{d})\right|\right|_2^2 + \left|\left|C_{\Lambda}^{-1/2}(\lambda - \bar{\lambda})\right|\right|_2^2\right), \tag{1.4.4}$$

where $||\cdot||_2$ denotes the standard Euclidean norm and the misfit function in (1.4.3) is specified by $\lambda' = \bar{\lambda}$ and $d' = \bar{d}$ in our notations.

The misfit function defined above may be understood term-by-term: given $\lambda \in \Lambda$, the term $||A\lambda - \bar{d}||_{\mathcal{D}}^2$ corresponds to finding the mismatch between the observed mean in the data space, $\bar{d}$ and $A\lambda$ in data space; the term $||\lambda - \bar{\lambda}||_{\Lambda}^2$ corresponds to performing *Tikhonov regularization*, ensures a unique solution. We note the Tikhonov regularization term forces the minimizer of S towards the direction of the mean of the prior in parameter space, $\bar{\lambda}$, in every direction and

The misfit function (1.4.4) appears in the posterior (i.e., statistical Bayesian solution) of the inverse problem, as $\pi_{\Lambda}^{\text{post}}(\lambda) \sim \exp(-S(\lambda))$. The minimizer of S is called the *maximum a posteriori point, (MAP point)* and denoted λ^{MAP}. The MAP point is the mean of the Gaussian posterior, $\pi_{\Lambda}^{\text{post}}$. As in Tarantola (2005), we write the covariance matrix associated with $\pi_{\Lambda}^{\text{post}}$ as

$$C_{\text{post}} := \left(A^{\top}C_{\mathcal{D}}^{-1}A + C_{\Lambda}^{-1}\right)^{-1}. \tag{1.4.5}$$

As well, we present the analytical form of the MAP point,

$$\lambda^{\text{MAP}} = \bar{\lambda} + C_{\text{post}}A^{\top}C_{\mathcal{D}}^{-1}\left(\bar{d} - A\bar{\lambda}\right), \tag{1.4.6}$$

also provided in Tarantola (2005).

In Wildey *et al.* (2018); Pilosov (2020), the objective function S is re-formulated so the minimizer is the data-consistent *Maximum Updated Desnity point (MUD point)*, denoted λ^{MUD}. The MUD point

is the mean of the associated updated solution π_Λ^{up}, which is normally distributed with linear and Gaussian assumptions similar to those above. An additional "deregularization" term is appended to S so that Tikhonov regularization is not applied in directions informed by the data. If a unique solution to the pure data misfit exists, the Tikhonov regularization is completely "turned off" (i.e., is zero) since all directions are informative. The *data-consistent misfit function* in our setting is

$$T(\lambda) = \frac{1}{2}\left(\left|\left|C_{\mathcal{D}}^{-1/2}(A\lambda - \bar{d})\right|\right|_2^2 + \left|\left|C_\Lambda^{-1/2}(\lambda - \bar{\lambda})\right|\right|_2^2 - \left|\left|C_A^{-1/2}(A(\lambda - \bar{\lambda}))\right|\right|_2^2\right), \tag{1.4.7}$$

where $C_A := AC_\Lambda A^T$; note that with $D = 1$, $C_A \in \mathbb{R}^{>0}$ and $C_A^{-1/2}$ is well-defined as long as $A \neq 0$. In the case that $\pi_\Lambda^{\text{init}}$ and $\pi_{\mathcal{D}}^{\text{obs}}$ are Gaussian – with respective means $\bar{\lambda}$ and $\bar{d}$ and covariance matrices C_Λ and $C_{\mathcal{D}}$ – and f is linear, the data-consistent solution to the inverse problem is given exactly by $\pi_\Lambda^{\text{up}}(\lambda) \sim \exp(-T(\lambda))$.

A major contribution of the recent dissertation Pilosov (2020) is the derivation of the closed-form covariance of the (Gaussian) updated density,

$$C_{\text{up}} := C_\Lambda - C_\Lambda A^\top C_A^{-1}\left(C_A - C_{\mathcal{D}}\right)C_A^{-1}AC_\Lambda, \tag{1.4.8}$$

As well, we present the analytical form of the MUD point,

$$\lambda^{\text{MUD}} = \bar{\lambda} + C_\Lambda A^\top C_A^{-1}\left(\bar{d} - A\bar{\lambda}\right), \tag{1.4.9}$$

also provided in Pilosov (2020).

The deregularization term in (1.4.7) ensures the MUD point is regularized only in uninformative (i.e., inactive) directions, and is analogous to the consequence of the division by the pushforward of the initial density in (1.4.2). In the case that a unique solution to the inverse problem exists (i.e., $P = D$ and A is full rank), the deregularization term equals the Tikhonov regularization term, so that the MUD point solves the data misfit exactly with no regularization at all and equals the MAP point. When the problem is under-determined ($P > D$), the data-consistent solution lies in a hyperplane in Λ containing all the possible points that could have produced the observed data. Regardless of whether $P = D$ or $P > D$, the MAP point obtained by minimizing the classical misfit S, in general, will not be a point that can reproduce the observed data (i.e., $A\lambda^{\text{MAP}} \neq \bar{d}$ in general), whereas the MUD point minimizing T guarantees the solution will lie on the observed data manifold (i.e., $A\lambda^{\text{MUD}} = \bar{d}$ by design).

In our setting, we use a stochastic linear model $\hat{f}(\lambda;\xi) = A(\lambda) + \epsilon(\xi)$ appearing in place of A in S and T. We will address the noise induced by this assumption in Chapter IV. For now, it is enough

to say that both S and T can be re-written as a function with additive noise, and although its noise has nonzero mean, we can readily make the noise zero-mean by subtracting its nonzero first moment, which does not change the *minimizer* of S or T. We highlight the difference in solutions to S and T (without a noisy signal) in the examples below.

Example 1. *Let* $f(\lambda) = 2\lambda$, $\bar{\lambda} = 0.1$, $C_\Lambda = [0.5]$, $\bar{d} = 0.25$, *and* $C_\mathcal{D} = [0.25]$*; note,* $P = D = 1$. *Then we find that* $S(\lambda) = 2((2\lambda - 0.25)^2 + (\lambda - 0.1)^2)$, *which is minimized by* $\lambda^{MAP} = 3/25$. *Assuming that* $\bar{\lambda}$ *and* $\bar{d}$ *are the means of Gaussian prior and data likelihood densities respectively, with variances corresponding to the given covariance matrices, we have* $\pi_\Lambda^{post} \sim \exp(-S(\lambda))$ *with mean* λ^{MAP}. *Notice* $f(\lambda^{MAP}) = 6/25 \neq \bar{d}$. *We write* $T(\lambda) = 2(2\lambda - 0.25)^2$, *which is minimized by* $\lambda^{MUD} = 1/8$. *Notice* $f(\lambda^{MUD}) = \bar{d}$. *With Gaussian assumptions on the initial and observed as well as* f *linear, we have* $\pi_\Lambda^{up} \sim \exp(-T(\lambda))$ *with mean* λ^{MUD}.

In the preceding example, we observe for linear f and $P = D$, the classical MAP point, given by $\lambda^{\text{MAP}} = 0.12$ and obtained by minimizing S, strikes a balance between the prior belief that $\lambda = 0.1$ and the fact that the value $\lambda = 0.125$ is the pre-image of 0.25 under f, the observed data point. The data-consistent MUD point, given by $\lambda^{\text{MUD}} = 0.125$, is exactly the pre-image of $\bar{d}$. Indeed, since the linear map A is full rank in Example 1, the third term in (1.4.7) fully deregularizes (cancels out) the second term, so that the minimizer of T is actually just the minimizer of the first term, the data mismatch. We see here that classical Bayesian inverse problem theory and DCI pose and answer different questions.

For a similar concrete example where the number of parameters P is greater than the dimension of the data space, D – so that the data misfit problem is under-determined without regularization – we present the following example directly from Wildey *et al.* (2018) (in our notations).

Example 2. *Let* $f(\lambda) = 2\lambda_1 - \lambda_2$, $\bar{\lambda} = (0.1 \quad 0.2)^\top$, $C_\Lambda = diag[0.5, 0.25]$, $\bar{d} = 0.1$, *and* $C_\mathcal{D} = [0.25]$. *Note that with* $P > D$, *the data misfit problem is under-determined. In this case, with* $P = 2$ *and* $D = 1$, *for any* $\lambda \in \Lambda$, $f(\lambda) = d = 2\lambda_1 - \lambda_2$. *We find*

$$S(\lambda) = 2(2\lambda_1 - \lambda_2 - 0.1)^2 + (\lambda_1 - 0.1)^2 + 2(\lambda_2 - 0.2)^2,$$

which is minimized by $\lambda^{MAP} = (7/50, 19/100)$. *Assuming that* $\bar{\lambda}$ *and* $\bar{d}$ *are the means of Gaussian prior and data likelihood densities with variances corresponding to the given covariance matrices, we have* $\pi_\Lambda^{post} \sim \exp(-S(\lambda))$ *with mean* λ^{MAP}. *Notice* $f(\lambda^{MAP}) = 9/100 \neq \bar{d}$. *In the data-consistent formulation, we have*

$$T(\lambda) = 2(2\lambda_1 - \lambda_2 - 0.1)^2 + \frac{1}{9}\left((\lambda_1 - 0.1)^2 + 8(\lambda_1 - 0.1)(\lambda_2 - 0.2) + 16(\lambda_2 - 0.2)^2\right),$$

minimized by $\lambda^{MUD} = (13/90, 17/90)$. *Notice* $f(\lambda^{MUD}) = \bar{d}$. *With Gaussian assumptions on the initial and observed, we have* $\pi_\Lambda^{up} \sim \exp(-T(\lambda))$ *with mean* λ^{MUD}. *Figure 1 shows plots of* λ^{MAP} *and* λ^{MUD} *along with the relevant visualizations of the terms in* S *and* T.

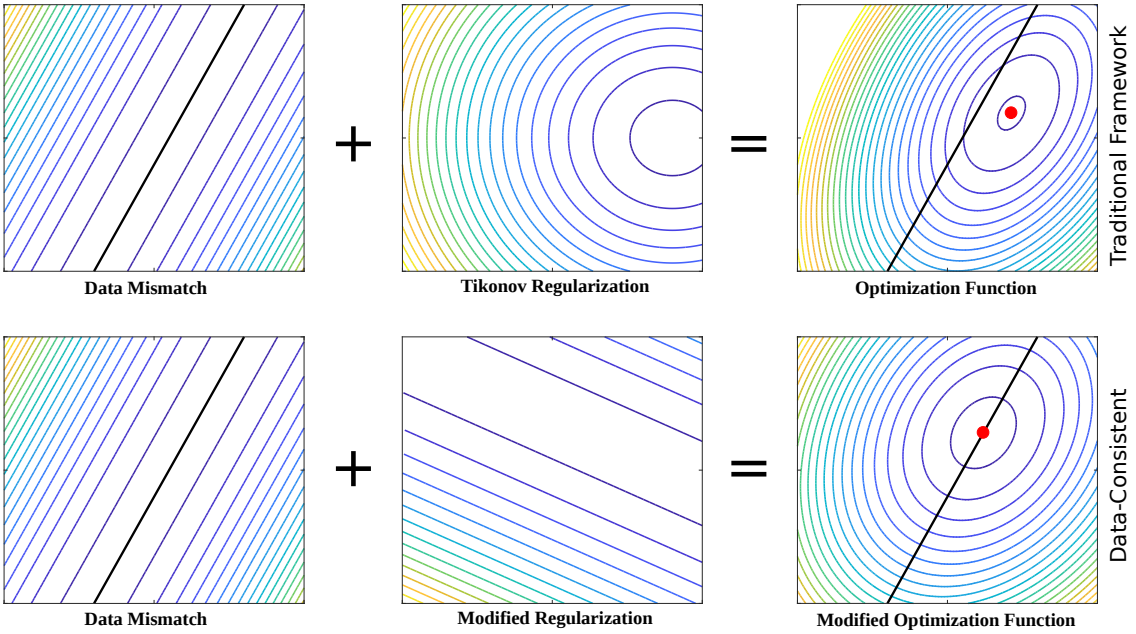


Figure 1.1: This figure from Wildey et al (2018) geometrically depicts the process of obtaining the statistical/classical Bayesian MAP point and data-consistent MUD point from solving the equivalent deterministic optimization problem – requiring Gaussian densities and f linear in Λ – in Example 2.

In the example above, we see that $\bar{d} = 0.1$ could be observed by any point λ on the line $\lambda_2 = 2\lambda_1 - 0.1$, represented by the solid black line seen in several plots in Figure 1.1 from Wildey *et al.* (2018) (where the horizontal axis is in λ_1 and the vertical axis is in λ_2). The left-hand plots in Figure 1.1 are identical, since the data mismatch term in S and T are equal. The Tikhonov regularization term is depicted in the top-middle plot, and the top-right panel shows the corresponding classical Bayesian solution with the MAP point λ^{MAP} shown as a red point and contours corresponding to the updated (Gaussian) prior distribution $c\exp(-S(\lambda))$. Just as we saw analytically in Example 2, we see that λ^{MAP} does not lie on the black line representing data-consistent solutions. The action of the modified regularization – i.e., the last two terms in (1.4.7) – is shown in the bottom-middle plot. Notice that the modified regularization is constant along directions perpendicular to the data, hence, the updated prior will be formed with prior information used only in directions that data is not informative. Thus, it is ensured that λ^{MUD} will lie on the line of data-consistent solutions, as seen in the bottom-right plot in Figure 1, where λ^{MUD} is represented as a red point and the data-consistent solution is given by contours of $\exp(-T(\lambda))$.

We have presented a deregularization term in (1.4.7) which will hold only for noise-free, linear f (i.e., $f(\lambda) = A\lambda$ for a $P \times D$ matrix A) and Gaussian assumptions on the parameter and data spaces. In Chapter IV, we make considerations for noisy $\hat{f}$ and $\hat{f}$ mildly nonlinear, which will require linearization. Then, as part of a solution process we propose, we apply our more efficient DFO methods to minimize T, obtaining and analyzing approximate MUD solutions.

CHAPTER II

ACCELERATING DFO VIA DIMENSION REDUCTION

2.1 Overview

The following chapter contains theoretical and numerical results as they relate to the main contribution of this thesis, an efficient DFO algorithm with certain automated learning features. Briefly, ASTARS blends the *Derivative Free Optimization (DFO)* algorithm called STARS Chen and Wild (2015) with parameter space dimension reduction performed via AS methods Constantine (2015); hence, we call the method *Active STARS (ASTARS)*.

We use ASTARS to solve problems in *Optimization Under Uncertainty* (OUU), which were introduced in the preceding chapter. We recall that solving OUU problems involve optimizing functions with stochastic noise, modeled either additively or multiplicatively.

Recall that since we assume $f : \Lambda \to \mathcal{D}$ with $\dim \Lambda =: P > D := \dim \mathcal{D}$; in particular, we will assume f is real-valued with $\mathcal{D} = \mathbb{R}$ in this chapter, so $D = 1$ throughout. Also recall that we assume f is a black-box function in the sense that f lacks closed form and we postulate a lack of $\nabla_\Lambda \hat{f} = \nabla_\Lambda f$ in closed form, as well.

Given $\hat{f}$ and its corresponding AS $\mathcal{A}$ (or its approximated AS $\tilde{\mathcal{A}}$), we are interested in investigating the effectiveness of optimizing $\hat{f}$ in $\mathcal{A}$. There are several approaches one may consider, and some of those approaches and their corresponding results are discussed in this chapter.

2.2 Methods

In the first section, we consider minimizing a closed-form surrogate obtained from sampling Λ, mainly for motivating key ideas used in ASTARS, as well as for comparisons to ASTARS/FAASTARS. In the second section, we introduce a technique which performing STARS in the true, known AS of the noise-free signal, f, denoted $\mathcal{A}$; we call this method Active STARS (ASTARS). In the third section, we outline Fully-Automoted ASTARS (FAASTARS), which automates the learning of hyperparamters as well as the numerically estimated AS $\mathcal{A}(\hat{f}; D_S)$ found by the Monte Carlo method discussed in Chapter I. Finally, in the fourth section, we present a technique we call Adaptive Sampling, which will update and improve estimates to $\mathcal{A}$ in FAASTARS.

2.2.1 Minimizing Surrogates

We motivate the subsequent methods in this section by first introducing a more standard approach for utilizing an AS to accelerate DFO of noisy, convex functions. One may minimize the closed-form surrogate obtained by learning an AS using Monte Carlo methods outlined in the previous chapter. Recall that in order to approximate the sensitivity matrix using Monte Carlo to detect an AS, one must

obtain a surrogate $F(\lambda)$ for f from samples in Λ in order to approximately evaluate ∇f for each sample in Λ. In particular, we use the gradient of the closed-form F for these ∇f evaluations; that is, we assume $\nabla f(\lambda) \approx \nabla F(\lambda)$ for $\lambda \in \Lambda$.

With the assumptions that F captures the behavior of f reasonably well (i.e., the correlation coefficient between these two functions approaches 1) and that F is also convex, it is straightforward to minimize f using F instead, leveraging the existence of the closed-form ∇F (which is also well-approximated by standard FD methods) to perform gradient-descent, and obtain a minimizer of F in Λ to approximately answer our convex optimization problem.

This approach is investigated in the literature, for instance by Queipo *et al.* (2005). The authors suggest an iterative approach in six main steps: fitting a surrogate to obtained f data; estimating the minimizer of the surrogate; evaluating f at that minimizer; check for convergence (stop if achieved); form updated surrogate with new data; iterate through previous steps until convergence criteria met.

In the Numerical Results section of this chapter, we will compare certain DFO methods directly to minimization via surrogate by forming a surrogate using as many samples of f as a given DFO algorithm evaluates f. For example, if a DFO method calls f 100 times, we will take 100 random samples in Λ, evaluate f at those 100 points, and fit a surrogate F through those data using a quadratic surrogate (as discussed in the previous chapter). We then apply any standard method (i.e., out-of-the-box optimization methods from a package such as SciPy) and obtain the minimizer of F, as in Queipo *et al.* (2005). Then, we may compare $\hat{f}$ at the minimizer obtained from F and any DFO routine.

We shall find that the surrogate-based approach may offer computational savings in certain cases, but is challenging and unreliable in others. Mainly, one must sample f in Λ near its minimizer; otherwise, when samples are not obtained near a minimizer, the surrogate must be extrapolated and quality is poor.

2.2.2 Active STARS (ASTARS)

Active STARS, or *ASTARS*, is a modification of STARS in which iterates only take random walks in directions lying in $\mathcal{A}$. ASTARS requires modifying the initialization and the second step of STARS, involving the draw of a random vector for exploration in Λ (see the STARS Algorithm 1). ASTARS also uses modified STARS hyperparameters, necessary for computational winnings earned by stepping in the j-dimensional $\mathcal{A}$, with $j < P$.

At iteration k, recall STARS uses random walks given by drawing a random vector $u^{(k)}$ of dimension P in which every component $u_i^{(k)}, i = 1, \ldots, P$ of $u^{(k)}$ is drawn from a specified normal distribution. Instead, given the first j eigenvectors $v^1, \ldots, v^j$ corresponding to the eigendecomposition of the sensitivity matrix W, ASTARS takes j draws from a specified normal distribution, denoted $r_i \sim N(0, \omega_i^2)$, $i = 1, \ldots, j$, defining the random vector $u_{\mathcal{A}}^{(k)}$ for the k-th random *active direction* as

$$u_{\mathcal{A}}^{(k)} = \sum_{i=1}^{j} r_i \tilde{v}^i, \quad r_i \sim N(0, \omega_i^2), \quad i = 1, \ldots, j, \tag{2.2.1}$$

a randomly weighted linear combination of the active directions of $\hat{f}$. In ASTARS (Algorithm 2 below), we equally weight the j active directions, in the sense that $\omega_i^2 = 1$ for $i = 1, \ldots, j$, matching the theoretical assumption in Chen and Wild (2015) needed to prove converge results later in this chapter. We present alternative weighting schemes in Chapter 3 in our considerations of extensions to the ASTARS method. Note in the case that there is not a large drop-off in the spectrum of W, then all P directions are treated as active variables, and performing ASTARS is equivalent performing STARS.

The convergence of STARS is dimension dependent, borne out in both the complexity results in Chen and Wild (2015) and perhaps more clearly (but less formally) by inspection of its hyperparameters. Recall the stepsize and smoothing factor are both inversely proportional to $\dim \Lambda = P$, meaning that as P increases, the STARS stepsize and smoothing factor both decrease, leading to small, cautious exploration steps and descent steps in the large parameter space. For the additive noise OUU problem, we define the *active hyperparameters* $\mu_{\mathcal{A}}^*$ and $h_{\mathcal{A}}$

$$\mu_{\mathcal{A}}^* := \left(\frac{8\sigma^2 j}{L_1^2 (j+6)^3} \right)^{1/4} \qquad h_{\mathcal{A}} := (4L_1(j+4))^{-1}, \tag{2.2.2}$$

the active smoothing factor and step length, respectively. For the multiplicative noise OUU problem, the step length $h_{\mathcal{A}}$ remains the same, exactly as in (2.2.2) above but the optimal *active smoothing factor* for the k-th iterate of ASTARS, $k = 1, \ldots, M$, is given by

$$(\mu_{\mathcal{A}}^*)_k := \left(\frac{16\sigma^2 \hat{f}(\lambda^{(k)})^2 j}{L_1^2 (1+3\sigma^2)(j+6)^3} \right)^{1/4}. \tag{2.2.3}$$

We may use $\mu_{\mathcal{A}}^*$ to generally denote and discuss the active smoothing factor hereon with the understanding that in the case of multiplicative noise one actually uses $(\mu_{\mathcal{A}}^*)_k$.

The remainder of the ASTARS routine is identical to STARS, but uses the random active direction $u_{\mathcal{A}}$ in place of u and the active hyperparameters in place of the STARS hyperparameters presented in the last chapter. In Algorithm 2 below, we present ASTARS.

ASTARS Corollary: *Let the vectors $u_{\mathcal{A}}^{(k)}$ denote those drawn using Algorithm 2 (zero mean, unit variance in each component); let $f \in \mathcal{C}^{1,1}(\Lambda)$ and assume f is convex; and assume that the i.i.d. noise draws $\epsilon(\xi)$ are additive, zero mean, with bounded variance σ^2 for all ξ. Fixing the step size $h_{\mathcal{A}}$ in* (2.2.2)*, the active smoothing parameter $\mu_{\mathcal{A}}^*$ in* (2.2.2) *minimizes the error between the gradient oracle in*

Algorithm 2: ASTARS

Input: `maxit`$=: M$; $\lambda^{(0)}$; $f_0 := \hat{f}(\lambda^{(0)})$; $h_{\mathcal{A}}$; $\tilde{V}_{\mathcal{A}} := \tilde{V}_{1:P,1:j}$; $k = 1$

while $k \leq M$ **do**

- Form smoothing factor $(\mu^*_{\mathcal{A}})_k$;
- Draw $r^{(k)}$, where $r_p^{(k)} \sim N(0,1)$ for $p = 1, \ldots, j$ and set $u_{\mathcal{A}}^{(k)} := \tilde{V}_{\mathcal{A}} r^{(k)}$;
- Evaluate $g_k := \hat{f}(\lambda^{(k-1)} + (\mu^*_{\mathcal{A}})_k u_{\mathcal{A}}^{(k)})$;
- Set $d^{(k)} := \frac{g_k - f_{k-1}}{(\mu^*_{\mathcal{A}})_k} u_{\mathcal{A}}^{(k)}$;
- Set $\lambda^{(k)} = \lambda^{(k-1)} - h_{\mathcal{A}} \cdot d^{(k)}$;
- Evaluate $f_k := \hat{f}(\lambda^{(k)})$;
- Set $k = k + 1$;

end

Output: $(\lambda^{(M)}, f_M)$

Algorithm 2, $\frac{\hat{f}(\lambda^{(k-1)} + \mu^*_{\mathcal{A}} u_{\mathcal{A}}^{(k)}) - \hat{f}(\lambda^{(k-1)})}{\mu^*_{\mathcal{A}}} u_{\mathcal{A}}^{(k)}$*, and the true directional derivative of f in the direction $u_{\mathcal{A}}^{(k)}$ in the j-dimensional AS $\mathcal{A}$.*

Remark: The preceding corollary implies that using the fixed step length $h_{\mathcal{A}}$, ASTARS uses an optimal choice of smoothing parameter $\mu^*_{\mathcal{A}}$, in the sense that $\mu^*_{\mathcal{A}}$ minimizes the error between our approximate directional derivative formed in Algorithm 2. Since ASTARS takes steps in the j-dimensional space $\mathcal{A}$, the hyperparameters from STARS must be redefined to remain optimal. The hyperparameters in (1.3.3) and (1.3.4) are proven to be optimal hyperparameters for the convergence of STARS in the P-dimensional space Λ by the authors in Theorem 4.3 of Chen and Wild (2015). Now, replacing $P = \dim \Lambda$ in (1.3.3) and (1.3.4) with $j = \dim \mathcal{A}$, we obtain (2.2.2) and (2.2.3). We will present a proof of the corollary above as ASTARS Corollary 2 in 2.3.3. Note also that the authors Chen and Wild (2015) have an analogous result for the case of multiplicative noise, and so we note that we, too, have an optimal active smoothing constant given in (2.2.3) in the case of multiplicative noise.

In 2.3.3 we also prove that the complexity of ASTARS is $M \sim \mathcal{O}\left(\frac{L_1 j R_{\mathcal{A}}^2}{\epsilon_{\text{tol}}}\right)$. Here, $R_{\mathcal{A}}^2$ is a bound on the squared norm of the difference between any initial iterate and the true minimum of f, both projected in the inactive subspace $\mathcal{I}$. $\epsilon_{\text{tol}} > 0$ is a final accuracy which is bounded below by terms involving the variance in the noise, as well as by terms involving the inactive subspace of f; for details, refer to 2.3.3.

2.2.3 Fully-Automated ASTARS (FAASTARS)

We now introduce a fully-automated version of ASTARS (Algorithm 2), in the sense that the user need not specify anything beyond an initial iterate $\lambda^{(0)}$, its evaluation $\hat{f}(\lambda^{(0)})$, the black-box objective function $\hat{f}$, and a maximum number of iterations, M. We call this algorithm *Fully-Automated ASTARS (FAASTARS).*

We first note that $\hat{f}$ need not be in closed form; we only require that $\hat{f}$ is a callable function, which we recall may actually represent a post-processed quantity from, for instance, a (noisy) solution of a PDE evaluated at some point in parameter (phase) space. FAASTARS estimates σ^2 and L_1 from a handful of upfront samples taken from performing ECNoise (and recycled for L_1 learning). As well, $\mathcal{A}$ is estimated from regular STARS iterates (supplemented with the original ECNoise samples as well) during a *STARS burn-in* phase.

In the following, we outline FAASTARS by breaking it down into three phases: (1.) a hyperparameter learning phase; (2.) a regular STARS burn-in phase; and (3.) an approximate ASTARS phase.

In the first phase, the estimator $\hat{\sigma}^2$ will be formed immediately by using ECNoise on 7 to 10 sampled points in Λ. As discussed above, the samples s^i created by ECNoise lend themselves to forming L_1^{init} in a finite-difference scheme approximating directional derivatives and $\hat{L_1} = L_1^{\text{init}}$ is used.

We can also form a surrogate F from the ECNoise points to estimate L_1. By obtaining the value of the closed-form hessian of the surrogate F, $\nabla^2 F$, we obtain a lower bound L_1, as we will see. The surrogate can be improved by incorporating STARS iterates into the set of samples used to form the surrogate, which we will present in the next section, about adaptive sampling.

We will first need approximated hyperparameters, given by

$$\hat{\mu}^* := \left(\frac{8\hat{\sigma}^2 P}{\hat{L_1}^2 (P+6)^3} \right)^{1/4} \qquad \hat{h} = (4\hat{L_1}(P+4))^{-1} \tag{2.2.4}$$

Note that the approximated active smoothing factor will be optimal in Λ – given the available information – in a somewhat analogous fashion as our result in ASTARS Corollary 2, but with a specified loss of optimality as the estimates $\hat{L_1}$ and $\hat{\sigma}$ stray from their true values. (See STARS Theorem 4.3 (Modified) in 2.3.2.) In particular, we find that STARS will either diverge or make no progress when the values are underestimated or overestimated, respectively. That is, underestimation of either value may lead to STARS' divergence, but for distinct reasons in each case.

When the variance in the noise σ^2 is underestimated, μ^* is also underestimated, and thus we may not take a large enough step to successfully perturb $\hat{f}$ enough to see a change in function value larger than the noise level itself, leading to inaccurate derivative information and potentially descent steps of poor quality. (Note σ^2 does not appear in the step size.) When the gradient Lipschitz constant L_1 is underestimated, both μ^* and h will be too large, meaning we may take too large of a step for the finite difference approximation to be accurate *and* we may take too large of a descent step (in a bad direction), causing a quick rise in the function valeus we see. Indeed, underestimating L_1 is to be avoided at all costs.

We may also form a linear surrogate F from ECNoise samples (if we have enough data) to initiate our approximation to L_1 by computing the matrix norm of the closed form Hessian of F, $||\nabla^2 F||$, for which we have $||\nabla^2 F|| \approx ||\nabla^2 f|| \leq L_1$. The first phase of FAASTARS is given in the algorithm below.

Algorithm 3: FAASTARS for Additive or Multiplicative OUU, Phase 1: Hyperparameter Learning

Input: $\lambda^{(0)}$; $k = 0$
while $k = 0$ **do**
  Run ECNoise using $\lambda^{(0)}$ as base point and obtain $\hat{\sigma}^2$;
  Initialize storage array D_S for samples of f and store $\{s^i\}$ formed by ECNoise;
  Use $\{s^i\}$ to form second-order FD approximation (or form linear surrogate F if $S > P + 1$ to compute $||\nabla^2 F||$) for L_1^{init};
end
Output: $\hat{\sigma}^2$; L_1^{init}; D_S

Next, in the second phase, we perform standard STARS (Algorithm 1) until enough samples are obtained to perform AS analysis via a surrogate to form needed ∇f evaluations. We let $M_{\mathcal{A}}$ denote the number of iterations needed to find the AS from samples, as we see in the first phase of FAASTARS above. We note that $M_{\mathcal{A}}$ will depend on the type of surrogate formed (e.g., using local linear versus global quadratic versus RBFs). FAASTARS will not begin its ASTARS routine until enough samples have been taken to form a surrogate, based on the chosen surrogate method – RBFs are the default surrogate method when none is provided using known formulas. For example, if one wishes to use a globally quadratic surrogate, $\frac{(P+1)(P+2)}{2}$ samples of f are required Smith (2013). Recalling that every STARS step requires *two* evaluations of f, $M_{\mathcal{A}} = \frac{(P+1)(P+2)}{4}$ will suffice for quadratic surrogates. $M_{\mathcal{A}}$ is not a value the user has to provide; FAASTARS will automatically terminate its STARS burn-in period as soon as k is large enough for the chosen or default surrogate to be formed.

After all steps of approximate STARS in Phase 2 are taken, $\tilde{\mathcal{A}}$ is found using the Monte Carlo method described in Chapter 1 using the samples/iterates $\{s^i\}$ collected from both ECNoise and $M_{\mathcal{A}}$ steps of standard STARS. We present the second phase of FAASTARS in the pseudocode below.

Note that at the end of each standard STARS iteration in the burn-in phase, we have the functionality to form candidates L_1^{update} for $\hat{L_1}$ via finite difference (FD) approximations to the directional derivatives in the direction of the corresponding descent step. We can also use a surrogate as in Phase 1. Regardless, we reject the candidate update anytime $L_1^{\text{update}} \leq L_1^{\text{init}}$, since we are always searching for the most pessimistic bound to L_1 to avoid divergence.

With $\tilde{\mathcal{A}}$ in hand, we must first update the hyperparameters so that they are computed with the value $\tilde{j} = \dim \tilde{\mathcal{A}}$ (and not $j = \dim \mathcal{A}$, since it is generally unknown in this setting). We define the approximated active hyperparamters,

Algorithm 4: FAASTARS for Additive or Multiplicative OUU, Phase 2: Approximate STARS burn-in

Input: $\lambda^{(0)}$; $f_0 := \hat{f}(\lambda^{(0)})$; $k = 1$; Surrogate method (optional, default is RBFs); $\hat{\sigma}^2$; L_1^{init}; D_S; eigenvalue threshold $0 < \tau < 1$ (optional, default is $\tau = 0.95$)

Determine $M_{\mathcal{A}}$ based on chosen surrogate method or RBFs if none is provided;

Form step length $\hat{h}$ using $\hat{\sigma}^2$ and L_1^{init}

while $1 \leq k \leq M_{\mathcal{A}}$ **do**

- Form smoothing factor $\hat{\mu}_k^*$ using $\hat{\sigma}^2$ and L_1^{init};
- Find $\lambda^{(k)}$ and evaluate $f_k := \hat{f}(\lambda^{(k)})$ via STARS (Algorithm 1);
- Store $(\lambda^{(k-1)} + \hat{\mu}_k^* u^{(k)}, g_k)$ and $(\lambda^{(k)}, f_k)$ as samples $\{s^i\}$ in D_S;
- **Optional:** Form $(L_1^{\text{update}})_k$ via FD with D_S (or use $||\nabla^2 F||$ for surrogate F with D_S);
- **Optional:** If $(L_1^{\text{update}})_k > L_1^{\text{init}}$, set $L_1^{\text{init}} = (L_1^{\text{update}})_k$ and re-compute h;
- Set $k = k + 1$;

Use D_S to form surrogate via selected method;

Form $\tilde{W}$, and apply SVD to obtain $\tilde{W} = \tilde{V}\tilde{Q}\tilde{V}^\top$;

Find a drop-off index $\tilde{j}$ for which $\tilde{q}_1 + \cdots + \tilde{q}_{\tilde{j}} \geq \tau(\tilde{q}_1 + \cdots + \tilde{q}_P)$

Set $\tilde{V}_{\tilde{\mathcal{A}}} := \tilde{V}_{1:P,1:\tilde{j}}$ and form $h_{\mathcal{A}}$ using $\tilde{j}$ for $\dim \tilde{\mathcal{A}}$; $\hat{\sigma}^2$ and L_1^{init};

Output: $\lambda^{(M_{\mathcal{A}})}$, $f_{M_{\mathcal{A}}} := \hat{f}(\lambda^{(M_{\mathcal{A}})})$, $\tilde{V}_{\tilde{\mathcal{A}}}$

$$\hat{\mu}_{\tilde{\mathcal{A}}}^* := \left(\frac{8\hat{\sigma}^2 \tilde{j}}{\hat{L_1}^2 (\tilde{j}+6)^3} \right)^{1/4} \qquad \hat{h}_{\tilde{\mathcal{A}}} := (4\hat{L_1}(\tilde{j}+4))^{-1}. \tag{2.2.5}$$

In the third phase, we pick up where standard STARS left off, and we perform ASTARS in the approximated AS for the remaining iterations until the maximum number of iterations, M, is met. We have the functionality to also update $\hat{L_1}$ in a similar fashion as the burn-in phase during the ASTARS phase. That is, we may continue to use finite differences, or use the surrogate F that we form for AS approximations to update our approximation to L_1. (In practice, this surrogate could be formed from the initial ECNoise points and also be recalculated at every step as we gain more samples of $\hat{f}$.) We may take $L_1^{\text{update}} = ||\nabla^2 F||$ and similarly to above reject the candidate update anytime $L_1^{\text{update}} \leq L_1^{\text{init}}$; otherwise, we have a new initial L_1, where $L_1^{\text{init}} = ||\nabla^2 F||$.

We now have all the necessary components for performing FAASTARS. In summary, FAASTARS has three major phases: (1.) an initial, relatively inexpensive learning phase where we acquire the estimates $\hat{\sigma}^2$ and L_1^{init}; (2.) a STARS burn-in phase (in full variables) where we acquire enough samples to compute an AS using the Monte Carlo methods above; and (3.) an ASTARS phase, where we use the learned AS, $\tilde{\mathcal{A}}$. Note that in both of the latter phases, we will update $\hat{L_1}$, if and only if we obtain a more pessimistic (larger) estimate.

In 2.3.4 we show that the approximately active hyperparameters (2.2.5) are chosen to minimize the error in finite difference approximations to directional derivatives. We also prove that the complexity

Algorithm 5: FAASTARS for Additive or Multiplicative OUU, Phase 3: Approximate ASTARS

Input: `maxit`=: M; $\lambda^{(M_{\mathcal{A}})}$; $f_{M_{\mathcal{A}}} := \hat{f}(\lambda^{(M_{\mathcal{A}})})$; $k = M_{\mathcal{A}} + 1$; $\hat{\sigma}^2$; L_1^{init}; $\tilde{V}_{\tilde{\mathcal{A}}}$

while $M_{\mathcal{A}} < k \leq M$ **do**

- Form step size $\hat{h}_{\tilde{\mathcal{A}}}$ and smoothing factor $\hat{\mu}^*_{\tilde{\mathcal{A}}}$ using $\tilde{j}$ for $\dim \tilde{\mathcal{A}}$; $\hat{\sigma}^2$ and L_1^{init};
- Draw $r^{(k)}$, where $r_p^{(k)} \sim N(0,1)$ for $p = 1, \ldots, \tilde{j}$ and set $u^{(k)}_{\tilde{\mathcal{A}}} := \tilde{V}_{\tilde{\mathcal{A}}} r^{(k)}$;
- Evaluate $g_k := \hat{f}(\lambda^{(k-1)} + \hat{\mu}^*_{\tilde{\mathcal{A}}} \cdot u^{(k)}_{\tilde{\mathcal{A}}})$;
- Set $d^{(k)} := \frac{g_k - f_{k-1}}{\hat{\mu}^*_{\tilde{\mathcal{A}}}} u^{(k)}_{\tilde{\mathcal{A}}}$;
- Set $\lambda^{(k)} = \lambda^{(k-1)} - \hat{h}_{\tilde{\mathcal{A}}} \cdot d^{(k)}$;
- Evaluate $f_k := \hat{f}(\lambda^{(k)})$;
- **Optional:** Update L_1^{init} with FD or surrogates as in Phase 2 (requires updating D_S);
- Set $k = k + 1$;

Output: $(\lambda^{(M)}, f_M)$

of FAASTARS is similar to that of ASTARS, but with a specified inflation of certain bounds, which arise from approximating both hyperparameters and $\mathcal{A}$.

2.2.4 Active Subspace Retraining

We introduce a method we call *active subspace retraining* which can be optionally applied in place of Phase 3 FAASTARS. Active subspace retraining begins with an initial approximation $\tilde{\mathcal{A}}$ for $\mathcal{A}$, which is obtained once the minimum number of samples $M_{\mathcal{A}}$ are gathered during FAASTARS Phase 2. We then take $k_T \geq 1$ approximate ASTARS steps using $\tilde{\mathcal{A}}$ (as in FAASTARS Phase 3) and recompute $\tilde{\mathcal{A}}$ with our new set of samples of f. We continue in this fashion until the maximum iteration count M is reached.

Samples obtained during FAASTARS Phase 3 contain information which is more local to a given Phase 3 iterate. Hence, adaptive sampling incorporates local information to approximate $\mathcal{A}$, more aligned with how f changes in the region of recent samples. It is almost always better to include more samples whenever we can to improve $\tilde{A}$, borne out in numerical results. Indeed, we use active subspace retraining to produce all FAASTARS results in this work. We present the adaptive sampling algorithm below.

Algorithm 6: Active Subspace Retraining

Input: `maxit`=: M; $\lambda^{(M_{\mathcal{A}})}$; $f_{M_{\mathcal{A}}} := \hat{f}(\lambda^{(M_{\mathcal{A}})})$; re-training phase length k_T; D_S; $\hat{\sigma}^2$; L_1^{init}; $\tilde{V}_{\tilde{\mathcal{A}}}$ from FAASTARS Phase 2; $l = 1$; $k = M_{\mathcal{A}} + 1$;

while $k \leq M_{\mathcal{A}} + l k_T$ **do**

- Form approximate active step size $\hat{h}_{\tilde{\mathcal{A}}}$ and approximate active smoothing factor $(\hat{\mu}^*_{\tilde{\mathcal{A}}})_k$;
- Take k_T steps of (approximate) ASTARS with $\tilde{\mathcal{A}}$ (as in FAASTARS Phase 3) and store all samples of f in D_S;
- Retrain $\tilde{\mathcal{A}}$ and recompute $\tilde{V}_{\tilde{\mathcal{A}}}$ with D_S (containing k_T new samples);
- Set $l = l + 1$ and exit loop only if $k_S + l k_T \geq M$;

end

Output: $(\lambda^{(M)}, f_M)$

At times, a challenge with adaptive sampling is poor quality in the samples obtained from the FAASTARS Phase 3 steps, after the original formation of $\tilde{A}$. When the step size is small and f changes very little – sometimes due to inaccurate descent directions – samples are generally uninformative about the behavior of f. This challenge is also present, more generally, in all AS approximations involving samples of $\hat{f}$ taken in a partially deterministic DFO algorithm. Again, these samples are often clustered together in Λ due to the small steps we take both in finite differencing and in a descent step. Some of these challenges are addressed in the next chapter. Regardless, incorporating as many samples as possible for computations involving $\mathcal{A}$ usually improves numerical results.

2.3 Theoretical Results

We present theoretical results regarding the convergence of the methods provided in the preceding section. In the first part of this section, we provide key results needed for proofs in the remaining parts. In the second part of this section, we prove a series of modified STARS results culminating in a statement about the convergence of the algorithm with approximate hyperparameters. In the third part, we prove a series of results showing the convergence of ASTARS with exact hyperparameters and an exact AS. Finally, we prove a series of FAASTARS results culminating in a statement about the convergence of FAASTARS with approximate hyperparamters and with an approximate AS.

Broadly, our contribution is showing: (1) STARS will still converge if L_1 and σ are unknown and replaced with estimates $\hat{L_1}$ and $\hat{\sigma}$ in the formation of STARS hyperparameters; (2) ASTARS will converge with exact hyperparameters and an exact AS; and (3) FAASTARS will analogously converge also with uncertain hyperparameters, and even with an approximated $\tilde{\mathcal{A}}$ in place of the true $\mathcal{A}$.

2.3.1 Preliminaries

We provide the equations and results needed from Nesterov and Spokoiny (2017), which are also summarized in Chen and Wild (2015). We also modify certain key results needed for ASTARS theoretical arguments.

We focus only on the case of additive noise in $\hat{f}$ and we assume f is convex and differentiable in Λ. Recall that $\Lambda = \mathbb{R}^P$ with λ's denoting vectors, $\lambda \in \Lambda$; at times $u, x, y, z \in \Lambda$ will denote vectors, too. Also, we note that for the FAASTARS , we leave out the optional steps in FAASTARS related to the updating of L_1^{init}. We will fix our approximation to L_1 at the beginning of FAASTARS using the samples formed via ECNoise Moré and Wild (2015); i.e., $\hat{L_1} = L_1^{\text{init}}$ throughout. We also note that we will not consider adaptive sampling methods for learning $\tilde{\mathcal{A}}$, so $\tilde{\mathcal{A}}$ will be fixed after FAASTARS Phase 2 (STARS burn-in phase).

We assume the true signal f is convex and differentiable – this is Assumption 4.1 in Chen and Wild (2015). The signal we access is $\hat{f}$, which has additive noise so $\hat{f}(\lambda;\xi) := f(\lambda)+\epsilon(\xi)$, where $\mathbb{E}_\xi(\epsilon(\xi)) = 0\ \forall\xi$ and $0 < \text{Var}_\xi(\epsilon(\xi)) = \sigma^2 < \infty\ \forall\xi$. These assumptions make up Assumption 4.2 in Chen and Wild (2015). First, let

$$f_\mu(\lambda) := \mathbb{E}_u[f(\lambda+\mu u)],\ u\in\mathbb{R}^P,\ u_i \sim N(0,1),\ i=1,\dots,P, \tag{2.3.1}$$

which is the expectation of the Gaussian-smoothed form of f.

Also, note for a direction $u = u_{\mathcal{A}} \in \mathcal{A} \subset \mathbb{R}^P$, the last $P-j$ components of $u_{\mathcal{A}}$ are zero and $u_i \sim N(0,1)$ for $i=1,\dots,j$. Hence, (2.3.1) becomes

$$f_\mu^{\mathcal{A}}(\lambda) = \mathbb{E}_{u_{\mathcal{A}}}[f(\lambda+\mu u_{\mathcal{A}})],\ u_{\mathcal{A}}\in\mathcal{A}\subset\mathbb{R}^P,\ u_i\sim N(0,1),\ i=1,\dots,j, \tag{2.3.2}$$

and similarly for $\tilde{\mathcal{A}}$ with $\dim\tilde{\mathcal{A}} = \tilde{j}$.

The existence of L_1 implies

$$|f(y)-f(x)-\langle\nabla f(x), y-x\rangle| \le \frac{L_1}{2}||x-y||^2 \quad \forall x,y\in\Lambda, \tag{2.3.3}$$

Now let λ^* denote a global minimizer of f. Then

$$||\nabla f(\lambda)||^2 \le 2L_1(f(\lambda)-f(\lambda^*)) \quad \forall\lambda\in\Lambda, \tag{2.3.4}$$

proven in Zhou (2017). A differentiable function f is convex iff

$$f(y) \ge f(x)+\langle\nabla f(x), y-x\rangle \quad \forall x,y\in\Lambda, \tag{2.3.5}$$

Note that this implies that the left-hand side of (2.3.3) is nonnegative, so long as f is convex. An interpretation of (2.3.5) is that a convex function is always underestimated by its linear approximation.

We now present the needed results on Gaussian smoothing from Nesterov and Spokoiny (2017), also presented in Chen and Wild (2015). First, some notation; for $\mu > 0$ and u a Gaussian random vector (as in (2.3.1) or (2.3.2)),

$$g_\mu(x) := \frac{f(x+\mu u)-f(x)}{\mu}u, \tag{2.3.6}$$

which is a first-order approximation to the directional derivative of f in the direction of u. If $u = u_{\mathcal{A}}$, then $g_\mu^{\mathcal{A}}$ will be the same object as (2.3.6), but with $u = u_{\mathcal{A}}$. Then $g_\mu^{\mathcal{A}}$ estimates the directional derivative of f for directions $u_{\mathcal{A}}$ strictly in $\mathcal{A}$.

Now for $p \geq 0$, let

$$M_p(u) := \mathbb{E}_u \left(||u||^p \right), \tag{2.3.7}$$

the p-th moment of the norm of the random vector u, $||u||$. Here and throughout this section, $|| \cdot ||$ denotes the 2-norm in $\mathbb{R}^P$. Let u continue to denote a random vector as in (2.3.1). For $M_p(u)$ defined in (2.3.7) above,

$$M_p(u) \leq (P)^{p/2}, \quad p \in [0, 2] \quad \text{and} \quad M_p(u) \leq (P + p)^{p/2}, \quad p > 2, \tag{2.3.8}$$

where we recall $P = \dim \Lambda$.

Now for $u = u_{\mathcal{A}}$ (as in (2.3.2)) and $j < P$, we have a sharper but analogous result for these moments, which we explain by considering the case of $p = 2$. From Nesterov and Spokoiny (2017), we have

$$M_2(u) = \frac{1}{c} \int_{\mathbb{R}^P} ||u||^2 \, e^{-\frac{1}{2}||u||^2} \, du = B^{-1}, \tag{2.3.9}$$

where B is the matrix specifying the norm $||u||^2 = \langle Bu, u \rangle$ for a given inner product $\langle \cdot, \cdot \rangle$ in $\mathbb{R}^P$ and c is a normalization factor. Throughout, we will let $B = I_P$ and use the Euclidean inner product, so that $|| \cdot ||$ will continue to denote the standard (Euclidean) 2-norm in Λ.

Note that since $u_{\mathcal{A}} \in \mathcal{A}$, we only integrate over the j parameters corresponding to $\mathcal{A}$ since the components of $u_{\mathcal{A}}$ in $\mathcal{I}$ are fixed and zero in expectation. Also, since $u_{\mathcal{A}}$ is zero in its last $P - j$ components, we can compute the norm of $u_{\mathcal{A}}$ truncated after its j-th entry in $\mathbb{R}^j$ instead of in $\mathbb{R}^P$. In particular, define $\underline{u_{\mathcal{A}}} = (u_{\mathcal{A}})_{1:j}$ and let $|| \cdot ||_{\underline{\mathcal{A}}}$ denote the 2-norm in $\mathbb{R}^j$ We have $||u_{\mathcal{A}}|| = ||\underline{u_{\mathcal{A}}}||_{\underline{\mathcal{A}}}$, and therefore

$$M_2(u_{\mathcal{A}}) = \frac{1}{c} \int_{\mathbb{R}^j} ||\underline{u_{\mathcal{A}}}||_{\underline{\mathcal{A}}}^2 \, e^{-\frac{1}{2}||\underline{u_{\mathcal{A}}}||_{\underline{\mathcal{A}}}^2} \, d\underline{u_{\mathcal{A}}} = I_j. \tag{2.3.10}$$

As in Nesterov and Spokoiny (2017), taking the inner product of the left-hand and right-hand sides with I_j shows $M_2(u_{\mathcal{A}}) = j$. Using similar arguments, one can prove the generalized bounds

$$M_p(u_{\mathcal{A}}) \leq (j)^{p/2}, \quad p \in [0, 2] \quad \text{and} \quad M_p(u_{\mathcal{A}}) \leq (j + p)^{p/2}, \quad p > 2, \tag{2.3.11}$$

which involves making similar changes to those outlined above. In particular, one must substitute u with $\underline{u_{\mathcal{A}}}$ and $||\cdot||$ with $||\cdot||_{\underline{\mathcal{A}}}$ and rewrite the integrals corresponding to the expectation over all $\mathbb{R}^P$ as integrals over $\mathbb{R}^j$, as we presented for M_2.

We now present Gaussian smoothing results in the case for f convex. If f is convex, then

$$f_\mu(x) \geq f(x) \quad \forall x \in \Lambda, \tag{2.3.12}$$

which can be verified by writing $\mathbb{E}_u(f(x+\mu u)) \geq \mathbb{E}_u(f(x) + \langle \nabla f(x), \mu u \rangle)$, where the inequality arises from applying the definition of convexity ((2.3.5) with $y = x + \mu u$ and $x = x$). Then since $f(x)$ is constant with respect to u and $\mathbb{E}_u(\langle \nabla f(x), \mu u \rangle) = 0$ (since each component of u is zero-mean), we obtain $\mathbb{E}_u(f(x) + \langle \nabla f(x), \mu u \rangle) = f(x)$, verifying (2.3.12). Also note that (2.3.12) holds with $f_\mu = f_\mu^{\mathcal{A}}$ (as in (2.3.2)).

If f is convex and $f \in \mathcal{C}^{1,1}(\Lambda)$, which is the space of functions $f : \Lambda = \mathbb{R}^P \to \mathbb{R}$ that are continuously differentiable and possess a L_1 Lipschitz constant, then

$$f_\mu(x) - f(x) \leq \frac{L_1 \mu^2}{2} P \quad \forall x \in \Lambda, \tag{2.3.13}$$

where we recall (2.3.12) implies the left-hand side of (2.3.13) above is nonnegative.

To verify (2.3.13), we use the fact that f is convex (i.e., (2.3.5) with $y = x + \mu u$ and $x = x$) to write

$$0 \leq f(x + \mu u) - f(x) - \langle \nabla f(x), \mu u \rangle.$$

Applying (2.3.3),

$$f(x + \mu u) - f(x) - \langle \nabla f(x), \mu u \rangle \leq \frac{L_1 \mu^2}{2} ||u||^2.$$

Applying the expectation in u to both sides and using (2.3.8) for the second moment of $||u||$, we obtain (2.3.13). If $u = u_{\mathcal{A}}$ and we have $f_\mu^{\mathcal{A}}$ (as in (2.3.2)), then using (2.3.11),

$$f_\mu^{\mathcal{A}}(x) - f(x) \leq \frac{L_1 \mu^2}{2} j \quad \forall x \in \Lambda. \tag{2.3.14}$$

We will also work with the objects $\nabla f_\mu(x)$ and $\nabla f_\mu^{\mathcal{A}}(x)$. To quote Chen and Wild (2015), we note that with "$\nabla f_\mu(x)$ we denote the gradient (with respect to x) of the Gaussian approximation" $f_\mu(x)$ defined in (2.3.1) and likewise for $\nabla f_\mu^{\mathcal{A}}(x)$, for $f_\mu^{\mathcal{A}}$ as in (2.3.2). In Nesterov and Spokoiny (2017), the authors obtain the form in (2.3.15) below for ∇f_μ by performing a substitution in the integral corresponding to the expectation in u arising in the definition of f_μ and differentiate both sides in x, yielding

$$\nabla f_\mu(x) = \frac{1}{c}\int_{\mathbb{R}^P} \frac{f(x+\mu u) - f(x)}{\mu} u e^{-\frac{1}{2}||u||^2}\, du = \frac{1}{c}\int_{\mathbb{R}^P} g_\mu(x) e^{-\frac{1}{2}||u||^2}\, du, \tag{2.3.15}$$

which shows by definition

$$\nabla f_\mu(x) = \mathbb{E}_u\left(g_\mu(x)\right) \quad \forall x \in \Lambda \tag{2.3.16}$$

Analogously, we have

$$\nabla f_\mu^{\mathcal{A}}(x) = \mathbb{E}_u\left(g_\mu^{\mathcal{A}}(x)\right) \quad \forall x \in \Lambda \tag{2.3.17}$$

by taking $u = u_{\mathcal{A}}$ in our arguments above.

If f is differentiable at x and $f \in \mathcal{C}^{1,1}$,

$$\mathbb{E}_u\left(||g_\mu(x)||^2\right) \leq 2(P+4)||\nabla f(x)||^2 + \frac{\mu^2}{2}L_1^2(P+6)^3 \quad \forall x \in \Lambda. \tag{2.3.18}$$

To verify (2.3.18), it is helpful to first bound the quantity $\mathbb{E}_u\left(||u||^2\langle \nabla f(x), u\rangle^2\right)$. We follow Nesterov and Spokoiny (2017). Applying Cauchy-Schwarz and write $||u||^2\langle \nabla f(x), u\rangle^2 \leq ||\nabla f(x)||^2||u||^4$. Here we could apply $\mathbb{E}_u(\cdot)$ and use (2.3.8) to bound $M_4(u)$, yielding

$$\mathbb{E}_u\left(||u||^2\langle \nabla f(x), u\rangle^2\right) \leq (P+4)^2||\nabla f(x)||^2.$$

However, the authors in Nesterov and Spokoiny (2017) obtain a tighter bound by minimizing the integral form of $\mathbb{E}_u\left(||u||^2\langle \nabla f(x), u\rangle^2\right)$. The proof is technical but mainly involves parameterizing and then minimizing the argument of the exponential function appearing in the integral form associated with the expectation over u. Nesterov and Spokoiny (2017) show

$$\mathbb{E}_u\left(||u||^2\langle \nabla f(x), u\rangle^2\right) \leq (P+4)||\nabla f(x)||^2.$$

For the case of $g_\mu^{\mathcal{A}}$, we can show

$$\mathbb{E}_u\left(||g_\mu(x)||^2\right) \leq 2(j+4)||\nabla f(x)||^2 + \frac{\mu^2}{2}L_1^2(j+6)^3 \quad \forall x \in \Lambda. \tag{2.3.19}$$

Justifying (2.3.19) requires the same substitutions as in justifying (2.3.11). Again, one must substitute u with $\underline{u_{\mathcal{A}}}$ and $||\cdot||$ with $||\cdot||_{\underline{\mathcal{A}}}$ and rewrite the integrals corresponding to the expectation over all $\mathbb{R}^P$ as integrals over $\mathbb{R}^j$. Then P's in the arguments above may be safely replaced with j's as in (2.3.19).

Next, the authors in Chen and Wild (2015) introduce two more pieces of notation integral to the STARS process. First, for an iterate k, let

$$s_{\mu_k}(x^{(k)}; u^{(k)}, \xi_{k-1}, \xi_k) := \frac{\hat{f}(x^{(k)} + \mu_k u^{(k)}; \xi_{k-1}) - \hat{f}(x^{(k)}; \xi_k)}{\mu_k} u^{(k)}, \tag{2.3.20}$$

which is the "stochastic gradient-free oracle" to the directional derivative of f (with noise) in the direction u. The authors also define the error between the oracle (the forward-difference approximation) and the true directional derivative of f in the direction u as

$$\mathcal{E}(\mu) = \mathcal{E}(\mu; x, u, \xi_1, \xi_2) = ||s_\mu(x; u, \xi_1, \xi_2) - \langle \nabla f(x), u \rangle u||^2. \tag{2.3.21}$$

Note when $u = u_{\mathcal{A}}$ in either (2.3.20) or (2.3.21), we write $s_{\mu_k}^{\mathcal{A}}$ and $\mathcal{E}^{\mathcal{A}}(\mu)$ to emphasize that these values are computed for $u_{\mathcal{A}} \in \mathcal{A}$. If we are working with $\tilde{\mathcal{A}}$ to approximate $\mathcal{A}$, one must replace $\mathcal{A}$ with $\tilde{\mathcal{A}}$ and j with $\tilde{j}$ to obtain the analogous definitions for the approximate $\tilde{\mathcal{A}}$ case.

2.3.2 STARS Convergence with Estimated Hyperparameters

In the following, we follow Chen and Wild (2015) closely. However, we provide more detail and in fact correct some minor details from their unpublished manuscript while generalizing their results to the case in which hyperparameters are estimated. In the latter sections, we build on results in Chen and Wild (2015) and Constantine *et al.* (2014), a paper which contains crucial theoretical results regarding the approximation of active subspaces.

Let the positive, finite values $\hat{L_1}$ and $\hat{\sigma}$ denote estimators to the true (also positive and finite values of L_1 and σ. Recall, in our setting, we will assume $0 < L_1 < \infty$ and $0 < \sigma^2 < \infty$. We won't let $L_1 = 0$ since that would imply f is constant; we won't let $\sigma^2 = 0$, since that would imply zero noise. Then there exist $K_1 > 0$ and $K_2 > 0$ so that

$$L_1^2 = K_1 \hat{L_1}^2 \qquad \sigma^2 = K_2 \hat{\sigma}^2. \tag{2.3.22}$$

Note that if a $K_i < 1$, $i = 1$ or 2, then we have overestimated the corresponding value, L_1 or σ^2 and similarly when $K_i > 1$, the corresponding value has been underestimated. Hence, as a particular $K_i \to 1$, the corresponding estimate to either L_1 or σ^2 approaches the truth. Finally, note that when the true values L_1 and σ^2 are unknown, K_1 and K_2 are also generally unknown.

Below, we recall the approximate smoothing size and step length to replace the STARS hyperparameters in the case that L_1 and σ are unknown and estimated by values $\hat{L_1}$ and $\hat{\sigma}$.

$$\widehat{\mu^*} := \left(\frac{8\hat{\sigma}^2 P}{\hat{L_1}^2 (P+6)^3} \right)^{1/4} \qquad \widehat{h} := (4\hat{L_1}(P+4))^{-1} \tag{2.3.23}$$

Shortly, we will precisely show how the bound $\mathcal{E}(\hat{\mu^*})$ is just a modification of the bound on $\mathcal{E}(\mu^*)$ proven in Chen and Wild (2015). With this point of view, one will see our choice of $\hat{\mu^*}$ is the best that we can do with uncertain estimators to L_1 and σ. We note that both in Chen and Wild (2015) and here in this work, h (and also $\hat{h}$) is a fixed choice, not necessarily optimal nor sub-optimal; more investigation could be done into the step length, but $\hat{h}$ works fine in our numerical experiments.

It is helpful to quote Chen and Wild (2015) here: "Our goal is to find μ^* that minimizes an upper bound on" $\mathbb{E}_{u,\xi_1,\xi_2}(\mathcal{E}(\mu))$, where we have $\mathcal{E}$ defined in (2.3.21) and the expectation is taken over the random vector u, as well as two draws of additive noise from the two function evaluations that occur in s_μ. The noise draws are denoted as $\epsilon(\xi_1)$ and $\epsilon(\xi_2)$.

The major difference between our result and the STARS result is that we will use estimations to L_1 and σ^2 rather than their true values, which we have postulated lack of access to.

STARS Theorem 4.3 (Modified): *We assume random vectors $u^{(k)}$ are drawn according to* (2.3.1)*; $f \in \mathcal{C}^{1,1}(\Lambda)$ and f is convex; and that the i.i.d. noise draws $\epsilon(\xi)$ are additive, zero mean, with bounded variance σ^2 for all ξ. Let u be drawn in the fashion described in* (2.3.1)*. If a smoothing stepsize is chosen as $\mu = \hat{\mu^*}$ in* (2.3.23)*, then for any iterate $x \in \Lambda$ and random vector u, noting $K_1 > 0$ and $K_2 > 0$ as in* (2.3.22)*,*

$$\mathbb{E}_{u,\xi_1,\xi_2}\left(\mathcal{E}\left(\hat{\mu^*}\right)\right) \leq \frac{K_1 + K_2}{\sqrt{2K_1K_2}} \sigma L_1 \sqrt{P(P+6)^3}. \tag{2.3.24}$$

Proof: Let $u \in \Lambda$ be a random vector as in (2.3.1), $x \in \Lambda$ denote a general STARS iterate, and $\epsilon(\xi_1)$ and $\epsilon(\xi_2)$ denote the two i.i.d. draws of the additive noise in $\hat{f}$ which appear in $\mathcal{E}(\mu)$, (2.3.21). Plugging equation (2.3.20) into equation (2.3.21), we obtain

$$\mathcal{E}(\mu) = \left|\left| \frac{f(x+\mu u) + \epsilon(\xi_1) - (f(x) + \epsilon(\xi_2))}{\mu} u - \langle \nabla f(x), u \rangle u \right|\right|^2. \tag{2.3.25}$$

Rearranging,

$$\mathcal{E}(\mu) = \left|\left| \left(\frac{(\epsilon(\xi_1) - \epsilon(\xi_2)) + (f(x+\mu u) - f(x) - \langle \nabla f(x), \mu u \rangle)}{\mu} \right) u \right|\right|^2. \tag{2.3.26}$$

We have

$$\mathcal{E}(\mu) \leq \frac{X^2}{\mu^2}||u||^2, \text{ where } X := (\epsilon(\xi_1) - \epsilon(\xi_2)) + (f(x+\mu u) - f(x) - \langle \nabla f(x), \mu u \rangle)\,. \tag{2.3.27}$$

Expanding the form of X,

$$\begin{aligned} X^2 = {} & \epsilon(\xi_1)^2 - 2\epsilon(\xi_1)\epsilon(\xi_2) + \epsilon(\xi_2)^2 + 2(\epsilon(\xi_1) - \epsilon(\xi_2))\left(f(x+\mu u) - f(x) - \langle \nabla f(x), \mu u \rangle\right) \\ & + \left(f(x+\mu u) - f(x) - \langle \nabla f(x), \mu u \rangle\right)^2. \end{aligned} \tag{2.3.28}$$

We begin by examining the expectation of X^2 with respect to the two stochastic noise draws. Recall that $\mathbb{E}_\xi(\epsilon(\xi)) = 0$ and $\text{Var}(\epsilon(\xi)) = \sigma^2 > 0$ for all draws ξ, hence, we also have $\mathbb{E}_\xi(\epsilon(\xi)^2) = \sigma^2$ for all draws ξ.

$$\mathbb{E}_{\xi_1,\xi_2}(X^2) = 2\sigma^2 + \left(f(x+\mu u) - f(x) - \langle \nabla f(x), \mu u \rangle\right)^2. \tag{2.3.29}$$

Noting that neither u nor μ depends on noise draws ξ, we have

$$\mathbb{E}_{u,\xi_1,\xi_2}(\mathcal{E}(\mu)) = \mathbb{E}_u\left(\frac{\mathbb{E}_{\xi_1,\xi_2}(X^2)||u||^2}{\mu^2}\right) = \frac{1}{\mu^2}\mathbb{E}_u\left(2\sigma^2||u||^2 + \left(f(x+\mu u) - f(x) - \langle \nabla f(x), \mu u \rangle\right)^2 ||u||^2\right). \tag{2.3.30}$$

Now replacing y with $y = x + \mu u$ and squaring both sides, we can re-write (2.3.3) as

$$\left(f(x+\mu u) - f(x) - \langle \nabla f(x), \mu u \rangle\right)^2 \leq \frac{L_1^2}{4}\mu^4||u||^4. \tag{2.3.31}$$

Now, combining (2.3.30) and (2.3.31), we have

$$\mathbb{E}_{u,\xi_1,\xi_2}(\mathcal{E}(\mu)) \leq \frac{1}{\mu^2}\mathbb{E}_u\left(2\sigma^2||u||^2 + \frac{L_1^2}{4}\mu^4||u||^6\right). \tag{2.3.32}$$

Using the bounds on the moments M_p of $||u||^p$ given in (2.3.8), using $p = 2$ and $p = 6$, we have

$$\mathbb{E}_{u,\xi_1,\xi_2}(\mathcal{E}(\mu)) \leq \frac{2\sigma^2}{\mu^2}M_2 + \frac{L_1^2\mu^2}{4}M_6 \leq \frac{2\sigma^2 P}{\mu^2} + \frac{L_1^2\mu^2(P+6)^3}{4}. \tag{2.3.33}$$

We have

$$\mathbb{E}_{u,\xi_1,\xi_2}(\mathcal{E}(\mu)) \leq (2\sigma^2 P)\frac{1}{\mu^2} + \left(\frac{L_1^2(P+6)^3}{4}\right)\mu^2. \tag{2.3.34}$$

The authors in Chen and Wild (2015) observe that the right-hand side of the above inequality is uniformly convex for $\mu > 0$, taking the form $t(\mu) = a\mu^{-2} + b\mu^2$ for positive constants a and b; calculus shows that the minimizer of t for $\mu > 0$ is $\mu^* := (a/b)^{1/4}$ with $t(\mu^*) = 2\sqrt{ab}$.

Using $a = 2\sigma^2 P$ and $b = (L_1^2(P+6)^3)/4$, we recover $\mu^* = \left(\frac{8\sigma^2 P}{L_1^2(P+6)^3}\right)^{1/4}$, the optimal (in the sense of minimizing the upper bound on $\mathcal{E}$) smoothing step length proven in Chen and Wild (2015). Our optimal choice of smoothing, given the information we have available, will require us to swap out L_1 and σ^2 in μ^* with their estimates, $\hat{L_1}$ and $\hat{\sigma}^2$, recovering $\hat{\mu}^*$ (2.3.23). This particular choice $\mu = \hat{\mu^*}$ can be plugged into (2.3.34), which gives us the bound

$$\mathbb{E}_{u,\xi_1,\xi_2}(\mathcal{E}(\hat{\mu^*})) \leq \frac{K_1 + K_2}{\sqrt{2K_1K_2}}\sigma L_1\sqrt{P(P+6)^3}, \tag{2.3.35}$$

our main result. ■

Note that we can recover the exact result of STARS Theorem 4.3 by taking $K_1 = K_2 = 1$, the case in which our hyperparameters are estimated exactly.

We next derive an upper bound on $\mathbb{E}(||s_{\mu_k}||^2)$, where $\mathbb{E}$ will now denote the expectation over every noise draw and random vector used in STARS up to (and including) the k-th iterate; that is, the expectations are now taken with respect to $\xi_0, \ldots, \xi_k$ and $u^{(1)}, \ldots, u^k$ unless stated otherwise. Recall that s_{μ_k}, the stochastic gradient-free oracle, was defined in (2.3.20). We prove this result in a modified Lemma very similar to STARS Lemma 4.4 in Chen and Wild (2015); similarly to the result above, we will only use the estimated values to L_1 and σ.

STARS Lemma 4.4 (Modified): *We assume random vectors $u^{(k)}$ are drawn according to* (2.3.1)*; $f \in \mathcal{C}^{1,1}(\Lambda)$ and f is convex; and that the i.i.d. noise draws $\epsilon(\xi)$ are additive, zero mean, with bounded variance σ^2 for all ξ. If we use $\mu_k = \hat{\mu^*}$ as in* (2.3.23) *for all STARS iterates k, then noting $K_1 > 0$ and $K_2 > 0$ as in* (2.3.22)*, STARS generates steps satisfying*

$$\mathbb{E}(||s_{\mu_k}||^2) \leq 2(P+4)||\nabla f(x^{(k)})||^2 + \frac{3K_1 + K_2}{\sqrt{2K_1K_2}}L_1\sigma\sqrt{P(P+6)^3}. \tag{2.3.36}$$

Proof: First, we set $\mu_k = \hat{\mu^*}$. For a STARS iterate k, let

$$g_0(x^{(k)}) := \langle \nabla f(x^{(k)}), u^{(k)} \rangle u^{(k)}, \tag{2.3.37}$$

which is the exact directional derivative of f in the direction of u at the point $x^{(k)} \in \Lambda$. We can use this notation to re-express (2.3.35) as

$$\mathbb{E}\left(||s_{\mu_k}||^2 - 2\langle s_{\mu_k}, g_0(x^{(k)})\rangle + ||g_0(x^{(k)})||^2\right) \leq \frac{K_1+K_2}{\sqrt{2K_1K_2}}\sigma L_1\sqrt{P(P+6)^3}, \tag{2.3.38}$$

where we have also expanded $\mathcal{E}$, defined in (2.3.21). Recalling that all draws of the noise are zero-mean, the expectation of the oracle s_{μ_k} (defined in (2.3.20)) with respect to the appearing noise draws ξ_{k-1} and ξ_k is given by

$$\mathbb{E}_{\xi_{k-1},\xi_k}(s_{\mu_k}) = \frac{f(x^{(k)}+\mu_k u^{(k)}) - f(x^{(k)})}{\mu_k}u^{(k)} = g_\mu(x^{(k)}), \tag{2.3.39}$$

which is the (noise-free) first-order approximation to the directional derivative of f in the direction of u, defined in (2.3.6). The linearity of $\mathbb{E}$ allows us to rewrite (2.3.38) as

$$\mathbb{E}\left(||s_{\mu_k}||^2\right) \leq \mathbb{E}\left(2\langle s_{\mu_k}, g_0(x^{(k)})\rangle - ||g_0(x^{(k)})||^2\right) + C_1, \tag{2.3.40}$$

where $C_1 := \frac{K_1+K_2}{\sqrt{2K_1K_2}}\sigma L_1\sqrt{P(P+6)^3}$. The only term involving noise draws on the right-hand side of (2.3.40) is s_{μ_k}; thus, passing through the expectation with respect to all noise draws ξ_k, we can use our result in (2.3.39) to write

$$\begin{aligned}\mathbb{E}\left(||s_{\mu_k}||^2\right) &\leq \mathbb{E}\left(2\langle s_{\mu_k}, g_0(x^{(k)})\rangle - ||g_0(x^{(k)})||^2\right) + C_1 \\ &= \mathbb{E}_{u^{(k)}}\left(2\langle g_\mu(x^{(k)}), g_0(x^{(k)})\rangle - ||g_0(x^{(k)})||^2\right) + C_1.\end{aligned} \tag{2.3.41}$$

Adding and subtracting by $||g_\mu(x^{(k)})||^2$ inside of the $\mathbb{E}_{u^{(k)}}$ (and then factoring) in (2.3.41) gives

$$\mathbb{E}\left(||s_{\mu_k}||^2\right) \leq \mathbb{E}_{u^{(k)}}\left(-||g_0(x^{(k)}) - g_\mu(x^{(k)})||^2 + ||g_0(x^{(k)})||^2\right) + C_1. \tag{2.3.42}$$

Using the linearity of $\mathbb{E}_{u^{(k)}}$ and observing that $-||x||^2 \leq 0$ for all $x \in \Lambda$ gives

$$\mathbb{E}\left(||s_{\mu_k}||^2\right) \leq \mathbb{E}_{u^{(k)}}\left(||g_0(x^{(k)})||^2\right) + C_1. \tag{2.3.43}$$

Recalling the result in (2.3.18), we have arrived at

$$\mathbb{E}\left(||s_{\mu_k}||^2\right) \leq 2(P+4)||\nabla f(x^{(k)})||^2 + \frac{\mu_k^2 L_1^2}{2}(P+6)^3 + C_1. \tag{2.3.44}$$

Equivalently, recalling (2.3.22) – which equates L_1^2 to $\hat{L_1}^2$ scaled by a positive constant K_1 – we also have

$$\mathbb{E}\left(||s_{\mu_k}||^2\right) \leq 2(P+4)||\nabla f(x^{(k)})||^2 + \frac{K_1\mu_k^2\hat{L_1}^2}{2}(P+6)^3 + C_1. \tag{2.3.45}$$

Recall we have set $\mu_k = \hat{\mu^*}$ (from (2.3.23)) in (2.3.44) for all iterations k. Plugging in this value, we obtain

$$\mathbb{E}\left(||s_{\mu_k}||^2\right) \leq 2(P+4)||\nabla f(x^{(k)})||^2 + C_2, \tag{2.3.46}$$

where $C_2 := \frac{3K_1+K_2}{\sqrt{2K_1K_2}}L_1\sigma\sqrt{P(P+6)^3} = \frac{3K_1+K_2}{\sqrt{2}}\hat{L_1}\hat{\sigma}\sqrt{P(P+6)^3}$, our main result. ■

Note that in a fashion analogous to our modification of STARS Theorem 4.3, we can recover the exact result of STARS Lemma 4.4 by taking $K_1 = K_2 = 1$.

We can now present the final result which shows that STARS converges with estimates replacing the exact values for L_1 and μ. We need just a bit more notation, borrowed directly from Chen and Wild (2015). Let $x^* \in \Lambda$ denote a minimizer with the associated stochastic-free function evaluation $f^* := f(x^*)$. Also, define $\mathcal{Q}_k := \{\xi_0, \ldots, \xi_k\}$and $\mathcal{U}_k := \{u^{(1)}, \ldots, u^{(k)}\}$, which are two sets containing all random variables that appear in STARS up through iteration k. Let $\phi_0 := f(x^{(0)})$ and $\phi_k := \mathbb{E}_{\mathcal{Q}_{k-1},\mathcal{U}_{k-1}}(f(x^{(k)}))$, $k \geq 1$. Define $M \in \mathbb{N}$ as the total number of STARS iterates performed.

STARS Theorem 4.5 (Modified): *Let Assumptions 3.1, 4.1, and 4.2 hold – here those assumptions mean that random vectors $u^{(k)}$ are drawn according to* (2.3.1)*; $f \in \mathcal{C}^{1,1}(\Lambda)$ and f is convex; and that the i.i.d. noise draws $\epsilon(\xi)$ are additive, zero mean, with bounded variance σ^2 for all ξ. Let $\{x^{(k)}\}_{k\geq 0}$ denote a sequence of STARS iterates formed using a fixed step length $h_k = \hat{h}$ and fixed smoothing $\mu_k = \hat{\mu^*}$ (both given in* (2.3.23)*) for all STARS iterates k.* ***Finally, we require*** *$0 < K_1 < 4$* ***and*** *$K_2 > 0$****, the values defined in*** (2.3.22)*. Then for any total number of STARS iterations M,*

$$\sum_{k=0}^{M}\frac{\phi_k - f^*}{M+1} \leq \frac{4L_1(P+4)||x^{(0)} - x^*||^2}{\sqrt{K_1}(2-\sqrt{K_1})(M+1)} + \frac{4\sigma(P+4)}{\sqrt{2K_2}(2-\sqrt{K_1})}C_5, \tag{2.3.47}$$

where $C_5 := \sqrt{K_1}\cdot 0.036 + \frac{3K_1+K_2}{16}\cdot 1.034$.

Proof: For a STARS iterate $k \geq 0$, let $r_k := ||x^{(k)} - x^*||$, the distance from a given STARS iterate to a true minimizer of f, denoted $x^* \in \Lambda$. Along the lines of Chen and Wild (2015), we will bound $\mathbb{E}(r_{k+1}^2) - r_k$, 'the expected change of x after each iteration in our setting. We note that with this viewpoint, every iterate so far is known; that is, the sequence of vectors $\{x^i\}_{i=0}^k$ are known/fixed up until index k, and thus, the sequence $\{r_i\}_{i=0}^k$ of distances are also known/fixed. In particular, both sequences are non-

stochastic, meaning that they are constant with respect to any expected values we may apply upon them, in the context of a given step.

First, observe that by using definitions, we may write

$$r_{k+1}^2 = ||x^{k+1} - x^*||^2 = ||x^{(k)} - \hat{h}s_{\mu_k} - x^*||^2. \tag{2.3.48}$$

Rearranging and expanding,

$$r_{k+1}^2 = ||(x^{(k)} - x^*) - hs_{\mu_k}||^2 = r_k^2 - 2\hat{h}\langle s_{\mu_k}, x^{(k)} - x^*\rangle + \hat{h}^2||s_{\mu_k}||^2. \tag{2.3.49}$$

Let $\mathbb{E}$ continue to denote the expectation over $\mathcal{Q}_k$ and $\mathcal{U}_k$, all of the random vectors and noise draws defining our first k iterates. Recall that one of our current assumptions is that all of the $x^{(k)}$'s (and thus also all of the r_k's) are given/fixed, as well as $\hat{h}$ and x^*. Hence, $\mathbb{E}(r_k^2) = r_k^2$, $\mathbb{E}(x^{(k)}) = x^{(k)}$, $\mathbb{E}(\hat{h}) = \hat{h}$, and $\mathbb{E}(x^*) = x^*$ as we already have these constant objects in hand. However, the next iterate, x^{k+1}, *will* depend on the stochastic direction $u^{(k)}$, as well as the stochastic noise values ξ_{k-1} and ξ_k – and since these stochastic objects literally define r_{k+1} and s_{μ_k}, the expectations will not drop from these terms.

Applying $\mathbb{E}$ to both sides of (2.3.49),

$$\mathbb{E}(r_{k+1}^2) = r_k^2 - 2\hat{h}\langle \mathbb{E}\left(s_{\mu_k}\right), x^{(k)} - x^*\rangle + \hat{h}^2\mathbb{E}\left(||s_{\mu_k}||^2\right). \tag{2.3.50}$$

We begin by noting the appearance of $\mathbb{E}\left(s_{\mu_k}\right)$, which we can characterize using a pair of previous results. First, we found in (2.3.39) that $\mathbb{E}_{\xi_{k-1},\xi_k}(s_{\mu_k}) = g_\mu(x^{(k)})$. Next, we recall (2.3.16), $\mathbb{E}_u\left(g_\mu(x)\right) = \nabla f_\mu(x) \quad \forall x \in \Lambda$. Putting these results together, we have $\mathbb{E}\left(s_{\mu_k}\right) = \nabla f_\mu(x^{(k)})$. We can invoke the main result in STARS Lemma 4.4 (Modified) (summarized by (2.3.46)) to bound $\mathbb{E}\left(||s_{\mu_k}||^2\right)$ and our new characterization of $\mathbb{E}\left(s_{\mu_k}\right) = \nabla f_\mu(x^{(k)})$ to write

$$\mathbb{E}(r_{k+1}^2) \leq r_k^2 - 2\hat{h}\langle \nabla f_\mu(x^{(k)}), x^{(k)} - x^*\rangle + \hat{h}^2\left(2(P+4)||\nabla f(x^{(k)})||^2 + C_2\right). \tag{2.3.51}$$

Now, to reach our next key result, we shall need to verify that f_μ is convex. Let $x, y \in \Lambda$ and $\mu > 0$. By definition, $f_\mu(y) = \mathbb{E}_u(f(y + \mu u))$. Recalling that we have assumed the convexity of f we invoke (2.3.5), writing

$$f_\mu(y) = \mathbb{E}_u(f(y + \mu u)) \geq \mathbb{E}_u\left(f(x + \mu u) + \langle \nabla f(x + \mu u), y - x\rangle\right). \tag{2.3.52}$$

Now using the properties of $\mathbb{E}_u$ and defintions,

$$f_\mu(y) \geq \mathbb{E}_u\left(f(x+\mu u)\right) + \langle \nabla\left(\mathbb{E}_u\left(f(x+\mu u)\right)\right), y-x\rangle = f_\mu(x) + \langle \nabla f_\mu(x), y-x\rangle, \tag{2.3.53}$$

which holds for any $x, y \in \Lambda$ and $\mu > 0$, proving that f_μ is convex (so long as f is also convex). We shall require a restatement of (2.3.53) for our purposes; we also have for any $x, y \in \Lambda$ and $\mu > 0$:

$$\langle \nabla f_\mu(x), x-y\rangle \geq f_\mu(x) - f_\mu(y). \tag{2.3.54}$$

Hence, plugging $x = x^{(k)}$ and $y = x^*$ into (2.3.54) and multiplying both sides of the inequality by $-2\hat{h}$, we obtain

$$-2\hat{h}\langle \nabla f_\mu(x^{(k)}), x^{(k)} - x^*\rangle \leq -2\hat{h}\left(f_\mu(x^{(k)}) - f_\mu(x^*)\right). \tag{2.3.55}$$

Now (2.3.12) implies $-2\hat{h}f_\mu(x^{(k)}) \leq -2\hat{h}f(x^{(k)})$. Hence,

$$-2\hat{h}\langle \nabla f_\mu(x^{(k)}), x^{(k)} - x^*\rangle \leq -2\hat{h}\left(f(x^{(k)}) - f_\mu(x^*)\right). \tag{2.3.56}$$

Recalling (2.3.4), we can write $||\nabla f(x^{(k)})||^2 \leq 2L_1(f(x^{(k)}) - f(x^*))$. Using this result along with (2.3.56), we can update our bound in (2.3.51), writing

$$\mathbb{E}(r_{k+1}^2) \leq r_k^2 - 2\hat{h}\left(f(x^{(k)}) - f_\mu(x^*)\right) + \hat{h}^2\left(4L_1(P+4)(f(x^{(k)}) - f(x^*)) + C_2\right). \tag{2.3.57}$$

Now we add and subtract $-2\hat{h}f(x^*)$ on the RHS of (2.3.57), obtaining

$$\mathbb{E}(r_{k+1}^2) \leq r_k^2 - 2\hat{h}\left(f(x^*) - f_\mu(x^*)\right) - 2\hat{h}(f(x^{(k)}) - f(x^*)) + \hat{h}^2\left(4L_1(P+4)(f(x^{(k)}) - f(x^*)) + C_2\right). \tag{2.3.58}$$

Now (2.3.13) implies $-\frac{\mu^2}{2}L_1P \leq f(x) - f_\mu(x) \implies -2\hat{h}(f(x) - f_\mu(x)) \leq \hat{h}\mu^2 L_1 P$ for all x, in particular for $x = x^*$; hence,

$$\mathbb{E}(r_{k+1}^2) \leq r_k^2 + \hat{h}\mu^2 L_1 P - 2\hat{h}(f(x^{(k)}) - f(x^*)) + \hat{h}^2\left(4L_1(P+4)(f(x^{(k)}) - f(x^*)) + C_2\right). \tag{2.3.59}$$

Manipulating and rearranging a little,

$$\mathbb{E}(r_{k+1}^2) \leq r_k^2 - 2\hat{h}(f(x^{(k)}) - f(x^*))(1 - 2\hat{h}L_1(P+4)) + C_3, \tag{2.3.60}$$

where $C_3 := \hat{h}\mu^2 L_1 P + \hat{h}^2 C_2$. Now recall that we have fixed $\hat{h} = (4\hat{L_1}(P+4))^{-1}$; plugging in this particular value, we obtain

$$\mathbb{E}(r_{k+1}^2) \leq r_k^2 - \frac{\sqrt{K_1}(2-\sqrt{K_1})}{4L_1(P+4)}(f(x^{(k)}) - f(x^*)) + C_3, \tag{2.3.61}$$

We rewrite C_3 (with our $\hat{h}$ and $\hat{\mu}^*$ plugged in) with a function $g(P)$, $P = \dim \Lambda$ given by:

$$C_3 = \frac{\sqrt{K_1}}{\sqrt{K_2}} \cdot \frac{\sigma}{\sqrt{2}L_1} g(P), \quad g(P) := \left(\frac{\sqrt{K_1}}{P+4} \cdot \left(\frac{P}{P+6} \right)^{3/2} + \frac{3K_1 + K_2}{16} \cdot \frac{\sqrt{P(P+6)^3}}{(P+4)^2} \right). \tag{2.3.62}$$

We may recover the function called g_1 from Chen and Wild (2015) by setting $K_1 = K_2 = 1$, so that our estimates to L_1 and σ^2 are exact. We shall bound our more general g by bounding each of its terms. Note that the asymptotic analysis of each term gives us hope to find such a bound, as the first term tends to zero as $P \to \infty$ and the second term tends to the constant $(3K_1 + K_2)/16$. Using calculus and numerics along the lines of Chen and Wild (2015), we find

$$g(P) \leq \sqrt{K_1} \cdot 0.036 + \frac{3K_1 + K_2}{16} \cdot 1.034. \tag{2.3.63}$$

With $K_1 = K_2 = 1$, we have $g(P) \approx 0.2895 < 3/10$ again matching Chen and Wild (2015) (using the bound of 3/10). We may now define a constant, C_4, which bounds C_3 over all dimensions P in terms of L_1, σ, K_1, and K_2:

$$C_3 \leq \frac{\sqrt{K_1}}{\sqrt{K_2}} \cdot \frac{\sigma}{\sqrt{2}L_1} \left(\sqrt{K_1} \cdot 0.036 + \frac{3K_1 + K_2}{16} \cdot 1.034 \right) =: C_4. \tag{2.3.64}$$

Thus, we can update the bound in (2.3.61) to write

$$\mathbb{E}(r_{k+1}^2) \leq r_k^2 - \frac{\sqrt{K_1}(2-\sqrt{K_1})}{4L_1(P+4)}(f(x^{(k)}) - f(x^*)) + C_4. \tag{2.3.65}$$

Applying the expectation over $\mathcal{U}_k$ and $\mathcal{P}_k$,

$$\mathbb{E}_{\mathcal{U}_k, \mathcal{P}_k}(r_{k+1}^2) \leq \mathbb{E}_{\mathcal{U}_{k-1}, \mathcal{P}_{k-1}}(r_k^2) - \frac{\sqrt{K_1}(2-\sqrt{K_1})}{4L_1(P+4)}(\phi_k - f^*) + C_4. \tag{2.3.66}$$

Rearranging, we have

$$\phi_k - f^* \leq \frac{4L_1(P+4)}{\sqrt{K_1}(2-\sqrt{K_1})} \left(\mathbb{E}_{\mathcal{U}_{k-1},\mathcal{P}_{k-1}}(r_k^2) - \mathbb{E}_{\mathcal{U}_k,\mathcal{P}_k}(r_{k+1}^2) + C_4\right). \tag{2.3.67}$$

Summing over $k = 0, \ldots, M$ and dividing by $M+1$, we obtain

$$\sum_{k=0}^{M} \frac{\phi_k - f^*}{M+1} \leq \frac{4L_1(P+4)}{\sqrt{K_1}(2-\sqrt{K_1})(M+1)} \left(r_0 - \mathbb{E}_{\mathcal{U}_k,\mathcal{P}_k}(r_{k+1}^2)\right) + \frac{4L_1(P+4)}{\sqrt{K_1}(2-\sqrt{K_1})} C_4. \tag{2.3.68}$$

Dropping the strictly negative term on the RHS (involving $\mathbb{E}_{\mathcal{U}_k,\mathcal{P}_k}(r_{k+1}^2)$) and plugging in the definition of r_0 (which is a constant, and not stochastic) and noting that we require $0 < K_1 < 4$ and $K_2 > 0$, we have found

$$\sum_{k=0}^{M} \frac{\phi_k - f^*}{M+1} \leq \frac{4L_1(P+4)||x^{(0)} - x^*||^2}{\sqrt{K_1}(2-\sqrt{K_1})(M+1)} + \frac{4\sigma(P+4)}{\sqrt{2K_2}(2-\sqrt{K_1})} C_5, \tag{2.3.69}$$

where $C_5 := \sqrt{K_1} \cdot 0.036 + \frac{3K_1+K_2}{16} \cdot 1.034$, our main result. ■

Again, we recover the exact result of STARS Theorem 4.5 by taking $K_1 = K_2 = 1$.

Remark: We now mimic the analysis in Chen and Wild (2015) to explain the implications of our modified Theorem 4.5. First, let $||x^{(0)} - x^*||^2 \leq R^2$. Define $x^\dagger := \text{argmin}_x\{f(x) : x \in \{x^{(0)}, \ldots, x^M\}\}$ and $\phi_\dagger := \mathbb{E}_{\mathcal{U}_{k-1},\mathcal{P}_{k-1}}(f(x^\dagger))$. Then the value $\phi_\dagger - f^*$ must be less than or equal to the average improvement for any given run of STARS; that is, (2.3.69), along with our new definitions, implies that

$$\phi_\dagger - f^* \leq \sum_{k=0}^{M} \frac{\phi_k - f^*}{M+1} \leq \frac{4L_1(P+4)}{\sqrt{K_1}(2-\sqrt{K_1})(M+1)} R^2 + \frac{4\sigma(P+4)}{\sqrt{2K_2}(2-\sqrt{K_1})} C_5. \tag{2.3.70}$$

Along the lines of Chen and Wild (2015), let us now assume that we wish to achieve a final accuracy of $\epsilon_{\text{tol}} > 0$. Then we will need $\phi_\dagger - f^* \leq \epsilon_{\text{tol}}$. If we take

$$\frac{4\sigma(P+4)}{\sqrt{2K_2}(2-\sqrt{K_1})} C_5 \leq \frac{\epsilon_{\text{tol}}}{2}, \tag{2.3.71}$$

then we must require that the noise not exceed the following threshold:

$$\sigma \leq \frac{\sqrt{2K_2}(2-\sqrt{K_1})\epsilon_{\text{tol}}}{8(P+4)C_4}, \tag{2.3.72}$$

If we satisfy (2.3.72), then we can achieve ϵ_{tol} accuracy as long as

$$\frac{4L_1(P+4)R^2}{\sqrt{K_1}(2-\sqrt{K_1})(M+1)} \leq \frac{\epsilon_{\text{tol}}}{2} \iff M \geq \frac{8L_1(P+4)R^2}{\sqrt{K_1}(2-\sqrt{K_1})\epsilon_{\text{tol}}} - 1. \tag{2.3.73}$$

Hence, we achieve ϵ_{tol} accuracy as long as the noise is small enough, and M is large enough, with details of those bounds given by (2.3.72) and (2.3.73). Also, we achieve ϵ_{tol} accuracy in

$$M \sim \mathcal{O}\left(\frac{L_1 P R^2}{\sqrt{K_1}(2-\sqrt{K_1})\epsilon_{\text{tol}}}\right). \tag{2.3.74}$$

This analysis also shows that given a particular variance in the noise, σ, the achievable accuracy can be no better (i.e., less) than the value given below:

$$\epsilon_{\text{tol}} \geq \frac{8(P+4)C_5}{\sqrt{2K_2}(2-\sqrt{K_1})}\sigma. \tag{2.3.75}$$

As usual, we recover the results in Chen and Wild (2015) by setting $K_1 = K_2 = 1$.

2.3.3 ASTARS Convergence

We now investigate the convergence of ASTARS. We will build upon the theoretical results of the last section, meaning Chen and Wild (2015) will be heavily invoked again in this section. Given the exact j-dimensional AS $\mathcal{A}$ of f, we shall also need results generally regarding the distance between the minimum of f and the minimum that ASTARS obtains. We shall also need to discuss the corresponding minimizers. Recall that we denote the minimizer of f with x^* and we have the stochastic-free minimum of f, $f^* = f(x^*)$, as before. Since ASTARS steps in $\mathcal{A}$ only, given an initial iterate $x^{(0)}$, the minimizer ASTARS is able to attain will be of the form $x^*_{\mathcal{A}} := P_{\mathcal{A}}(x^*) + P_{\mathcal{I}}(x^{(0)})$. We analogously define the stochastic-free $f^*_{\mathcal{A}} := f(x^*_{\mathcal{A}})$. Since the initial iterate will not be changed in $\mathcal{I}$ during ASTARS, the components in $x^*_{\mathcal{A}}$ are fixed in $\mathcal{I}$ at the given initial iterate, given by the inactive coordinates of $P_{\mathcal{I}}(x^{(0)})$. However, we do step towards the true x^* in its coordinates corresponding to $\mathcal{A}$, which is why we also obtain $P_{\mathcal{A}}(x^*)$ in the definition of $x^*_{\mathcal{A}}$. Notice that with our definitions, $x^* - x^*_{\mathcal{A}} = P_{\mathcal{I}}(x^{(0)} - x^*)$. Again – the difference between x^* and $x^*_{\mathcal{A}}$ will be in the inactive subspace $\mathcal{I}$, given exactly by the projection of $x^{(0)}$ into $\mathcal{I}$, since ASTARS iterations do not perturb $\mathcal{I}$-coordinates.

ASTARS estimates directional derivatives of f strictly for directions in $\mathcal{A}$ – iterates are not perturbed in $\mathcal{I}$. Consequently, the ASTARS gradient oracle can only provide gradient information in the j active directions of f. Hence, the gradients we approximate in ASTARS are denoted $\nabla_{\mathcal{A}} f(x) \in \mathcal{A}$, which is the gradient of f in $\mathcal{A}$. Gradients in $\mathcal{I}$ will also be needed for our proofs, and they are defined similarly with $\nabla_{\mathcal{I}} f(x) \in \mathcal{I}$. Note the *subspace* gradients $\nabla_{\mathcal{A}} f(x)$ and $\nabla_{\mathcal{I}} f(x)$ are still computed in Λ, but each will

fall into their respective subspaces upon computation. In particular, $\nabla_{\mathcal{A}} f(x)_i = 0$ for $i = j+1, \ldots, P$ and $\nabla_{\mathcal{I}} f(x)_i = 0$ for $i = 1, \ldots, j$.

We first present a lemma which will be used in both the ASTARS and FAASTARS convergence analyses. Recall that the vectors $r^{(k)}$ and $\tilde{r}^{(k)}$ have components which are drawn from a $N(0,1)$ distribution. These vectors are used to form random coefficients in a linear combination in $\mathcal{A}$ (or $\tilde{\mathcal{A}}$) to perform ASTARS steps. Here, we write $r^{(k)} \sim N(0, I_j)$, a multivariate normal distribution, where 0 denotes the zero vector in Λ^j and I_j is the $j \times j$ identity matrix, so that the covariance is 1 for every element in $r^{(k)}$ but all elements are independent, with zero covariance between elements. Analogously, we have $\tilde{r}_p^{(k)} \sim N(0, I_{\tilde{j}})$. In the first lemma, we show that the random directions for ASTARS steps $u^{(k)}$ and $\tilde{u}^{(k)}$ are also distributed normally with zero mean and unit covariance.

ASTARS/FAASTARS Lemma 1: *Let $\tilde{\mathcal{A}}$ denote a $\tilde{j}$-dimensional AS of $\hat{f}$ and let $\mathcal{A}$ denote the true j-dimensional AS of $\hat{f}$. Recall $V_{\mathcal{A}} := V_{1:P,1:j}$, where V comes for the eigendecomposition (ED) of the exact sensitivity matrix W; as well, recall that $\tilde{V}_{\tilde{\mathcal{A}}} := \tilde{V}_{1:P,1:\tilde{j}}$, where $\tilde{V}$ comes from the ED of the sensitivity matrix $\tilde{W}$, approximated from samples of $\hat{f}$. Let $r^{(k)}$ denote a random vector such that $r^{(k)} \sim N(0, I_j)$; likewise, let $\tilde{r}^{(k)}$ denote a random vector such that $\tilde{r}^{(k)} \sim N(0, I_{\tilde{j}})$. Let $u^{(k)} := V_{\mathcal{A}} r^{(k)}$ and $\tilde{u}^{(k)} := \tilde{V}_{\tilde{\mathcal{A}}} \tilde{r}^{(k)}$. Then both $u^{(k)}$ and $\tilde{u}^{(k)}$ are normal random vectors; i.e., $u^{(k)} \sim N(0, I_j)$ and $\tilde{u}_p^{(k)} \sim N(0, I_{\tilde{j}})$. Also, $u^{(k)} \in \mathcal{A}$ and $\tilde{u}^{(k)} \in \tilde{\mathcal{A}}$ for all k.*

Proof: We begin by considering the case in which we have the exact AS of $\hat{f}$, $\mathcal{A}$. We recall that since W is a real $P \times P$ symmetric matrix, its ED is $W = VQV^\top$ where V contains the P eigenvectors of W which are orthonormal in this case, due to the symmetry of W, meaning V is a unitary matrix. (Note Q contains the eigenvalues of W along its diagonal in descending order.) Recall that $V_{\mathcal{A}} := V_{1:P,1:j}$; hence, $V_{\mathcal{A}}$ is also a unitary matrix.

By definition, $u^{(k)} = V_{\mathcal{A}} r^{(k)}$. Since every component of $r^{(k)}$ is a $N(0,1)$ random variable, $u^{(k)}$ is distributed as $u^{(k)} \sim N(0, (V_{\mathcal{A}})^\top (V_{\mathcal{A}}))$. Now since $V_{\mathcal{A}}$ is unitary, we know $(V_{\mathcal{A}})^\top (V_{\mathcal{A}}) = I_j$. Therefore $u^{(k)} \sim N(0, I_j)$, our desired result for the case of an exact AS.

In the case that we are dealing with an approximated $\tilde{j}$-dimensional AS $\tilde{\mathcal{A}}$, it is still the case that $\tilde{V}_{\tilde{\mathcal{A}}}$ is unitary by construction, and so we can follow the proof above analogously, only replacing j with $\tilde{j}$, and state $\tilde{u}_p^{(k)} \sim N(0, I_{\tilde{j}})$ as well.

To verify $u^{(k)} \in \mathcal{A}$ and $\tilde{u}^{(k)} \in \tilde{\mathcal{A}}$, recall $u^{(k)} = V_{\mathcal{A}} r^{(k)}$. Then $u^{(k)}$ is a linear combination of the columns of $V_{\mathcal{A}}$ with coefficients given by $r^{(k)}$. The columns of V_A are the eigenvectors v^i, $i = 1, \ldots, j$ of the associated sensitivity matrix W meaning $u^{(k)}$ a linear combination of the first j eigenvectors of W. Since the span of those j eigenvectors equals $\mathcal{A}$ by definition, $u^{(k)} \in \mathcal{A}$. (Subspaces are closed under linear combinations of their elements.) The argument is analogous for $\tilde{u}^{(k)} \in \tilde{\mathcal{A}}$. ■

We now formulate a series of ASTARS results, where we assume $\mathcal{A}$ is correct and not estimated. We now show that using a fixed step size $h_{\mathcal{A}}$ in (2.2.2), the active smoothing parameter $\mu^*_{\mathcal{A}}$ in (2.2.2) is optimal, in the sense that the error in the gradient oracle used in 2 is minimized. This result is a direct corollary of ASTARS Proposition 1 above and STARS Theorem 4.3 (Modified) in the previous section, but with $K_1 = K_2 = 1$, so that we have L_1 and σ^2 exactly to form the active hyperparameters.

ASTARS Corollary 2: *Let the vectors $u^{(k)}_{\mathcal{A}}$ denote those drawn using Algorithm 2; let $f \in \mathcal{C}^{1,1}(\Lambda)$ and assume f is convex; and assume that the i.i.d. noise draws $\epsilon(\xi)$ are additive, zero mean, with bounded variance σ^2 for all ξ. By fixing the step size $h_{\mathcal{A}}$ in* (2.2.2)*, the active smoothing parameter $\mu^*_{\mathcal{A}}$ in* (2.2.2) *minimizes the error between the gradient oracle in Algorithm 2 and the true directional derivative of f in the direction $u^{(k)}_{\mathcal{A}}$ in the j-dimensional AS $\mathcal{A}$. That is, $\mathcal{E}^{\mathcal{A}}(\mu)$ in* (2.3.21) *(with $u = u^{(k)}_{\mathcal{A}}$) is minimized by the choice $\mu = \mu^*_{\mathcal{A}}$. In particular, we have the bound*

$$\mathbb{E}_{u^{(k)}_{\mathcal{A}},\xi_1,\xi_2}\left(\mathcal{E}^{\mathcal{A}}\left(\mu^*_{\mathcal{A}}\right)\right) \leq \sqrt{2}\sigma L_1 \sqrt{j(j+6)^3}. \tag{2.3.76}$$

Proof: Replacing $\mathcal{E}$ with $\mathcal{E}^{\mathcal{A}}$ and taking the expectation over the noise and $u = u_{\mathcal{A}}$, the proof is identical to the proof of STARS Theorem 4.3 (Modified), until we formulate (as in (2.3.32))

$$\mathbb{E}_{u_{\mathcal{A}},\xi_1,\xi_2}(\mathcal{E}^{\mathcal{A}}(\mu)) \leq \frac{1}{\mu^2}\mathbb{E}_u\left(2\sigma^2||u||^2 + \frac{L_1^2}{4}\mu^4||u||^6\right) \tag{2.3.77}$$

and proceed to bound the right hand side. Applying ASTARS Proposition 1, we have $(u^{(k)}_{\mathcal{A}})_p \sim N(0,1)$ for $p = 1,\ldots,P$. Taking $u = u^{(k)}_{\mathcal{A}}$ and noting $u^{(k)}_{\mathcal{A}} \in \mathcal{A}$ (and $\dim \mathcal{A} = j$), we apply the bounds on the moments M_p of $||u||^p$ given in (2.3.11). Using $p = 2$ and $p = 6$ we have

$$\mathbb{E}_{u^{(k)}_{\mathcal{A}},\xi_1,\xi_2}(\mathcal{E}^{\mathcal{A}}(\mu)) \leq (2\sigma^2 j)\frac{1}{\mu^2} + \left(\frac{L_1^2(j+6)^3}{4}\right)\mu^2. \tag{2.3.78}$$

We again observe that the right-hand side of the above inequality is uniformly convex for $\mu > 0$ with minimizer $\mu^*_{\mathcal{A}} := \left(\frac{8\sigma^2 j}{L_1^2(j+6)^3}\right)^{1/4}$. This particular choice $\mu = \hat{\mu^*_{\mathcal{A}}}$ can be plugged into (2.3.78), and we obtain the bound in (2.3.76), our main result. ■

Next, define $\mathcal{P}_k := \{\xi_k\}_{k=1}^M$ and $\mathcal{U}_k^{\mathcal{A}} := \{u^{(k)}_{\mathcal{A}}\}_{k=1}^M$, which are two sets containing all random variables that appear in ASTARS, iterations $k = 1,\ldots,M$. Let $\phi_0 := f(x^{(0)})$ and $\phi_k^{\mathcal{A}} := \mathbb{E}_{\mathcal{Q}_{k-1},\mathcal{U}^{\mathcal{A}}_{k-1}}(f(x^{(k)}))$, $k \geq 1$, where the $x^{(k)}$'s are now ASTARS iterates. $\mathbb{E}$ will now denote the expectation over every noise draw and random vector used in STARS up to (and including) the k-th iterate; that is, the expectations are now taken with respect to $\xi_0,\ldots,\xi_k$ and $u^{(1)}_{\mathcal{A}},\ldots,u^{(k)}_{\mathcal{A}}$ unless stated otherwise.

Now, given that the active smoothing parameter $\hat{\mu^*_{\mathcal{A}}}$ is optimal, in the sense of minimizing $\mathcal{E}^{\mathcal{A}}$, we present the following result, showing the convergence of ASTARS. The following result is a direct corollary of STARS Lemma 4.4 (Modified), STARS Theorem 4.5 (Modified), ASTARS/FAASTARS Lemma 1, and FAASTARS Corollary 2.

ASTARS Corollary 3: *Let random vectors $u_{\mathcal{A}}^{(k)}$ be drawn according to 2; $f \in \mathcal{C}^{1,1}(\Lambda)$ and f is convex; and that the i.i.d. noise draws $\epsilon(\xi)$ are additive, zero mean, with bounded variance σ^2 for all ξ. Let $\{x^{(k)}\}_{k\geq 0}$ denote a sequence of ASTARS iterates formed using a fixed active step length $h_{\mathcal{A}}$ and fixed active smoothing $\mu = \mu^*_{\mathcal{A}}$ (both given in (2.2.2)) for all ASTARS iterates k. Then for any total number of ASTARS iterations M,*

$$\sum_{k=0}^{M} \frac{\phi_k^{\mathcal{A}} - f^*_{\mathcal{A}}}{M+1} \leq \frac{4L_1(j+4)||P_{\mathcal{A}}(x^{(0)} - x^*)||^2}{(M+1)} + \frac{3\sqrt{2}\sigma(j+4)}{5}. \tag{2.3.79}$$

Proof: The proof is almost identical to the proofs of STARS Lemma 4.4 (Modified) and STARS Theorem 4.5 (Modified), but with j's replacing the roles of P's (since we take steps with j-dimensional $u_{\mathcal{A}}^{(k)}$ vectors and not $u^{(k)} \in \Lambda$), $\mathcal{E}^{\mathcal{A}}$ replacing $\mathcal{E}$, $\mathcal{U}^{\mathcal{A}}$ replacing $\mathcal{U}$, $\mu^*_{\mathcal{A}}$ replacing $\hat{\mu}^*$, $\phi_k^{\mathcal{A}}$ replacing ϕ_k, $x^*_{\mathcal{A}}$ replacing x^* (since we are converging to the minimum of $f_{\mathcal{A}}$, and $K_1 = K_2 = 1$, since we are assuming exact active hyperparameters, formed with the true values for L_1 and σ^2. Note $||\cdot|| = ||\cdot||_\Lambda$, a norm on Λ throughout. We outline the required changes one must make to the proofs of STARS Lemma 4.4 (Modified) and STARS Theorem 4.5 (Modified) to obtain our desired result.

We begin by obtaining a bound analogous to that of STARS Lemma 4.4 (Modified), but note that ASTARS steps are taken with random vectors $u_{\mathcal{A}}^{(k)} \in \mathcal{A}$, and $\dim \mathcal{A} = j$. First, we replace $u^{(k)}$ in (2.3.37) with $u_{\mathcal{A}}^{(k)}$, so $g_0(x^{(k)}) := \langle \nabla f(x^{(k)}), u_{\mathcal{A}}^{(k)} \rangle u_{\mathcal{A}}^{(k)}$. Similarly, note that we set $u^{(k)} = u_{\mathcal{A}}^{(k)}$ in (2.3.20) Then, using ASTARS Corollary 2 – instead of the STARS Theorem 4.3 (Modified) – we have

$$\mathbb{E}\left(||s^{\mathcal{A}}_{\mu^*_{\mathcal{A}}}||^2 - 2\langle s^{\mathcal{A}}_{\mu^*_{\mathcal{A}}}, g_0(x^{(k)})\rangle + ||g_0(x^{(k)})||^2\right) \leq \sqrt{2}\sigma L_1 \sqrt{j(j+6)^3}. \tag{2.3.80}$$

Using (2.3.19), we have

$$\mathbb{E}\left(||s^{\mathcal{A}}_{\mu^*_{\mathcal{A}}}||^2\right) \leq 2(j+4)||\nabla f(x^{(k)})||^2 + \frac{(\mu^*_{\mathcal{A}})^2 L_1^2}{2}(j+6)^3 + C_1, \tag{2.3.81}$$

where we recall we have $C_1 = \sqrt{2}\sigma L_1 \sqrt{j(j+6)^3}$ here. Plugging in the value of $\mu^*_{\mathcal{A}}$, we obtain the bound

$$\mathbb{E}\left(||s^{\mathcal{A}}_{\mu^*_{\mathcal{A}}}||^2\right) \leq 2(j+4)||\nabla f(x^{(k)})||^2 + C_2, \tag{2.3.82}$$

where $C_2 = 2\sqrt{2}L_1\sigma\sqrt{j(j+6)^3}$. The bound in (2.3.81) is the analogous result to STARS Lemma 4.4 (Modified) in the case of ASTARS performed in the known and exact $\mathcal{A}$ and with $K_1 = K_2 = 1$ (exact hyperparameters).

We now proceed to proving the analogous result to STARS Theorem 4.5 (Modified) in our case. We redefine $r_k := ||x^{(k)} - x^*_{\mathcal{A}}||$ for ASTARS iterates $x^{(k)}$. The first three equations appearing in the proof of STARS Theorem 4.5 (Modified), (2.3.48) through (2.3.50), are nearly identical for this proof – one must replace x^* with $x^*_{\mathcal{A}}$, replace $\hat{h}$ with $h_{\mathcal{A}}$, and note that here we have $s_{\mu^{(k)}} = s^{\mathcal{A}}_{\mu^*_{\mathcal{A}}}$. Then, invoking (2.3.82), we rewrite (2.3.51) in our case as

$$\mathbb{E}(r^2_{k+1}) \leq r^2_k - 2h_{\mathcal{A}}\langle \nabla f^{\mathcal{A}}_{\mu}(x^{(k)}), x^{(k)} - x^*_{\mathcal{A}}\rangle + h^2_{\mathcal{A}}\left(2(j+4)||\nabla f(x^{(k)})||^2 + C_2\right), \tag{2.3.83}$$

where we recall $C_2 = 2\sqrt{2}L_1\sigma\sqrt{j(j+6)^3}$ here. Now, again, equations (2.3.52) through (2.3.56) are identical for this proof as long as $x^*_{\mathcal{A}}$ replaces x^* and $h_{\mathcal{A}}$ replaces $\hat{h}$ throughout, and similarly for equations (2.3.57) through (2.3.59) with the additional needed replacement of j for P and using C_2 as we have defined it here. Taking $C_3 = h_{\mathcal{A}}\mu^*_{\mathcal{A}}L_1 j + h^2_{\mathcal{A}}C_2$, (2.3.60) holds (with the usual replacments) and plugging $h_{\mathcal{A}}$ into the modified (2.3.60) gives the following modification to (2.3.61):

$$\mathbb{E}(r^2_{k+1}) \leq r^2_k - \frac{1}{4L_1(j+4)}(f(x^{(k)}) - f(x^*_{\mathcal{A}})) + C_3. \tag{2.3.84}$$

Now plugging $h_{\mathcal{A}}$ into C_3, we obtain $C_3 \leq \frac{3\sigma}{10\sqrt{2}L_1} =: C_4$, so

$$\mathbb{E}(r^2_{k+1}) \leq r^2_k - \frac{1}{4L_1(j+4)}(f(x^{(k)}) - f(x^*_{\mathcal{A}})) + C_4. \tag{2.3.85}$$

We may now apply the expectation over $\mathcal{P}_k$ and $\mathcal{U}^{\mathcal{A}}_k$, rearrange, and sum over $k = 0, \ldots, M$ as before. We obtain a modification to (2.3.68) with

$$\sum_{k=0}^{M} \frac{\phi^{\mathcal{A}}_k - f^*_{\mathcal{A}}}{M+1} \leq \frac{4L_1(j+4)}{(M+1)}\left(r_0 - \mathbb{E}_{\mathcal{U}^{\mathcal{A}}_k, \mathcal{P}_k}(r^2_{k+1})\right) + 4L_1(j+4)C_4. \tag{2.3.86}$$

We drop the strictly negative term (again involving $\mathbb{E}_{\mathcal{P}_k, \mathcal{U}^{\mathcal{A}}_k}$) and plug in the definition of $r_0 = ||x^{(0)} - x^*_{\mathcal{A}}||^2$. Then, recalling $x^*_{\mathcal{A}} = P_{\mathcal{A}}(x^*) + P_{\mathcal{I}}(x^{(0)})$, writing $x^{(0)} = P_{\mathcal{A}}(x^{(0)}) + P_{\mathcal{I}}(x^{(0)})$, and noting that $P_{\mathcal{A}}$ is linear, we have $r_0 = ||P_{\mathcal{A}}(x^{(0)} - x^*)||^2$. We obtain (2.3.79). ■

We have shown the convergence of ASTARS to the minimum $f^*_{\mathcal{A}}$ of $f_{\mathcal{A}}$ with correct hyperparameters and the correct and known $\mathcal{A}$. Ultimately, to obtain complexity results for ASTARS, we desire a

statement about the convergence of ASTARS to f^*, the minimum of f. We pay a price for stepping only in active variables in ASTARS, which is that inactive variables are not minimized or even perturbed at all. Because ASTARS converges to $f^*_{\mathcal{A}}$, we will not minimize f in its inactive variables $\mathcal{I}$, and $|f^* - f^*_{\mathcal{A}}|$ may be nonzero. By modifying results in Constantine *et al.* (2014), we show that this difference will usually be negligible, as it is bounded by the square root of the sum of the eigenvalues associated with $\mathcal{I}$ (which are usually small), scaled by a constant related to the distance from x^* to $x^*_{\mathcal{A}}$ (which is also small in many cases).

ASTARS Corollary 4: *Let $x^{(0)}$ denote any initial iterate for ASTARS. Let x^* denote a true minimizer of f with the corresponding stochastic-free function evaluation given by f^*. Let $\mathcal{A}$ continue to denote the true j-dimensional AS of f. Denote the ASTARS minimizer with $x^*_{\mathcal{A}}$ and corresponding noiseless function evaluation $f^*_{\mathcal{A}}$. Assume $||x^* - x^*_{\mathcal{A}}||^2_\Lambda < \infty$, where $||\cdot||_\Lambda$ denotes a norm. Then we may bound the difference between $|f^* - f^*_{\mathcal{A}}|$ with*

$$|f^* - f^*_{\mathcal{A}}| \leq \sqrt{a_1(q_{j+1} + \cdots + q_P)}, \tag{2.3.87}$$

where $0 \leq a_1 < \infty$ is an eigenvalue of the positive-semi definite matrix $(x^* - x^*_{\mathcal{A}})(x^* - x^*_{\mathcal{A}})^\top$ and the q's are our notations for the exact eigenvalues associated with the eigendecomposition of the sensitivity matrix W. Also, $a_1 = ||P_{\mathcal{I}}(x^{(0)} - x^*)||^2_\Lambda$.

Proof: We bound the quantity $(f^* - f^*_{\mathcal{A}})^2$. First, we expand $f^* = f(x^*)$ around $x^*_{\mathcal{A}}$ using a special case of Taylor's theorem, sometimes called the *Extended Mean Value Theorem*. For $c \in [0,1]$ and $z := cx^*_{\mathcal{A}} + (1-c)x^*$ we have $f^* = f(x^*) = f(x^*_{\mathcal{A}}) + \nabla f(z)^\top (x^* - x^*_{\mathcal{A}})$. Note that since the components of $x^*_{\mathcal{A}}$ and x^* must match for indices $i = 1, \ldots, j$ (by definition), the point z varies along $\mathcal{I}$ only; that is, z is fixed in $\mathcal{A}$. Thus, $\nabla f(z) = \nabla_{\mathcal{I}} f(z)$, the gradient taken in the inactive subspace only. The expansion of f^* around $x^*_{\mathcal{A}}$ allows us to write $(f^* - f^*_{\mathcal{A}})^2 = \left(\nabla_{\mathcal{I}} f(z)^\top (x^* - x^*_{\mathcal{A}})\right)^2 = \nabla_{\mathcal{I}} f(z)^\top A \nabla_{\mathcal{I}} f(z)$, where $A := (x^* - x^*_{\mathcal{A}})(x^* - x^*_{\mathcal{A}})^\top$ is a $P \times P$ matrix. Note that A is a square, positive semi-definite, rank 1 matrix. Hence, it has 1 eigenvalue that is positive or zero, which we denote with $a_1 \geq 0$; all other $P - 1$ eigenvalues are 0. We find $a_1 = ||x^* - x^*_{\mathcal{A}}||^2_\Lambda$ and so by definition, $a_1 = ||P_{\mathcal{I}}(x^{(0)} - x^*)||^2_\Lambda$. Observe $a_1 < \infty$ because $||x^* - x^*_{\mathcal{A}}||_\Lambda < \infty$ and also $a_1 = 0 \iff x^* = x^*_{\mathcal{A}}$, in which case $f^* = f^*_{\mathcal{A}}$ so that $|f^* - f^*_{\mathcal{A}}| = 0$. We have $(f^* - f^*_{\mathcal{A}})^2 \leq a_1 \nabla_{\mathcal{I}} f(z)^\top \nabla_{\mathcal{I}} f(z)$. Applying the expectation over $\mathcal{I}$ to both sides (where the left-hand side is constant with respect to this expectation) we have $(f^* - f^*_{\mathcal{A}}) \leq a_1 \mathbb{E}_{\mathcal{I}} \left(\nabla_{\mathcal{I}} f(z)^\top \nabla_{\mathcal{I}} f(z)\right)$. Citing Constantine *et al.* (2014) Lemma 2.2, we have $\mathbb{E}_{\mathcal{I}} \left(\nabla_{\mathcal{I}} f(z)^\top \nabla_{\mathcal{I}} f(z)\right) < q_{j+1} + \cdots + q_P$, where q_i, $i = j+1, \ldots, P$ are the last $P - j$ eigenvalues of the sensitivity matrix W. Hence, $(f^* - f^*_{\mathcal{A}})^2 \leq a_1(q_{j+1} + \cdots + q_P)$. Applying a square root to both sides, we obtain (2.3.87). ■

We now present a statement about the convergence of ASTARS to f^* by combining the previous two results. Recall $\phi_0 := f(x^{(0)})$ and $\phi_k^{\mathcal{A}} := \mathbb{E}_{\mathcal{Q}_{k-1},\mathcal{U}_{k-1}^{\mathcal{A}}}(f(x^{(k)}))$, $k \geq 1$, where the $x^{(k)}$'s are ASTARS iterates.

ASTARS Theorem 5: *Let random vectors $u_{\mathcal{A}}^{(k)}$ be drawn according to 2; $f \in \mathcal{C}^{1,1}(\Lambda)$ and f is convex; and that the i.i.d. noise draws $\epsilon(\xi)$ are additive, zero mean, with bounded variance σ^2 for all ξ. Let $\{x^{(k)}\}_{k\geq 0}$ denote a sequence of ASTARS iterates formed using a fixed active step length $h_{\mathcal{A}}$ and fixed active smoothing $\mu = \mu_{\mathcal{A}}^*$ (both given in (2.2.2)) for all ASTARS iterates k. For any total number of ASTARS iterations $M \geq 1$,*

$$\sum_{k=0}^{M} \frac{\phi_k^{\mathcal{A}} - f^*}{M+1} \leq \frac{4L_1(j+4)||P_{\mathcal{A}}(x^{(0)} - x^*)||^2}{(M+1)} + \frac{3\sigma(j+4)}{5\sqrt{2}} + \sqrt{a_1(q_{j+1} + \cdots + q_P)} \tag{2.3.88}$$

Proof: By the triangle inequality, we have $|\phi_k - f^*| \leq |\phi_k - f_{\mathcal{A}}^*| + |f^* - f_{\mathcal{A}}^*|$. Noting that $\phi_k - f_{\mathcal{A}}^* > 0$ for all k, we can bound the left-hand side of (2.3.86), writing

$$\sum_{k=0}^{M} \frac{\phi_k - f^*}{M+1} \leq \sum_{k=0}^{M} \frac{\phi_k - f_{\mathcal{A}}^*}{M+1} + |f^* - f_{\mathcal{A}}^*|. \tag{2.3.89}$$

Now the first term on the right-hand side of (2.3.89) is bounded by ASTARS Corollary 3 and the second term is bounded by ASTARS Corollary 4. Plugging in those bounds, we obtain (2.3.88). ■

We use the results above to analyze the complexity of ASTARS in the following remark.

Remark: We now mimic the complexity analysis we performed in the preceding section for STARS with approximated hyperparameters for our case in this section, ASTARS with correct hyperparameters and $\mathcal{A}$. Let $||\cdot||$ denote a norm in Λ throughout. Define $R_{\mathcal{A}}^2$ as a bound $||P_{\mathcal{A}}(x^{(0)} - x^*)||^2 \leq R_{\mathcal{A}}^2$. We recall $a_1 = ||P_{\mathcal{I}}(x^{(0)} - x^*)||_\Lambda^2$ and define a bound $a_1 \leq R_{\mathcal{I}}^2$.

Now define $x^\dagger := \text{argmin}_x\{f(x) : x \in \{x^{(0)}, \ldots, x^{(M)}\}\}$ and $\phi_\dagger := \mathbb{E}_{\mathcal{U}_{k-1},\mathcal{P}_{k-1}}(f(x^\dagger))$. Then the value $\phi_\dagger - f^*$ must be less than or equal to the average improvement for any given run of STARS; that is, (2.3.88), along with our new definitions, implies that

$$\phi_\dagger - f^* \leq \sum_{k=0}^{M} \frac{\phi_k - f^*}{M+1} \leq \frac{4L_1(j+4)}{(M+1)} R_{\mathcal{A}}^2 + \frac{3\sigma(j+4)}{5\sqrt{2}} + R_{\mathcal{I}}\sqrt{(q_{j+1} + \cdots + q_P)}. \tag{2.3.90}$$

We assume that we wish to achieve a final accuracy of $\epsilon_{\text{tol}} > 0$. Then we will need $\phi_\dagger - f^* \leq \epsilon_{\text{tol}}$. If we take

$$\frac{3\sigma(j+4)}{5\sqrt{2}} \leq \frac{\epsilon_{\text{tol}}}{3}, \tag{2.3.91}$$

then we must require that the noise not exceed the following threshold:

$$\sigma \leq \frac{5\sqrt{2}\epsilon_{\text{tol}}}{9\sigma(j+4)}. \tag{2.3.92}$$

If we satisfy (2.3.92) and also have

$$R_{\mathcal{I}}\sqrt{(q_{j+1}+\cdots+q_P)} \leq \frac{\epsilon_{\text{tol}}}{3}, \tag{2.3.93}$$

then we can achieve ϵ_{tol} accuracy as long as

$$\frac{4L_1(j+4)}{(M+1)}R_{\mathcal{A}}^2 \leq \frac{\epsilon_{\text{tol}}}{3} \iff M \geq \frac{12L_1(j+4)R_{\mathcal{A}}^2}{\epsilon_{\text{tol}}} - 1. \tag{2.3.94}$$

Hence, we achieve ϵ_{tol} accuracy as long as: the noise is small enough; the eigenvalues of the inactive subspace and distance from x^* to $x^*_{\mathcal{A}}$ is small enough; and M is large enough. Details of those required bounds given by (2.3.92), (2.3.93), and (2.3.94), respectively. With these assumptions, we achieve ϵ_{tol} accuracy in

$$M \sim \mathcal{O}\left(\frac{L_1 j R_{\mathcal{A}}^2}{\epsilon_{\text{tol}}}\right). \tag{2.3.95}$$

This analysis also shows that given a particular variance in the noise, σ, as well as the term involving the eigenvalues of the inactive subspace and distance from x^* to $x^*_{\mathcal{A}}$, the achievable accuracy can be no better (i.e., less) than the value given below:

$$\epsilon_{\text{tol}} \geq \max\left\{\frac{9(j+4)}{5\sqrt{2}}\sigma, 3R_{\mathcal{I}}\sqrt{(q_{j+1}+\cdots+q_p)}\right\}. \tag{2.3.96}$$

2.3.4 FAASTARS Convergence

Now we focus on analyzing the convergence of FAASTARS. Here, we must consider that FAASTARS uses approximate information both for hyperparameters and for the AS in its phases. We have already analyzed the convergence of performing STARS with estimated hyperparameters in 2.3.2, which corresponds to the first phase of FAASTARS. Before we can state our main result about the convergence of FAASTARS, we will need results analogous to those in 2.3.3, but with estimated hyperparameters *and* and estimated AS $\tilde{A}$. We first reintroduce the approximately-optimal ASTARS hyperparameters, need to perform the third and final phase of FAASTARS:

$$\hat{\mu}^*_{\tilde{\mathcal{A}}} := \left(\frac{8\hat{\sigma}^2 \tilde{j}}{\hat{L_1}^2 (\tilde{j}+6)^3} \right)^{1/4} \qquad \hat{h}_{\tilde{\mathcal{A}}} := (4\hat{L_1}(\tilde{j}+4))^{-1}. \tag{2.3.97}$$

We also must modify our definitions from the previous section to account for the approximated subspace. Recall that the third phase of FAASTARS will begin with the initial iterate $x^{(M_{\mathcal{A}})}$ (the last iterate from phase two). Define $x^*_{\tilde{\mathcal{A}}} := P_{\tilde{\mathcal{A}}}(x^*) + P_{\tilde{\mathcal{I}}}(x^{(M_{\mathcal{A}})})$ with its associated stochastic-free $f^*_{\tilde{\mathcal{A}}} := f(x^*_{\tilde{\mathcal{A}}})$. Observe that with our definitions, $x^* - x^*_{\tilde{\mathcal{A}}} = P_{\tilde{\mathcal{I}}}(x^{(M_{\mathcal{A}})})$. We define $f|_{\tilde{\mathcal{A}}}(\lambda) := f(V_{\tilde{\mathcal{A}}} V_{\tilde{\mathcal{A}}}^\top \lambda) = f(P_{\tilde{\mathcal{A}}}(\lambda))$ and let $f_{\tilde{\mathcal{A}}} := f|_{\tilde{\mathcal{A}}}$. Note that $f_{\tilde{\mathcal{A}}}$ is convex since f is convex and we can define $f_{\tilde{\mathcal{I}}}$ analogously. Also, when we evaluate gradients for points $x \in \tilde{\mathcal{A}}$, we note we obtain the object $\nabla_{\tilde{\mathcal{A}}} f(x) \in \tilde{\mathcal{A}}$, and similarly for $\tilde{\mathcal{I}}$.

We begin by providing a modification to ASTARS Corollary 2 for the case of estimated hyperparameters and $\tilde{\mathcal{A}}$. The proof is a blend of the proofs of STARS Theorem 4.3 (Modified) and ASTARS Corollary 2, with additional consideration for the now $\tilde{j}$-dimensional $\tilde{\mathcal{A}}$.

FAASTARS Corollary 1 (Modified ASTARS Corollary 2): *Let the vectors $u^{(k)}_{\tilde{\mathcal{A}}}$ denote those drawn using Algorithm 5; let $f \in \mathcal{C}^{1,1}(\Lambda)$ and assume f is convex; and assume that the i.i.d. noise draws $\epsilon(\xi)$ are additive, zero mean, with bounded variance σ^2 for all ξ. We assume we have fixed estimates $\hat{L_1}$ and $\hat{\sigma}$ with $K_1 > 0$ and $K_2 > 0$ as in* (2.3.22)*. By fixing the step size as $\hat{h}_{\tilde{\mathcal{A}}}$ in* (2.3.97)*, the approximately active smoothing parameter $\hat{\mu}^*_{\tilde{\mathcal{A}}}$ in* (2.3.97) *minimizes the error between the gradient oracle in Algorithm 5 and the true directional derivative of f in the direction $u^{(k)}_{\tilde{\mathcal{A}}}$ in the $\tilde{j}$-dimensional AS $\tilde{\mathcal{A}}$. That is, $\mathcal{E}(\mu)$ in* (2.3.21) *(with $u = u^{(k)}_{\tilde{\mathcal{A}}}$) is minimized by the choice $\mu = \hat{\mu}^*_{\tilde{\mathcal{A}}}$. In particular, we have the bound*

$$\mathbb{E}_{u^{(k)}_{\tilde{\mathcal{A}}}, \xi_1, \xi_2} \left(\mathcal{E}^{\tilde{\mathcal{A}}} \left(\mu^*_{\tilde{\mathcal{A}}} \right) \right) \leq \frac{K_1 + K_2}{\sqrt{2K_1K_2}} \sigma L_1 \sqrt{\tilde{j}(\tilde{j}+6)^3}. \tag{2.3.98}$$

Proof: The proof is identical to the proof of STARS Theorem 4.3 (Modified) and ASTARS Corollary 2 (making the usual substitutions), until we formulate (as in (2.3.32))

$$\mathbb{E}_{u,\xi_1,\xi_2}(\mathcal{E}^{\tilde{\mathcal{A}}}(\mu)) \leq \frac{1}{\mu^2} \mathbb{E}_u \left(2\sigma^2 ||u||^2 + \frac{L_1^2}{4} \mu^4 ||u||^6 \right) \tag{2.3.99}$$

and proceed to bound the right hand side. Applying ASTARS Proposition 1, we have $(u^{(k)}_{\tilde{\mathcal{A}}})_p \sim N(0,1)$ for $p = 1, \ldots, P$. Taking $u = u^{(k)}_{\tilde{\mathcal{A}}}$ and noting $u^{(k)}_{\tilde{\mathcal{A}}} \in \tilde{\mathcal{A}}$ (and $\dim \tilde{\mathcal{A}} = \tilde{j}$), we apply the bounds on the moments M_p of $||u||^p$ given in (2.3.11). Using $p = 2$ and $p = 6$ – but replacing the AS dimension j with $\tilde{j}$ – we have

$$\mathbb{E}_{u^{(k)}_{\tilde{\mathcal{A}}}, \xi_1, \xi_2}(\mathcal{E}(\mu)) \leq (2\sigma^2 \tilde{j}) \frac{1}{\mu^2} + \left(\frac{L_1^2 (\tilde{j}+6)^3}{4} \right) \mu^2. \tag{2.3.100}$$

We again observe that the right-hand side of the above inequality is uniformly convex for $\mu > 0$ with minimizer $\mu^*_{\tilde{\mathcal{A}}} := \left(\frac{8\sigma^2\tilde{j}}{L_1^2(\tilde{j}+6)^3}\right)^{1/4}$. Analogously to the proof of STARS Theorem 4.3 (Modified), our optimal choice of smoothing, given the information we have available, will require us to swap out L_1 and σ^2 in $\mu^*_{\mathcal{A}}$ with their estimates, $\hat{L_1}$ and $\hat{\sigma}^2$, and to swap out j for $\tilde{j}$, recovering $\hat{\mu}^*_{\tilde{\mathcal{A}}}$ (2.3.97). This particular choice $\mu = \hat{\mu}^*_{\tilde{\mathcal{A}}}$ can be plugged into (2.3.100), which gives us the bound in (2.3.98), our main result. ■

Next, we redefine $\mathcal{Q}_k := \{\xi_k\}_{k=1}^{M_{\mathcal{A}}}$ and $\mathcal{U}_k := \{u^{(k)}\}_{k=1}^{M_{\mathcal{A}}}$, which are two sets containing all random variables that appear in FAASTARS' regular STARS burn-in phase, iterations $k = 1, \ldots, M_{\mathcal{A}}$. Likewise, we extend the definitions of each set so that $\mathcal{Q}_k = \{\xi_k\}_{k=M_{\mathcal{A}}+1}^{M}$ and $\mathcal{U}_k^{\tilde{\mathcal{A}}} = \{u^{(k)}_{\tilde{\mathcal{A}}}\}_{k=M_{\mathcal{A}}+1}^{M}$ also contain all random variables that appear in FAASTARS' approximate ASTARS phase, iterations $k = M_{\mathcal{A}}+1, \ldots, M$.

Let $\phi_0 := f(x^{(0)})$, $\phi_k := \mathbb{E}_{\mathcal{Q}_{k-1},\mathcal{U}_{k-1}}(f(x^{(k)}))$, $1 \leq k \leq M_{\mathcal{A}}$, and $\phi_k := \mathbb{E}_{\mathcal{Q}_{k-1},\mathcal{U}^{\tilde{\mathcal{A}}}_{k-1}}(f(x^{(k)}))$, $M_{\mathcal{A}}+1 \leq k \leq M$ where the $x^{(k)}$'s are now STARS iterates for $1 \leq k \leq M_{\mathcal{A}}$, and FAASTARS iterates for $M_{\mathcal{A}}+1 \leq k \leq M$.

FAASTARS Corollary 2 (FAASTARS, approximate ASTARS phase result): *For all FAASTARS iterates in phase 3, $k = M_{\mathcal{A}}+1, \ldots, M$, let the vectors $u^{(k)}_{\tilde{\mathcal{A}}}$ denote those drawn using Algorithm 5. Also, let $f \in \mathcal{C}^{1,1}(\Lambda)$ with f convex, and let i.i.d. noise draws $\epsilon(\xi)$ be additive, zero mean, with bounded variance σ^2 for all appearing ξ. We assume we have fixed estimates $\hat{L_1}$ and $\hat{\sigma}$ with $K_1 > 0$ and $K_2 > 0$ as in* (2.3.22)*. Let $\tilde{\mathcal{A}}$ denote the approximated $\tilde{j}$-dimensional AS of f. Let $x^{(M_{\mathcal{A}})}$ be fixed and given and let $\{x^{(k)}\}_{k=M_{\mathcal{A}}+1}^{M}$ denote a sequence of FAASTARS iterates formed using the approximate active hyperparameters in* (2.3.97)*).* ***Finally, we require*** $0 < K_1 < 4$ ***and*** $K_2 > 0$***, the values defined in*** (2.3.22)***.*** *Then for any $M - M_{\mathcal{A}}$ total number of approximate ASTARS iterations within FAASTARS, $k = M_{\mathcal{A}}+1, \ldots, M$,*

$$\sum_{k=M_{\mathcal{A}}}^{M} \frac{\phi_k - f^*_{\tilde{\mathcal{A}}}}{M - M_{\mathcal{A}}} \leq \frac{4L_1(\tilde{j}+4)||P_{\tilde{\mathcal{A}}}(x^{(M_{\mathcal{A}})} - x^*)||^2}{\sqrt{K_1}(2-\sqrt{K_1})(M - M_{\mathcal{A}}+1)} + \frac{4\sigma(\tilde{j}+4)}{\sqrt{2K_2}(2-\sqrt{K_1})}C_5, \tag{2.3.101}$$

where $C_5 := \sqrt{K_1} \cdot 0.036 + \frac{3K_1+K_2}{16} \cdot 1.034$.

Proof: The proof is almost identical to the proof of ASTARS Corollary 3, but with $\tilde{j}$'s replacing the roles of j's (since we take steps with $\tilde{j}$-dimensional $u^{(k)}_{\tilde{\mathcal{A}}}$ vectors and not $u^{(k)}_{\mathcal{A}} \in \mathcal{A}$), $\hat{\mu}^*_{\tilde{\mathcal{A}}}$ replacing $\mu^*_{\mathcal{A}}$, and $x^*_{\tilde{\mathcal{A}}}$ replacing $x^*_{\mathcal{A}}$ (since we are converging to the minimum of $f_{\tilde{\mathcal{A}}}$). Here, K_1 and K_2 are not necessarily equal to 1 as before, since we are assuming inexact active hyperparameters, formed with the estimates $\hat{L_1}$ and $\hat{\sigma}^2$. We outline the required changes one must make to the proofs of STARS Lemma 4.4 (Modified) and STARS Theorem 4.5 (Modified) to obtain our desired result. These changes essentially amount

to keeping the logic from STARS Lemma 4.4 (Modified) and STARS Theorem 4.5 (Modified) to account for K_1 and K_2 but to replacing j with $\tilde{j}$.

We begin by obtaining a bound analogous to that of STARS Lemma 4.4 (Modified), but note that the approximate ASTARS steps are taken with random vectors $u_{\tilde{\mathcal{A}}}^{(k)} \in \tilde{\mathcal{A}}$, and $\dim \tilde{\mathcal{A}} = \tilde{j}$. First, we replace $u^{(k)}$ in (2.3.37) with $u_{\tilde{\mathcal{A}}}^{(k)}$, so $g_0(x^{(k)}) := \langle \nabla f(x^{(k)}), u_{\tilde{\mathcal{A}}}^{(k)} \rangle u_{\tilde{\mathcal{A}}}^{(k)}$. Similarly, note that we set $u^{(k)} = u_{\tilde{\mathcal{A}}}^{(k)}$ in (2.3.20). Also, let $s_{\hat{\mu}_{\tilde{\mathcal{A}}}^*} = s_{\hat{\mu}_{\tilde{\mathcal{A}}}^*}^{\tilde{\mathcal{A}}}$ for cleaner notation. Then, using FAASTARS Corollary 1 – instead of ASTARS Corollary 2 – (2.3.38) becomes

$$\mathbb{E}\left(||s_{\hat{\mu}_{\tilde{\mathcal{A}}}^*}||^2 - 2\langle s_{\hat{\mu}_{\tilde{\mathcal{A}}}^*}, g_0(x^{(k)})\rangle + ||g_0(x^{(k)})||^2\right) \leq \frac{K_1 + K_2}{\sqrt{2K_1K_2}} \sigma L_1 \sqrt{\tilde{j}(\tilde{j}+6)^3}. \tag{2.3.102}$$

Continuing with $u_{\tilde{\mathcal{A}}}^{(k)}$ replacing $u^{(k)}$, we proceed identically, noting $C_1 = \frac{K_1+K_2}{\sqrt{2K_1K_2}} \sigma L_1 \sqrt{\tilde{j}(\tilde{j}+6)^3}$. Modifying (2.3.19) for $x^{(k)} \in \tilde{\mathcal{A}}$ with $\dim \tilde{\mathcal{A}} = \tilde{j}$, we have

$$\mathbb{E}\left(||s_{\hat{\mu}_{\tilde{\mathcal{A}}}^*}||^2\right) \leq 2(\tilde{j}+4)||\nabla f(x^{(k)})||^2 + \frac{(\hat{\mu}_{\tilde{\mathcal{A}}}^*)^2 L_1^2}{2}(\tilde{j}+6)^3 + C_1, \tag{2.3.103}$$

Plugging in the value of $\hat{\mu}_{\tilde{\mathcal{A}}}^*$, we obtain the bound

$$\mathbb{E}\left(||s_{\hat{\mu}_{\tilde{\mathcal{A}}}^*}||^2\right) \leq 2(\tilde{j}+4)||\nabla f(x^{(k)})||^2 + C_2, \tag{2.3.104}$$

where $C_2 := \frac{3K_1+K_2}{\sqrt{2K_1K_2}} L_1 \sigma \sqrt{\tilde{j}(\tilde{j}+6)^3} = \frac{3K_1+K_2}{\sqrt{2}} \hat{L_1} \hat{\sigma} \sqrt{\tilde{j}(\tilde{j}+6)^3}$. The bound in (2.3.104) is the analogous result to STARS Lemma 4.4 (Modified) in the case of ASTARS performed in the estimated $\tilde{\mathcal{A}}$ and with inexact hyperparameters.

We now proceed to proving the analogous result to STARS Theorem 4.5 (Modified) in our case. We redefine $r_k := ||x^{(k)} - x_{\tilde{\mathcal{A}}}^*||$ for approximate ASTARS iterates $x^{(k)}$. The first three equations appearing in the proof of STARS Theorem 4.5 (Modified), (2.3.48) through (2.3.50), are nearly identical for this proof – one must replace x^* with $x_{\tilde{\mathcal{A}}}^*$, replace $\hat{h}$ with $\hat{h}_{\tilde{\mathcal{A}}}$, and note that here we have $s_{\mu^{(k)}} = s_{\hat{\mu}_{\tilde{\mathcal{A}}}^*}^{\tilde{\mathcal{A}}}$. Then, invoking (2.3.104), we rewrite (2.3.51) in our case as

$$\mathbb{E}(r_{k+1}^2) \leq r_k^2 - 2\hat{h}_{\tilde{\mathcal{A}}} \langle \nabla f_\mu(x^{(k)}), x^{(k)} - x_{\tilde{\mathcal{A}}}^* \rangle + \hat{h}_{\tilde{\mathcal{A}}}^2 \left(2(\tilde{j}+4)||\nabla f(x^{(k)})||^2 + C_2\right), \tag{2.3.105}$$

where we recall $C_2 = \frac{3K_1+K_2}{\sqrt{2}} \hat{L_1} \hat{\sigma} \sqrt{\tilde{j}(\tilde{j}+6)^3}$ here. Now, again, equations (2.3.52) through (2.3.56) are identical for this proof as long as $x_{\tilde{\mathcal{A}}}^*$ replaces x^* and $\hat{h}_{\tilde{\mathcal{A}}}$ replaces $\hat{h}$ throughout, and similarly for equations (2.3.57) through (2.3.59) with the additional needed replacement of $\tilde{j}$ for P and using C_2 as we have de-

fined it here. Taking $C_3 = \hat{h}_{\tilde{\mathcal{A}}}\hat{\mu}^*_{\tilde{\mathcal{A}}}L_1\tilde{j} + \hat{h}^2_{\tilde{\mathcal{A}}}C_2$, (2.3.60) holds (with the usual replacments) and plugging $\hat{h}_{\tilde{\mathcal{A}}}$ in (2.3.105):

$$\mathbb{E}(r^2_{k+1}) \leq r^2_k - \frac{1}{4L_1(\tilde{j}+4)}(f(x^{(k)}) - f(x^*_{\tilde{\mathcal{A}}})) + C_3. \tag{2.3.106}$$

Now plugging $\hat{h}_{\tilde{\mathcal{A}}}$ into C_3, we obtain $C_3 \leq \frac{\sqrt{K_1}}{\sqrt{K_2}} \cdot \frac{\sigma}{\sqrt{2}L_1}\left(\sqrt{K_1}\cdot 0.036 + \frac{3K_1+K_2}{16}\cdot 1.034\right) =: C_4$, so

$$\mathbb{E}(r^2_{k+1}) \leq r^2_k - \frac{1}{4L_1(\tilde{j}+4)}(f(x^{(k)}) - f(x^*_{\tilde{\mathcal{A}}})) + C_4. \tag{2.3.107}$$

We may now apply the expectation over $\mathcal{P}_k$ and $\mathcal{U}^{\tilde{\mathcal{A}}}_k$, $k = M_{\mathcal{A}}, \ldots, M$, rearrange, and sum over $k = M_{\mathcal{A}}, \ldots, M$, similarly to before. We obtain a modification to (2.3.68) with

$$\sum_{k=M_{\mathcal{A}}}^{M} \frac{\phi_k - f^*_{\tilde{\mathcal{A}}}}{M - M_{\mathcal{A}} + 1} \leq \frac{4L_1(\tilde{j}+4)}{(M - M_{\mathcal{A}}+1)}\left(r_{M_{\mathcal{A}}} - \mathbb{E}_{\mathcal{U}_k,\mathcal{P}_k}(r^2_{k+1})\right) + 4L_1(\tilde{j}+4)C_4. \tag{2.3.108}$$

We drop the strictly negative term (again involving $\mathbb{E}_{\mathcal{P}_k,\mathcal{U}^{\tilde{\mathcal{A}}}_k}$) and plug in the definition of $r_{M_{\mathcal{A}}} = ||x^{(M_{\mathcal{A}})} - x^*_{\tilde{\mathcal{A}}}||^2$. Then, recalling $x^*_{\tilde{\mathcal{A}}} = P_{\tilde{\mathcal{A}}}(x^*) + P_{\tilde{\mathcal{I}}}(x^{(M_{\mathcal{A}})})$, writing $x^{(M_{\mathcal{A}})} = P_{\tilde{\mathcal{A}}}(x^{(M_{\mathcal{A}})}) + P_{\tilde{\mathcal{I}}}(x^{(M_{\mathcal{A}})})$, and noting that $P_{\tilde{\mathcal{A}}}$ is linear, we have $r_{M_{\mathcal{A}}} = ||P_{\tilde{\mathcal{A}}}(x^{(M_{\mathcal{A}})} - x^*)||^2$. Plugging in the value of C_4, we obtain (2.3.101). ■

FAASTARS Corollary 2 shows that its iterates during its third phase (approximate ASTARS) converge to $f^*_{\tilde{\mathcal{A}}}$; however, we need a result for the convergence to f^* like we did in the last section. In particular, we need a result analogous to ASTARS Corollary 4, generally regarding the distance between evaluations of f and $f_{\tilde{\mathcal{A}}}$ (instead of $f_{\mathcal{A}}$) at their respective minimizers, x^* and $x^*_{\tilde{\mathcal{A}}}$. Recall, by $f_{\tilde{\mathcal{A}}}$, we mean $f_{\mathcal{A}}(\lambda) := f(V_{\tilde{\mathcal{A}}}V^\top_{\tilde{\mathcal{A}}}\lambda)$, where the application of $V_{\tilde{\mathcal{A}}}V^\top_{\tilde{\mathcal{A}}}$ is a linear transformation of λ into $\tilde{\mathcal{A}}$, as in Constantine *et al.* (2014), which we again rely on heavily for the following result.

FAASTARS Corollary 3: *Let $x^{(M_{\mathcal{A}})}$ denote any initial iterate for phase 3 of FAASTARS. Let $\mathcal{A}$ continue to denote the true j-dimensional AS of f and let $\tilde{\mathcal{A}}$ denote the approximate $\tilde{j}$-dimensional AS of f and the true minimizer of $f_{\tilde{\mathcal{A}}}$ with $x^*_{\tilde{\mathcal{A}}}$ and corresponding noiseless function evaluation $f^*_{\tilde{\mathcal{A}}}$. Assume $||x^* - x^*_{\tilde{\mathcal{A}}}||_\Lambda < \infty$, where $||\cdot||_\Lambda$ denotes a norm. Assume $\tilde{j} = j$ and for $\delta > 0$, $||V - \tilde{V}||_2 < \delta$, where $||\cdot||_2$ here denotes the matrix 2-norm induced by the 2-norm in Λ. Also, assume the sign of $(\tilde{V}_{\tilde{\mathcal{I}}})_i$, the i-th column of $\tilde{V}_{\tilde{\mathcal{I}}}$ to be chosen so that $||(\tilde{V}_{\tilde{\mathcal{I}}})_i - (V_{\mathcal{I}})_i||_2$ is minimized for $i = j+1, \ldots, P$, where $||\cdot||_2$ denotes the vector 2-norm. Then the difference between $|f^* - f^*_{\tilde{\mathcal{A}}}|$ is bounded by*

$$|f^* - f^*_{\tilde{\mathcal{A}}}| \leq \sqrt{\tilde{a}_1}\left(\delta\sqrt{q_1 + \cdots + q_j} + \sqrt{q_{j+1} + \cdots + q_P}\right), \tag{2.3.109}$$

where $0 \leq \tilde{a}_1 < \infty$ *is an eigenvalue of the positive-semi definite matrix* $(x^* - x^*_{\tilde{\mathcal{A}}})(x^* - x^*_{\tilde{\mathcal{A}}})^\top$ *and the q's are our notations for the exact eigenvalues associated with the eigendecomposition of the sensitivity matrix* W*. Also,* $\tilde{a}_1 = ||P_{\tilde{\mathcal{I}}}(x^{(M_{\mathcal{A}})} - x^*)||^2_\Lambda$.

Proof: We bound the quantity $(f^* - f^*_{\tilde{\mathcal{A}}})^2$. First, we expand $f* = f(x^*)$ around $x^*_{\tilde{\mathcal{A}}}$ by applying *Extended Mean Value Theorem* analogously to ASTARS Corollary 4. For $c \in [0,1]$ and $\tilde{z} := cx^*_{\tilde{\mathcal{A}}} + (1-c)x^*$ we have $f^* = f(x^*) = f(x^*_{\tilde{\mathcal{A}}}) + \nabla f(\tilde{z})^\top (x^* - x^*_{\tilde{\mathcal{A}}})$. Note that since the components of $x^*_{\tilde{\mathcal{A}}}$ and rotated $V_{\tilde{\mathcal{A}}} V^\top_{\tilde{\mathcal{A}}} x^*$ must match for indices $i = 1, \ldots, j$, the point $\tilde{z}$ varies along $\tilde{\mathcal{I}}$ only; that is, $\tilde{z}$ is fixed in $\tilde{\mathcal{A}}$. Thus, $\nabla f(\tilde{z}) = \nabla_{\tilde{\mathcal{I}}} f(\tilde{z})$, the gradient taken in the approximate inactive subspace only.

The expansion of f^* around $x^*_{\tilde{\mathcal{A}}}$ writes $(f^* - f^*_{\tilde{\mathcal{A}}})^2 = \left(\nabla_{\tilde{\mathcal{I}}} f(\tilde{z})^\top (x^* - x^*_{\tilde{\mathcal{A}}})\right)^2 = \nabla_{\tilde{\mathcal{I}}} f(\tilde{z})^\top \tilde{A} \nabla_{\tilde{\mathcal{I}}} f(\tilde{z})$, where $\tilde{A} := (x^* - x^*_{\tilde{\mathcal{A}}})(x^* - x^*_{\tilde{\mathcal{A}}})^\top$ is a $P \times P$ matrix. Note that $\tilde{A}$ is a square, positive semi-definite, rank 1 matrix. Hence, it has 1 eigenvalue that is positive or zero, which we denote with $\tilde{a}_1 \geq 0$; all other $P - 1$ eigenvalues are 0. We find $\tilde{a}_1 = ||x^* - x^*_{\tilde{\mathcal{A}}}||^2_\Lambda$ so by definition, $\tilde{a}_1 = ||P_{\tilde{\mathcal{I}}}(x^{(M_{\mathcal{A}})} - x^*)||^2_\Lambda$. Observe $\tilde{a}_1 < \infty$ because $||x^* - x^*_{\tilde{\mathcal{A}}}||_\Lambda < \infty$ and also $\tilde{a}_1 = 0 \iff x^* = x^*_{\tilde{\mathcal{A}}}$, which case $f^* = f^*_{\tilde{\mathcal{A}}}$ so that $|f^* - f^*_{\tilde{\mathcal{A}}}| = 0$. Then $(f^* - f^*_{\tilde{\mathcal{A}}})^2 \leq \tilde{a}_1 \nabla_{\tilde{\mathcal{I}}} f(\tilde{z})^\top \nabla_{\tilde{\mathcal{I}}} f(\tilde{z})$.

Recall $\tilde{j} = j$ and for $\delta > 0$, $||V - \tilde{V}||_2 < \delta$. Then we have *comparable partitions*, in the sense that the submatrices $V_{\mathcal{A}}$ and $\tilde{V}_{\tilde{\mathcal{A}}}$ (of V and $\tilde{V}$ respectively) are both j-dimensional, and likewise the submatrices $V_{\mathcal{I}}$ and $V_{\tilde{\mathcal{I}}}$ are both $P - j$-dimensional. We shall need the results from Lemma 3.4 in Constantine *et al.* (2014) which state $||V^\top_{\mathcal{I}} \tilde{V}_{\tilde{\mathcal{I}}}||_2 \leq 1$ and $||V^\top_{\mathcal{A}} \tilde{V}_{\tilde{\mathcal{I}}}||_2 \leq \delta$, where $||\cdot||_2$ denotes the matrix 2-norm. Note that Lemma 3.4 requires that the sign of $(\tilde{V}_{\tilde{\mathcal{I}}})_i$, the i-th column of $\tilde{V}_{\tilde{\mathcal{I}}}$ to be chosen so that $||(\tilde{V}_{\tilde{\mathcal{I}}})_i - (V_{\mathcal{I}})_i||_2$ is minimized for $i = j+1, \ldots, P$, where $||\cdot||_2$ denotes the vector 2-norm. Now the chain rule provides $\nabla_{\tilde{\mathcal{I}}} f = V^\top_{\mathcal{I}} \tilde{V}_{\tilde{\mathcal{I}}} \nabla_{\mathcal{I}} f + V^\top_{\mathcal{A}} \tilde{V}_{\tilde{\mathcal{I}}} \nabla_{\mathcal{A}} f$ (Constantine *et al.* (2014), pp. A1510).

Applying the expectation over both $\mathcal{I}$ and $\mathcal{A}$ to both sides (where the left-hand side is constant with respect to this expectation) we have $(f^* - f^*_{\tilde{\mathcal{A}}}) \leq \tilde{a}_1 \mathbb{E}_{\mathcal{A},\mathcal{I}} \left(\nabla_{\tilde{\mathcal{I}}} f(\tilde{z})^\top \nabla_{\tilde{\mathcal{I}}} f(\tilde{z})\right)$. Using the chain rule and $||V^\top_{\mathcal{I}} \tilde{V}_{\tilde{\mathcal{I}}}||_2 \leq 1$ and $||V^\top_{\mathcal{A}} \tilde{V}_{\tilde{\mathcal{I}}}||_2 \leq \delta$ gives

$$(f^* - f^*_{\tilde{\mathcal{A}}})^2 \leq \tilde{a}_1 \left(\mathbb{E}_{\mathcal{I}} \left(\nabla_{\mathcal{I}} f(\tilde{z})^\top \nabla_{\mathcal{I}} f(\tilde{z})\right) + 2\delta\, \mathbb{E}_{\mathcal{A},\mathcal{I}} \left(\nabla_{\mathcal{I}} f(\tilde{z})^\top \nabla_{\mathcal{A}} f(\tilde{z})\right) + \delta^2 \mathbb{E}_{\mathcal{A}} \left(\nabla_{\mathcal{A}} f(\tilde{z})^\top \nabla_{\mathcal{A}} f(\tilde{z})\right)\right), \tag{2.3.110}$$

where linearity of $\mathbb{E}$ allows for breaking the expectation of the appearing sum into a sum of expectations taken over only $\mathcal{A}$, only $\mathcal{I}$, or both $\mathcal{A}$ and $\mathcal{I}$, depending on which types of gradients appear. The Cauchy-Schwarz inequality may be applied to the second term on the right-hand side of (2.3.110) to write

$$2\delta \mathbb{E}_{\mathcal{A},\mathcal{I}}\left(\nabla_{\mathcal{I}} f(\tilde{z})^\top \nabla_{\mathcal{A}} f(\tilde{z})\right) \leq 2\delta\, \mathbb{E}_{\mathcal{I}}\left(\nabla_{\mathcal{I}} f(\tilde{z})^\top \nabla_{\mathcal{I}} f(\tilde{z})\right) \mathbb{E}_{\mathcal{A}}\left(\nabla_{\mathcal{A}} f(\tilde{z})^\top \nabla_{\mathcal{I}} f(\tilde{z})\right), \tag{2.3.111}$$

where we can split up the two terms inside $\mathbb{E}_{\mathcal{A},\mathcal{I}}(\cdot)$ into two separate expectations taken over $\mathcal{I}$ and $\mathcal{A}$ individually, due to their independence. Substituting this bound into (2.3.86) and then factoring the resulting terms will yield

$$(f^* - f^*_{\tilde{\mathcal{A}}})^2 \leq \tilde{a}_1 \left(\mathbb{E}_{\mathcal{I}}\left(\nabla_{\mathcal{I}} f(\tilde{z})^\top \nabla_{\mathcal{I}} f(\tilde{z})\right)^{1/2} + \delta\, \mathbb{E}_{\mathcal{A}}\left(\nabla_{\mathcal{A}} f(\tilde{z})^\top \nabla_{\mathcal{A}} f(\tilde{z})\right)^{1/2}\right)^2, \tag{2.3.112}$$

Citing Constantine *et al.* (2014) Lemma 2.2, we have $\mathbb{E}_{\mathcal{A}}\left(\nabla_{\mathcal{A}} f(\tilde{z})^\top \nabla_{\mathcal{A}} f(\tilde{z})\right) < q_1 + \cdots + q_j$, where q_i, $i = 1, \ldots, j$ are the first j eigenvalues of the sensitivity matrix W and $\mathbb{E}_{\mathcal{I}}\left(\nabla_{\mathcal{I}} f(\tilde{z})^\top \nabla_{\mathcal{I}} f(\tilde{z})\right) < q_{j+1} + \cdots + q_P$, where q_i, $i = j+1, \ldots, P$ are the last $P-j$ eigenvalues of the sensitivity matrix W. Hence, $(f^* - f^*_{\mathcal{A}})^2 \leq \tilde{a}_1 \left(\delta \sqrt{q_1 + \cdots + q_j} + \sqrt{q_{j+1} + \cdots + q_P}\right)^2$. Applying a square root to both sides, we obtain (2.3.109). ■

We now present a statement about the convergence of FAASTARS to f^* by combining the previous two results, along with STARS Theorem 4.5 (Modified).

Recall for $k = 1, \ldots, M_{\mathcal{A}}$ iterates correspond to STARS with estimated hyperparameters and for $k = M_{\mathcal{A}} + 1, \ldots, M$ the iterates correspond to ASTARS with estimated hyperparameters and estimated $\mathcal{A}$.

FAASTARS Theorem 4: *For $k = M_{\mathcal{A}}, \ldots, M$, let the assumptions from FAASTARS Corollaries 3 and 4 hold. For any total number of FAASTARS iterations $M \geq 1$,*

$$\begin{aligned}\sum_{k=M_{\mathcal{A}}}^{M} \frac{\phi_k - f^*}{M - M_{\mathcal{A}} + 1} \leq & \frac{4L_1(j+4)||P_{\tilde{\mathcal{A}}}(x^{(M_{\mathcal{A}})} - x^*)||^2}{\sqrt{K_1}(2 - \sqrt{K_1})(M - M_{\mathcal{A}} + 1)} + \frac{4\sigma(j+4)}{\sqrt{2K_2}(2 - \sqrt{K_1})} C_5 \\ & + \sqrt{\tilde{a}_1}\left(\delta\sqrt{q_1 + \cdots + q_j} + \sqrt{q_{j+1} + \cdots + q_P}\right),\end{aligned} \tag{2.3.113}$$

where $C_5 := \sqrt{K_1} \cdot 0.036 + \frac{3K_1 + K_2}{16} \cdot 1.034$.

Proof: Using the triangle inequality, we have $|\phi_k - f^*| \leq |\phi_k - f^*_{\tilde{\mathcal{A}}}| + |f^* - f^*_{\tilde{\mathcal{A}}}|$. Thus, noting that the quantity $\phi_k - f^*_{\tilde{\mathcal{A}}}$ is always nonnegative, we use the triangle inequality to rewrite the summation on the left-hand side of (2.3.113) as

$$\sum_{k=M_{\mathcal{A}}}^{M} \frac{\phi_k - f^*}{M - M_{\mathcal{A}} + 1} \leq \sum_{k=M_{\mathcal{A}}}^{M} \frac{\phi_k - f^*_{\tilde{\mathcal{A}}}}{M - M_{\mathcal{A}} + 1} + |f^* - f^*_{\tilde{\mathcal{A}}}|. \tag{2.3.114}$$

Now the second sum on the right-hand side of (2.3.115) is in a more useful form for us, since the final phase of iterates, $k = M_{\mathcal{A}}, \ldots, M$ are (approximate) ASTARS iterates converging to $f^*_{\tilde{\mathcal{A}}}$ (not f^*); thus, we recall that we already proved a bound for this sum in FAASTARS Corollary 2. Note that since we assume in FAASTARS Corollary 3 that $\tilde{j} = j$, we replace $\tilde{j}$ with j in the result from FAASTARS Corollary 2. Otherwise, we just invoke FAASTARS Corollary 3, bounding the last term on the right-hand side of (2.3.114) to obtain (2.3.113). ■

We use the results above to analyze the complexity of FAASTARS in the following remark.

Remark: We again mimic the complexity analyses we performed in the preceding sections for FAASTARS. Let $||\cdot||$ denote a norm in Λ throughout. Define $R^2_{\tilde{\mathcal{A}}}$ as a bound $||P_{\tilde{\mathcal{A}}}(x^{(M_{\mathcal{A}})} - x^*)||^2 \leq R^2_{\tilde{\mathcal{A}}}$. We recall $\tilde{a}_1 = ||P_{\tilde{\mathcal{I}}}(x^{(M_{\mathcal{A}})} - x^*)||^2_\Lambda$ and define a bound $\tilde{a}_1 \leq R^2_{\tilde{\mathcal{I}}}$.

Define $x^\dagger := \text{argmin}_x\{f(x) : x \in \{x^{(0)}, \ldots, x^M\}\}$ and $\phi_\dagger := \mathbb{E}_{\mathcal{U}_{k-1},\mathcal{P}_{k-1}}(f(x^\dagger))$ as before. Then the value $\phi_\dagger - f^*$ must be less than or equal to the average improvement for any given run of STARS; that is, (2.3.113), along with our new definitions, implies that

$$\begin{aligned}\phi_\dagger - f^* \leq & \frac{4L_1(j+4)}{\sqrt{K_1}(2-\sqrt{K_1})(M - M_{\mathcal{A}} + 1)} R^2_{\tilde{\mathcal{A}}} + \frac{4\sigma(j+4)}{\sqrt{2K_2}(2-\sqrt{K_1})} C_5 \\ & + R_{\tilde{\mathcal{I}}}\left(\delta\sqrt{q_1 + \ldots + q_j} + \sqrt{q_{j+1} + \cdots + q_P}\right),\end{aligned} \tag{2.3.115}$$

Now in this analysis, we are only taking enough approximate STARS steps to learn a surrogate for f to obtain $\tilde{\mathcal{A}}$ Thus, we have $M_{\mathcal{A}} \sim \mathcal{O}(L(P))$, where L is a function of P that depends on the surrogate method used to learn $\mathcal{A}$. For instance, if we use a linear surrogate or RBF's – which begin by fitting a linear surrogate, and then later a quadratic surrogate once enough ASTARS steps are taken – then $L(P) = P$. If we use quadratic surrogates, then $L(P) = P^2$. (Typically we use quadratic surrogates for higher-quality active subspaces.)

The terms involving Phase 3 of FAASTARS (approximate ASTARS phase) are the three terms on the right-hand side of (2.3.115). We assume that we wish to achieve a final accuracy of $\epsilon_{\text{tol}} > 0$. Then we will need $\phi_\dagger - f^* \leq \epsilon_{\text{tol}}$. If we take

$$\frac{4\sigma(j+4)}{\sqrt{2K_2}(2-\sqrt{K_1})} C_5 \leq \frac{\epsilon_{\text{tol}}}{3}, \tag{2.3.116}$$

then we must require that the noise not exceed the following threshold:

$$\sigma \leq \frac{\sqrt{2K_2}(2-\sqrt{K_1})\epsilon_{\text{tol}}}{12\sigma(j+4)}. \tag{2.3.117}$$

If we satisfy (2.3.117) and also have

$$R_{\tilde{\mathcal{I}}}\left(\delta\sqrt{q_1+\ldots+q_j}+\sqrt{q_{j+1}+\cdots+q_P}\right)\leq\frac{\epsilon_{\text{tol}}}{3}, \tag{2.3.118}$$

then we can achieve ϵ_{tol} accuracy as long as

$$\frac{4L_1(j+4)}{\sqrt{K_1}(2-\sqrt{K_1})(M-M_{\mathcal{A}}+1)}R_{\tilde{\mathcal{A}}}^2\leq\frac{\epsilon_{\text{tol}}}{3}\iff M\geq M_{\mathcal{A}}+\frac{12L_1(j+4)R_{\tilde{\mathcal{A}}}^2}{\epsilon_{\text{tol}}\sqrt{K_1}(2-\sqrt{K_1})}-1. \tag{2.3.119}$$

Hence, we achieve ϵ_{tol} accuracy as long as: the noise is small enough; the eigenvalues of the inactive subspace and distance from x^* to $x^*_{\tilde{\mathcal{A}}}$ is small enough; and M is large enough. Details of those required bounds given by (2.3.117), (2.3.118), and (2.3.119), respectively. With these assumptions, we achieve ϵ_{tol} accuracy in

$$M\sim\mathcal{O}\left(\max\left\{L(P),\,\frac{L_1jR_{\tilde{\mathcal{A}}}^2}{\epsilon_{\text{tol}}\sqrt{K_1}(2-\sqrt{K_1})}\right\}\right). \tag{2.3.120}$$

This analysis also shows that given a particular variance in the noise, σ, as well as the term involving the eigenvalues of the inactive subspace and distance from x^* to $x^*_{\tilde{\mathcal{A}}}$, the achievable accuracy can be no better (i.e., less) than the value given below:

$$\epsilon_{\text{tol}}\geq\max\left\{\frac{12(j+4)C_5}{\sqrt{K_1}(2-\sqrt{K_1})}\sigma,\,3R_{\tilde{\mathcal{I}}}\left(\delta\sqrt{q_1+\cdots+q_j}+\sqrt{q_{j+1}+\cdots+q_P}\right)\right\}, \tag{2.3.121}$$

showing that the achievable accuracy will either be limited (mainly) by hyperparameter approximations or (mainly) by the error in $\tilde{\mathcal{A}}$.

We find similar complexity results to that of ASTARS, but also pay a price for approximations to $\tilde{\mathcal{A}}$, especially when the approximation is poor, usually do to both an insufficient number of samples and samples that do not explore Λ sufficiently. (Recall that since $\tilde{\mathcal{A}}$ is formed using information from a surrogate trained on STARS samples in the FAASTARS routine, it is really poor *surrogates* that hurt convergence.)

2.4 Numerical Results

In this section, we present various results obtained using STARS, ASTARS (with exact $\mathcal{A}$), and Fully-Automated ASTARS on several examples with convex, noisy, high-dimensional objective functions. Here, we only consider additive noise in $\hat{f}$, a case for which we presented theoretical results about the con-

vergence of our methods in the last section. (An example of STARS/ASTARS/FAASTARS with multiplicative noise is provided in the appendix of this chapter.)

In the first two examples, we present results from using both exact and inexact (i.e., learned) hyperparameters to perform STARS, ASTARS, and FAASTARS. In the third example, we only use exact hyperparameters; extensions to our methods are required to improve learning in this case, which we discuss in the next chapter. In the fourth example, we scale the associated L_1 value in the problem by various factors to determine how sensitive STARS is to changes in L_1 (which effect both stepsize and smoothing). In our fifth and final example, we perform FAASTARS with its usual learned $\tilde{\mathcal{A}}$, but fix the number of eigenvectors spanning $\tilde{\mathcal{A}}$ at various $\tilde{j}$ to demonstrate a phenomenon we call *flatlining*, where FAASTARS minimizes f in the $\tilde{\mathcal{A}}$ variables as much as possible, but cannot minimize f further, since it is restricted to the (wrong) $\tilde{j}$-dimensional active set.

2.4.1 STARS, ASTARS, and FAASTARS Convergence

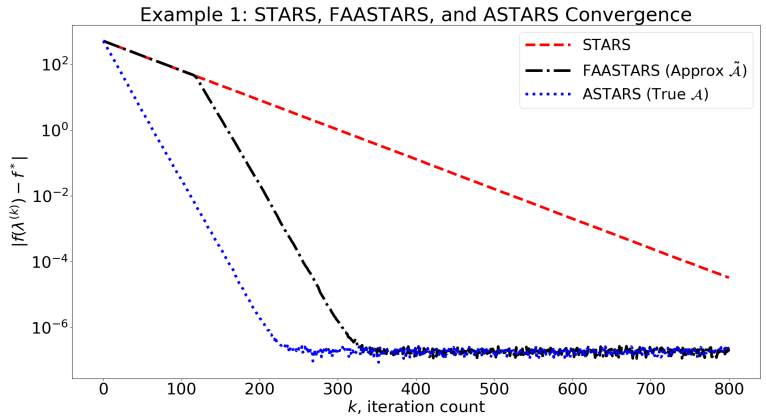
We begin in the first example by considering a case where extreme computational winnings are possible – in which $\mathcal{A}$ is one-dimensional, but Λ is 20-dimensional (i.e., $j = 1$ and $P = 20$). From our theoretical results, we know the convergence of all presented algorithms are dimension dependent, and so in this example, the benefits of stepping in $\mathcal{A}$ are as large as possible for this particular Λ (without, say, $\hat{f} \equiv 0$). We demonstrate the effectiveness of our methods both with the use of exact hyperparameters (using exact L_1 and σ^2) and inexact, learned hyperparameters. Note that throughout ASTARS is only performed in the case that we use exact hyperparameters.

Example 1: *Let $\hat{f} : \Lambda = \mathbb{R}^P \to \mathbb{R} = \mathcal{D}$. Fixing a weight vector, $w \in \Lambda$, where $w \neq 0$, we define*

$$\hat{f}(\lambda; \xi) := \left(w^\top \lambda\right)^2 + \epsilon(\xi), \tag{2.4.1}$$

$\epsilon(\cdot) \sim N(0, \sigma^2)$, where $\sigma^2 = 1 \times 10^{-12}$. We note the noise-free signal f is convex. The non-unique minimizers of $\hat{f}$ are given by $\{\lambda \in \Lambda : \lambda_1 + \cdots + \lambda_{10} = 1$, all with associated minimum $f^* = 0$. Here, the noise-free signal f has a one-dimensional active subspace (i.e., $j = 1$) in the direction w. Also note $L_1 = 2$. We considered $P = 20$ and note $D = 1$. We took $w_i = 1$ in all directions and an initial iterate $\lambda^{(0)}$ with components drawn from a zero-mean Gaussian with unit variance, scaled by 10. First, we performed 1000 trials each of STARS using exact hyperparameters, ASTARS using exact active hyperparemters as well as the exact active subspace, and using exact hyperparameters but learned active subspace, we performed FAASTARS. A maximum iteration count of $2P^2 = 800$ was used. Active subspaces were re-trained every $2P = 40$ steps for FAASTARS using an eigenvalue threshold of 99%. No noise regularization was

required. We then performed 500 trials each of STARS and FAASTARS using estimated hyperparameters and a maximum iteration count of 500*. Noise was increased to* $\sigma^2 = 1 \times 10^{-8}$*. In this case, active subspaces were relearned every* $P = 20$ *steps for FAASTARS using an eigenvalue threshold of 95% and a noise regularization of* σ^2 *was used.*



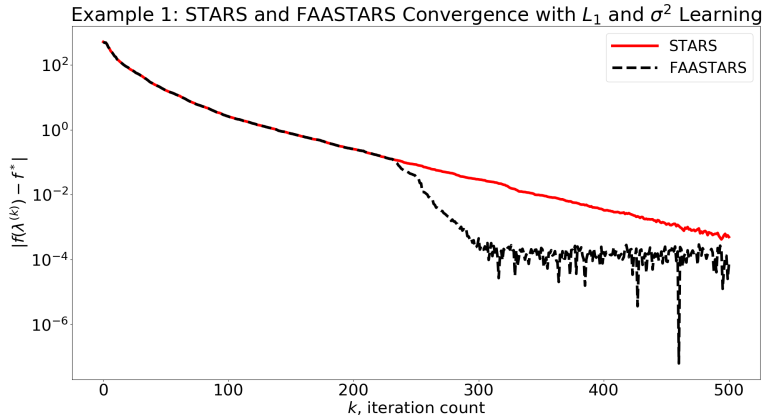


Figure 2.1: We show the average convergence of STARS, FAASTARS, and ASTARS with and without hyperparameter learning for Example 1.

In the first plot in 2.1 that ASTARS converges – i.e., reaches the level of the observable noise in the absolute difference between f^* and $f(\lambda^{(k)})$ – in roughly 200 iterations, whereas STARS has still not reached the level of the noise after 800 iterates, showing the clear computational benefit of stepping in the true $\mathcal{A}$ in this contrived example. For the price of roughly 150 iterations, FAASTARS is able to learn $\tilde{\mathcal{A}}$ from its "STARS burn-in phase," then produce a convergence rate identical to that of ASTARS, once it begins stepping in $\tilde{\mathcal{A}}$, which in this case, was well-approximated in most trials. FAASTARS requires only about 100 more iterations than ASTARS to converge, still easily beating STARS in full variables.

In the second plot, in which STARS and FAASTARS used learned hyperparameters, observe that the FAASTARS method converges in about 300 iterations, similar to the result with exact hyperparameters, demonstrating the effectiveness of our hyperparameters learning routines in this example. Interesting, STARS is greatly accelerated by the use of learned hyperparameters. This is due to underestimations of L_1, allowing larger descent steps in STARS, and in some ways mimicking the effect of dimension reduction. (We should note that throughout, we often find σ^2 to be excellently approximated by ECNoise; estimating L_1 is a bigger challenge, though.) In detail, since STARS will have less pessimistic (i.e., larger) hyperparamters, it can (correctly) smooth and scale steps as if f were defined in a smaller dimensional space. However, STARS still takes steps in Λ, and so it cannot achieve as fast a convergence as FAASTARS here, despite some benefits from underestimations to L_1.

In the next example, we consider an example with another 20-dimensional parameter space, but with a 10-dimensional $\mathcal{A}$, so that half of the variables are active. We call this function the *active sphere* function, since it is the sphere function defined only in j variables, which are active by consequence. In this scenario, we will find that STARS convergence will be greatly *hindered* in the case of learned hyperparameters, unlike what we just saw above, where learning *improved* STARS. In this case, we find L_1 is overestimated, mainly due to the larger dimensional AS and slightly higher level of noise.

Example 2: *Let* $\hat{f} : \Lambda = \mathbb{R}^P \to \mathbb{R} = \mathcal{D}$*,*

$$\hat{f}(\lambda; \xi) := \sum_{i=1}^{j} \lambda_i^2 + \epsilon(\xi), \tag{2.4.2}$$

$\epsilon(\cdot) \sim N(0, \sigma^2)$*, where* $\sigma^2 = 1 \times 10^{-3}$*. We note the noise-free signal* f *is convex. The minimizer of* $\hat{f}$ *is given by* $0 \in \mathcal{A}$ *with arbitrary components in* $\mathcal{I}$ *with minimum* $f^* = 0$*. Here,* $\hat{f}$ *has a* j*-dimensional AS spanned by the first* j *standard basis vectors in* Λ*. Also note* $L_1 = 2$*. We considered* $P = 20$*,* $j = 10$*, and note* $D = 1$*. We took initial iterate* $\lambda^{(0)}$ *with components drawn from a zero-mean Gaussian with unit variance, scaled by a factor of 10. First, we performed 100 trails each of STARS using exact hyperparameters, ASTARS using exact active hyperparemters as well as the exact AS, and using exact hyperparameters but learned AS, we performed FAASTARS. A maximum iteration count of* $2P^2 = 800$ *was used. Active subspaces were re-trained every* $P = 20$ *steps using an eigenvalue threshold of 99.9%. Noise regularization of* σ^2 *improved results. We then performed 250 trials of STARS and FAASTARS using estimated hyperparameters and a maximum iteration count of* $4P^2 = 1600$ *In this case, active subspaces were relearned every* $P = 20$ *steps for FAASTARS using an eigenvalue threshold of 99.95% and a noise regularization of* σ^2 *was used.*

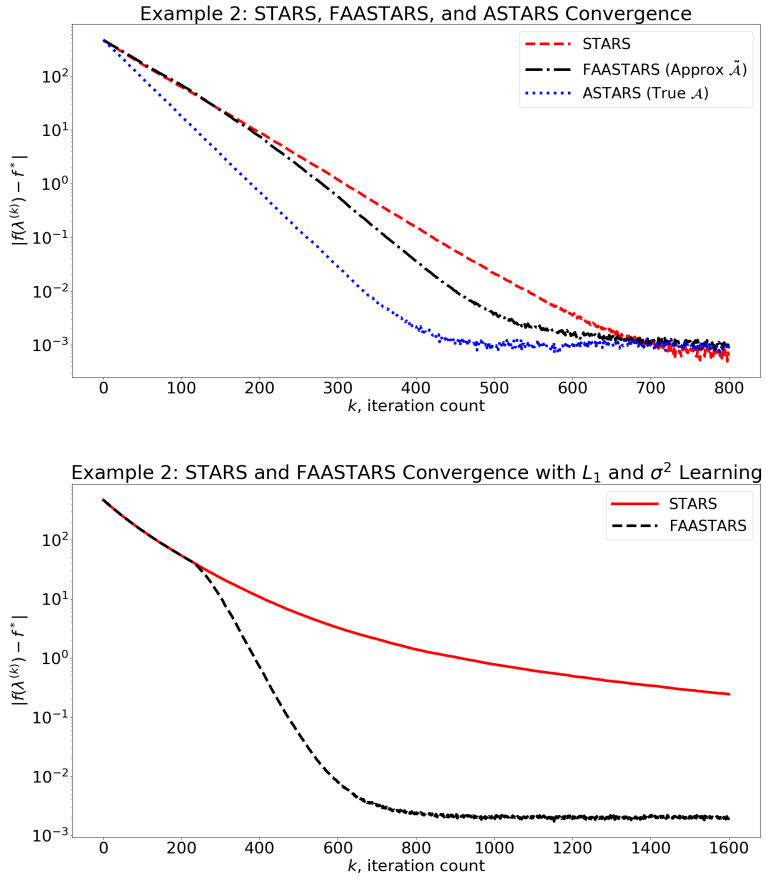


Figure 2.2: We show the average convergence of STARS, FAASTARS, and ASTARS with and without hyperparameter learning for Example 2.

In a fashion practically identical to Example 1, we see in 2.2 the clear computational benefit of ASTARS, and FAASTARS, which again achieves an identical convergence rate to ASTARS on average, after its upfront learning costs, on average. We find FAASTARS is able to reach the level of the noise using the same iterations even in the case of learning hyperparameters. However, as we discussed above, due to the noise and larger j, L_1 is overestimated in this case, so that STARS suffers from a convergence rate appearing twice as slow, on average, as STARS with exact hyperparameters.

In the next example, we consider a higher-dimensional Λ, with $P = 50$. We modified a test function appearing in Nesterov and Spokoiny (2017); Gorbunov *et al.* (2019) so that not all variables are active. In this higher-dimensional space, learning hyperparameters is more challenging, and extensions to our methods, such as more sophisticated methods for updating $\hat{L_1}$, are needed and discussed in the next chapter.

Example 3: *Let* $\hat{f} : \Lambda = \mathbb{R}^P \to \mathbb{R} = \mathcal{D}$,

$$\hat{f}(\lambda;\xi) = \frac{1}{2}\left(\lambda_1^2 + \sum_{i=1}^{j-1}(\lambda_i - \lambda_{i+1})^2 + \lambda_j^2\right) - \lambda_1 + \epsilon(\xi), \tag{2.4.3}$$

$\epsilon(\cdot) \sim N(0,\sigma^2)$, *where* $\sigma^2 = 1 \times 10^{-4}$. $\hat{f}$ *possesses additive noise with variance* σ^2. *This function is a test function used in Nesterov and Spokoiny (2017) we have modified so that there is a distinct AS. We considered* $P = 50$ *and* $j = 5$, *note* $D = 1$, *and the non-stochastic* f *is convex. Here, we have a* j*-dimensional* $\mathcal{A}$ *spanned by the standard basis vectors* e^i, $i = 1,\ldots,j$. *Note that the minimizer of* f, λ^* *is given by* $\lambda_i^* = 1 - i/(j+1)$ *for* $i = 1,\ldots,j$ *and* λ_i^* *is arbitrary for* $i = j+1,\ldots,P$ *and has minimum value* $f^* = -1/2(1-1/(j+1))$ *Nesterov and Spokoiny (2017). Also,* $L_1 = 4$ *Nesterov and Spokoiny (2017). We performed 50 trials each of STARS using exact hyperparameters, ASTARS using exact active hyperparemters as well as the exact AS, and using exact hyperparameters but learned AS, we performed FAASTARS. A maximum iteration count of* $3P^2 = 7500$ *was used. Active subspaces were relearned every* $2P = 100$ *steps using an eigenvalue threshold of 99.9%. Noise regularization of* σ^2 *improved results. We do not present hyperparameter learning in this example – further extensions to our methods are needed and discussed in the next chapter.*

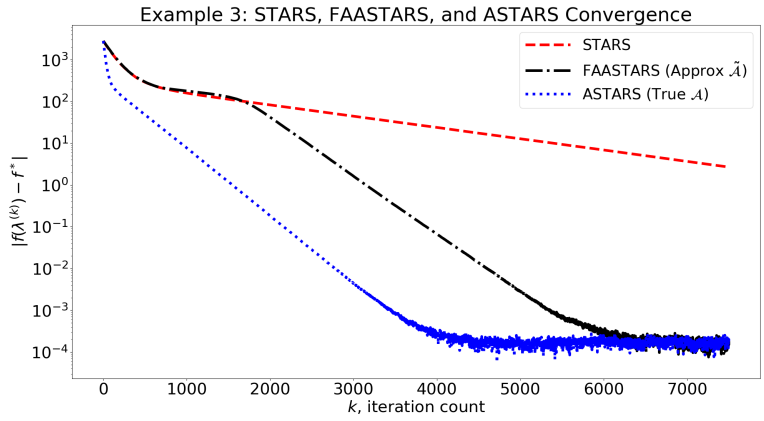


Figure 2.3: We show the average convergence of STARS, FAASTARS, and ASTARS without hyperparameter learning for Example 3.

Once again, ASTARS and FAASTARS offer computational savings, and FAASTARS converges at the rate of ASTARS after its learning phase, which is more expensive in 50-dimensional space, requiring upwards of 1,300 samples, shown in 2.3. STARS does not converge within the number of iterations we

have plotted, and requiring tens of thousands of iterates to reach the level of noise in the difference between $f(\lambda^{(k)})$ and f^*.

In the next example, we investigated the sensitivity of STARS to the value of L_1, which is used to form the algorithm's stepsize and smoothing hyperparameters. We used cL_1 in place of L_1, and took $c =$ 0.1, 0.2, and 4. We also used $c = 1$ – to obtain the correct value for L_1 – for comparing optimal STARS to the sub-optimal versions with incorrect L_1 used. Note, in this example, $\mathcal{A} = \Lambda$, thus we do not investigate the winnings of dimension reduction. Finally, note that we used the correct value for the variance in the noise, σ^2.

Example 4: *Let* $\hat{f} : \Lambda = \mathbb{R}^P \to \mathbb{R} = \mathcal{D}$,

$$\hat{f}(\lambda; \xi) := \sum_{i=1}^{P} \omega_i \lambda_i^2 + \epsilon(\xi), \tag{2.4.4}$$

$\epsilon(\cdot) \sim N(0, \sigma^2)$*, where* $\sigma^2 = 1 \times 10^{-5}$*. We took* $P = 10$ *and* $\omega_i = 1$*,* $i = 1, \ldots, P$ *so* $\hat{f}$ *the sphere function (defined in all variables in* $\mathbb{R}^P$*) with additive noise. The minimizer is* $0 \in \Lambda$ *with minimum* $f^* = 0$*. Note* $L_1 = 2$ *here. We examined the extent to which the presented theoretical bounds for* L_1 *estimates hold in practice. Recall we defined* $K_1 > 0$ *as the scale factor for which* $L_1^2 = K_1 \hat{L_1}^2$*. In STARS Theorem 4.5, we found* $0 < K_1 < 4$*. We write* $\hat{L_1} = c\, L_1$*, where* $c = 1/\sqrt{K_1}$ *and we have* $\frac{1}{2} < c < \infty$*. We used STARS to minimize* (2.4.4) *where we used the correct* σ^2 *(not estimated) and fixed* $\hat{L_1} = c\, L_1$ *for* $c = 0.1$, 0.2, 1, *and* 4*. We performed 100 trials for each value c and a maximum iteration count of* $2P^3 = 2000$*. Note that we allow* $c < 1/2$*; in some cases we find c may be less than* $1/2$ *and STARS with estimated hyperparameters will still converge.*

In 2.4, we see that the STARS method reaches the minimizer (to the level of noise) in around 500 iteartions. Notice that by scaling L_1 by a factor of 0.2, STARS converges rapidly, but is not able to reach the true level of the noise. This is due to the fact that underestimating L_1 leads to larger steps, and in this case, STARS cannot reach the level of the noise since it unable to take small enough steps to avoid perturbing f in the process as well. Our theoretical results tell us that any $c < 1/2$ risks STARS divergence; however, on average, we found that we can allow for significant underestimation of L_1 and avoid STARS blowup.

For any test problem, though, there will exist a cutoff at which cL_1 becomes too small for STARS to converge. Recall that since the smoothing and stepsize hyperparameters are inversely proportional to L_1, they grow as L_1 shrinks. STARS will diverge for L_1 too small because the hyperparameters will be

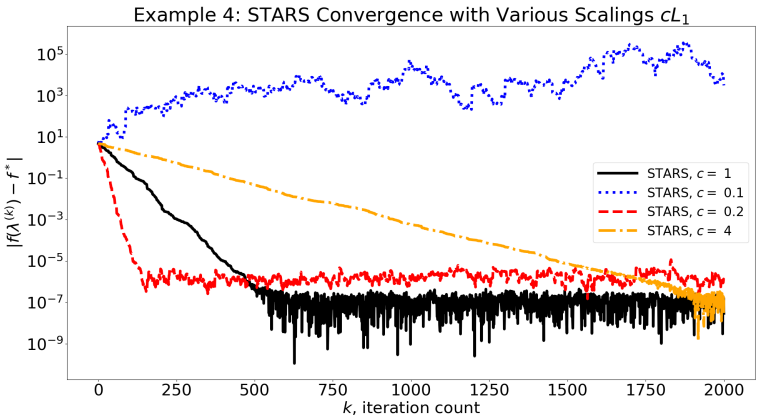


Figure 2.4: We show the average convergence of STARS for various $c\,L_1$ for Example 4.

too large, and in a sense, overly-optimistic about how large of a step is permissible, causing perturbations inflating f rapidly.

When $c > 1$ (but not extremely large) we observe steady but slow convergence of STARS. When L_1 is overestimated, STARS takes pessimistically small steps to avoid the exact blowup issue we reproduced by taking $c < 1$ (small enough). As $c \to \infty$, STARS will flat line, since it will be able to perturb f enough to either minimize *or* maximize the function.

In the next example, we demonstrate the effects of using a fixed number of active variables to construct $\tilde{\mathcal{A}}$. We find that while some fixed dimensions cause flat-lining (and converge more slowly than STARS), others may cause improvement (versus STARS), and still others may mimic STARS convergence.

Example 5: *Let* $\hat{f} : \Lambda = \mathbb{R}^P \to \mathbb{R} = \mathcal{D}$,

$$\hat{f}(\lambda;\xi) = \sum_{i=1}^{P} 2^{(-1)^{(i-1)}(i-1)}\lambda_i^2 + \epsilon(\xi), \tag{2.4.5}$$

$\epsilon(\cdot) \sim N(0,\sigma^2)$, *where* $\sigma^2 = 1 \times 10^{-3}$. *We note* f *is convex. We considered* $P = 10$ *and note* $D = 1$. *Note that the minimizer of* $\hat{f}$ *is given by* $0 \in \Lambda$ *with* $f^* = 0$. *Here, as* i *increases, terms in* $\hat{f}$ *become either more important or less important, depending on whether* i *is even or odd. We take an initial iterate similar to prior examples. Also note* $L_1 = 2^{P+1}$ *as long as* P *is odd; if* P *is even then* $L_1 = 2^P$. *Thus, with* $P = 10$, *we have* $L_1 = 1024$. *Here, the determination of which variables are active depends completely on one's choice of eigenvalue threshold, which determines the number of directions to include in* $\tilde{\mathcal{A}}$. *We show convergence of STARS compared with FAASTARS using fixed active dimensions* $\tilde{j} = 2$, 4, *and* 8

in turn, mimicking the effect of the use of various fixed eigenvalue thresholds. We performed 25 trials for each method and used a maximum iteration count of $5P^3 = 5000$. *Exact hyperparameters were used.*

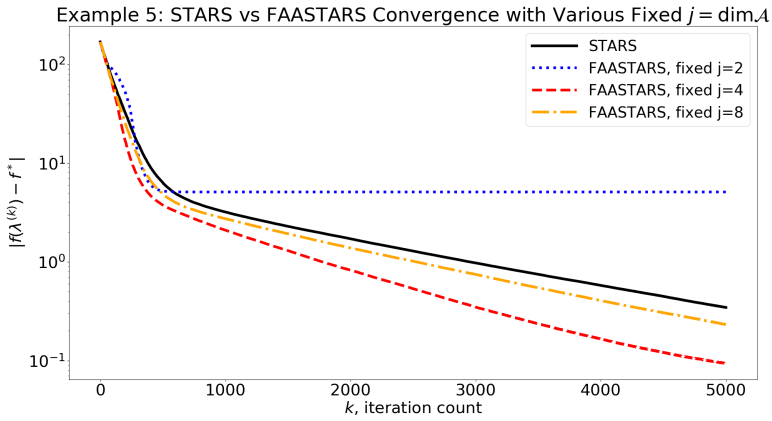


Figure 2.5: We show the average convergence of STARS versus FAASTARS for various fixed $\tilde{j}$ for Example 5.

The AS in this example is less clear than the previous examples, since the spectrum of the sensitivity matrix W does not exhibit as large of a drop-off between a pair of eigenvalues $q_{\tilde{j}}, q_{\tilde{j}+1}$ as the others. To capture the computational savings in particular active subspaces, we instead used the fixed $\tilde{j}$ number of learned active directions in every step, making the choice of τ irrelevant.

In 2.5, we find neither STARS nor FAASTARS (with any fixed dimension) converges in 5,000 iterations or less. We see that for fixed $j = 2$, FAASTARS flat lines quickly, after only 500 or so iteartions, and will not converge. This is because while the first two active variables were minimized, nontrivial variables (of order 1) are left untouched in the approximate ASTARS phase, and have not been fully minimized. For fixed $j = 4$, we find that FAASTARS minimized most of the variables with larger magnitudes, and outpaces STARS. However, with more iterations, there is a danger of flat-lining (before reaching the noise) as soon as f is minimized in the 4 used active directions. As $j \to 10$, we recover the convergence of STARS, as we see when $j = 8$, offering mild computational savings compared to the full 10-dimensional space. (We did not run the trials until achievable accuracy was reached due its great computational expense.)

In our examples, we saw the computational benefit in stepping in the AS instead of in the full variables. Recall that the hyperparameters in STARS are dimension-dependent, so anytime an AS resolved $\hat{f}$ well – which occurred in the above examples by obvious design – we expected ASTARS in the exact active variables to converge more quickly than STARS in full variables. We observed that ASTARS using learned active directions may actually fail to converge at all if too few training samples are used.

The minimizers obtained in Examples 1, 2, 3, and 5 are non-unique. Regularization of the objective function is required to obtain unique solutions in these examples. As we shall see in Chapter IV, in practice, regularization often appears naturally in $\hat{f}$ due to assumptions about uncertainty in Λ. For now, we are satisfied with ASTARS and FAASTARS, which offer computational improvements to obtain both unique and non-unique minimizers to OUU problems in our setting.

2.4.2 Comparisons to Surrogate-Based Approach

For Examples 1, 2, and 3, we noted the iteration at which FAASTARS converges (i.e., reaches the level of the noise in our plots). Denoting the iterate at which FAASTARS convergence was (roughly) achieved with M^*, we had $M^* \approx 350$, 650, and 6500 in Examples 1, 2, and 3, respectively (in the case of exact hyperparemters). Then, recalling that for a single iteration, FAASTARS takes two evaluations of $\hat{f}$, we trained surrogates using $2M^*$ random samples in Λ. In detail, using $2M^*$ random samples $\lambda^i \in \Lambda$ drawn from $N(0, cI_P)$, we fit a quadratic surrogate using λ^i and $f(\lambda^i)$, $i = 1, \ldots, 2M^*$. We then minimized the obtained quadratic surrogate using standard gradient-based methods (i.e., Python's built-in gradient-based optimizer). We then compare f^* with $\hat{f}(\lambda_F^*)$ for an obtained minimizer λ_F^* of F. (We also consider the value of $F(\lambda_F^*)$, which is indicates the extent to which the surrogate was minimized.)

Using the surrogate-based method for the objective functions in all three examples, we found corresponding minimizers almost identical to those obtained via FAASTARS (in their active variables) and their corresponding minima are at the level of the noise. However, we find that the quality of the obtained minimizer via the surrogate-based method is dependent on the samples used to form the surrogate. Here, since the minimizers are either exact $0 \in \Lambda$ – or not very far away from that point – our zero-mean Gaussian samples cover the region near the minimizer.

We also considered uniform samples in hypercubes within Λ to form surrogates; e.g., $\lambda \sim U([-1, 1]^P)$. Again – if and when a minimizer λ^* of f lies inside the support of our samples, the surrogate method is quite accurate and efficient. Typically the minimum number of samples to form a quadratic surrogate yields a high quality solution for the normal and uniform draws used. If, for instance, λ^* is outside $[-1, 1]^P \in \Lambda$ (by shifting any of our examples), our solution quality is poor.

First, note that using a bounded distribution to draw random points within Λ writes a *constrained* optimization problem. In this case, we employed the standard gradient-based method of *Sequential Least Squares Quadratic Programming (SLSQP)* to solve the constrained problem when we used bounded samples. Drawing random $\lambda \in \Lambda$ from distributions with support equal to Λ avoids constraints in our problem. However, any sampling method that does not take draws near λ^* will produce a surrogate that must be extrapolated into the region around λ^*, often producing an unreliable surrogate in this region and the obtained minimizer is usually quite inaccurate. An advantage of ASTARS-based methods is that we do not

need to worry about either: (1.) knowledge about the location of $\lambda^* \in \Lambda$; nor (2.) constraints caused by sampling distributions – ours have a support of Λ.

2.4.3 Notes on Used Software and Hardware

A Python 3.7 package called ASTARS was used to produce results in this section, and is open-source and publicly-available online Hall and Carey (2020). The ASTARS package has the functionality to perform STARS, ASTARS, and FAASTARS. STARS is not otherwise publicly-available, to our knowledge. The algorithms used here which are open-source and publicly-available online include the AS software of Constantine (2020) – updated in Python 3.6 (required for ASTARS package) by Varis Carey (2020) – and ECNoise by Moré and Wild (2020).

Computations here were performed on a Dell laptop running Ubuntu 20.04 LTS with an i7 Intel processor and 32 GB of RAM. Most simulations (i.e., series of trials) used to create the plots in this chapter took less than an hour to run – indeed, some only take a minute or two (e.g., Example 1, which had 1000 trials). In Example 3, since we considered a 50-dimensional parameter space, the simulation took much longer (several hours). Results could undoubtedly be obtained more quickly using a computational server; on the other hand, our methods would take longer on a machine with less processing power than the one used here.

2.5 Discussion

We presented combinations and modifications made to well-documented algorithms including STARS Chen and Wild (2015), Monte Carlo AS learning Constantine *et al.* (2015), noise variance learning Moré and Wild (2015), and Lipschitz constant learning Calliess (2017) to produce the fully-automated ASTARS algorithm. In addition, we presented several model problems that were used for testing ASTARS and FAASTARS.

There is not any guarantee that a general stochastic-free mapping $f : \Lambda \to \mathcal{D}$ permits dimension reduction via AS. AS methods may fail to improve STARS for many reasons – sometimes occurring in combination with each other – including (nearly) equal importance of modeled parameters, a ∇f that is poorly approximated by surrogates, or too much noise. Regardless, in the case that no AS is clearly defined, recall that ASTARS is equivalent to STARS in full variables.

If an AS exists, we observed that performing ASTARS and FAASTARS provided computational savings in our numerical examples when compared to STARS, since its convergence outpaces STARS convergence on average; see our technical report for detailed theoretical statements. We note that at times, any of the presented algorithms may take a very *lucky* step in Λ, causing quick convergence. After all, the considered DFO algorithms depend on random (but smoothed) steps.

We also observed that it is possible to learn the AS of the functions considered using the samples obtained from deterministic ECNoise and STARS iterates, which greatly reduces the usual computational expense of an AS computation. This result is not obvious, since AS discovery typically relies on (many) random, iid samples in Λ.

Sometimes, when FAASTARS has minimized $\hat{f}$ in $\tilde{\mathcal{A}}$ as much as possible, there may be remaining variables in the inactive subspace that are not minimized, as in Example 5 above. Indeed, we saw that for various fixed AS dimensions $\tilde{j}$, ASTARS may behave almost identically to STARS, or much worse than STARS, depending on $\tilde{j}$. Even without a fixed AS dimension, we still find that if FAASTARS determines $\tilde{j} < j$ (or $\tilde{j} > j$), iterates may not provide enough information for FAASTARS to update $\tilde{j}$ closer to j, incorrectly fixing $\tilde{j}$, and causing behavior identical to the flat-lining we produced in Example 5.

At its core, this problem is directly related to the choice of eigenvalue threshold τ, which determines how many directions to include in $\tilde{\mathcal{A}}$. When τ is fixed and samples are not informative enough to increase (or decrease) $\tilde{j}$, $\tilde{\mathcal{A}}$ cannot be substantially updated based on local information.

In the next chapter, we address the flat-lining behavior witnessed in some numerical examples by introducing a method of *adaptive thresholding*, which will change τ if and when flat-lining occurs. Other extensions of the ASTARS method, such as alternative weighting schemes and other approaches to address flat-lining, may also improve convergence and behavior, and are considered in the next chapter, as well.

In Appendix A, we present a few brief technical results related to certain (average) observed $\tilde{j}$ for our learned $\tilde{\mathcal{A}}$, showing the effect of AS retraining, which steadily reduces $\tilde{j}$ over the course of a FAASTARS trial. We also provide brief considerations for multiplicative noise – while we do not present theoretical results for the algorithms in this setting, we implemented the necessary modifications to STARS, ASTARS, and FAASTARS in order to produce numerical results in the presence of multiplicative noise. (We present one of those results.)

In their unpublished manuscript, the authors in Gorbunov *et al.* (2019) present an accelerated version of DFO algorithms of Nesterov's classic approach called RS_μ in Nesterov and Spokoiny (2017) (which is similar to STARS in Chen and Wild (2015)) leveraging techniques of *mirror descent*. As authors in Gorbunov *et al.* (2019) observe, complexity in M – the maximum number of iterations – is equivalent to the complexity of oracle calls (i.e., the number of times we must approximate certain directional derivatives). These complexities are then proportional to the number of $\hat{f}$ evaluations, as well. Since we postulate that $\hat{f}$ calls are expensive, we always seek methods that evaluate our map as few times as possible. We will note, though, that we also know that there is a tradeoff between fewer evaluations (samples) and the quality of the numerically estimated AS.

For a more concise comparison between the complexity results here and in Chen and Wild (2015); Nesterov and Spokoiny (2017); Gorbunov *et al.* (2019), we let $\epsilon = \epsilon_{\text{tol}}$ and recall $R^2 = ||\lambda^{(0)} - \lambda^*||_2^2$ denotes the distance from the true minimizer to our initial iterate. Recall, for all STARS-based methods, we achieve the main complexity statement in (2.3.74) so long as

$$\sigma \leq \frac{\sqrt{2K_2}(2-\sqrt{K_1})\epsilon_{\text{tol}}}{8(P+4)C_4}, \tag{2.5.1}$$

If we satisfy (2.5.1), then we obtain the results shown in Table 2.1 below. Other methods will have σ^2 appearing in their complexity results, since no bounds are assumed directly on the variance in the noise.

Table 2.1: Complexity of DFO Algorithms

DFO Algorithm	**Complexity**, $\mathcal{O}(\cdot)$
STARS, Chen and Wild (2015)	$\frac{L_1 P R^2}{\epsilon}$
RS_μ, Nesterov and Spokoiny (2017)	$\frac{L_1 P R^2}{\epsilon}$
STARS (estimated hyperparameters)	$\frac{L_1 P R^2}{\sqrt{K_1}(2-\sqrt{K_1})\epsilon}$
ASTARS (estimated hyperparameters)	$\frac{L_1 j R^2}{\sqrt{K_1}(2-\sqrt{K_1})\epsilon}$
FAASTARS (estimated hyperparameters)	$\frac{L_1 \tilde{j} R^2}{\sqrt{K_1}(2-\sqrt{K_1})\epsilon}$
ARDFDS, Gorbunov *et al.* (2019)	$\max\left\{P^{\frac{1}{2}+\frac{1}{q}}\sqrt{\frac{L_1 R^2}{\epsilon}}, \frac{P^{\frac{2}{q}}\sigma^2 R^2}{\epsilon^2}\right\}$

The methods in Gorbunov *et al.* (2019) achieve accelerated convergence by utilizing a different random direction in forming their gradient oracle, drawn randomly from the unit hypersphere in Λ in a mirror descent scheme. For future work, we propose investigating whether dimension reduction could accelerate the methods in Gorbunov *et al.* (2019).

As a final note of caution, anytime the variance of the noise σ^2 approaches the magnitude of f values, we expect failure of all methods presented. The assumptions we made in Chapter 1 – assumptions ubiquitous in the DFO literature – forbid noise of this order, and with good reason: anytime the noise is on the order of function evaluations, it becomes difficult (and eventually impossible) to distinguish the true signal from the effects of noise. Filtering or smoothing methods must be used in this scenario, which is outside the scope and focus of this dissertation Nesterov (2005).

CHAPTER III

ENHANCEMENTS AND EXTENSIONS TO ASTARS-BASED METHODS

3.1 Overview

In this chapter, we consider enhancing ASTARS-based methods using more advanced strategies for: (1.) methods for training and using $\mathcal{A}$ within the FAASTARS routine; and (2.) weighting schemes within the FAASTARS for steps in $\mathcal{A}$. We also propose extending the ASTARS/FAASTARS methods to the case of minimizing vector-valued functions, where $\dim \mathcal{D} = D > 1$.

We motivate this chapter by recalling a major limitation we saw demonstrated in the previous chapter's Methods section with ASTARS-based methods as presented so far: we witness an eventual flat-lining of f values in ASTARS (i.e., ASTARS steps are not minimizing f) when the algorithm is basing its active steps on incorrect AS information from a poor approximation $\tilde{A}$ for $\mathcal{A}$. Even with the use of active subspace retraining, we still observe flat-lining in certain examples. Poor approximations arise mainly due to a large number of dimensions P of Λ, poor gradients from badly-approximated surrogates used in training $\tilde{\mathcal{A}}$, and limited deterministic samples, many of which are also close to one another in parameter space.

To remedy poor approximations to the true $\mathcal{A}$, we consider methods to the supplement (or replace) active subspace retraining we introduced in the previous chapter to form $\tilde{A}$ for active steps in the FAASTARS routine. We present methods we call *adaptive thresholding* and *active subcycling*, which are both meant to address flat-lining behaviors in FAASTARS in some cases. In adaptive thresholding, the eigenvalue threshold τ (which controls the number of active variables included in $\tilde{\mathcal{A}}$) is updated adaptively when slow convergence is detected, often caused by a choice τ which is too small or large. In active subcylcing, when poor convergence is observed, we switch back to STARS in full variables, stopping only if $\hat{f}$ evaluations once again flat-line – in which case we retrain $\tilde{\mathcal{A}}$ with our new samples of $\hat{f}$ and again switch to ASTARS, potentially cycling between the two steps until convergence is reached.

We then propose various weighting schemes within the pure ASTARS routine. Recall the standard weighting scheme in ASTARS in which steps are taken randomly in every active direction with unit variance. Alternative weighting schemes we present take even more advantage of the AS by weighting the most important directions in the AS the most. We use eigenvalue information to assign a weight to each active (and potentially even inactive) directions.

Finally, we present a method for performing STARS (as well as ASTARS/FAASTARS) in the case that $\hat{f}$ maps to $\mathcal{D}$ with $D > 1$, meaning $\hat{f}$ is a vector-valued function (with stochastic noise). We consider a standard approach which will transform a minimization problem involving a vector-valued objective function into a minimization problem involving a real-valued objective function. Then, we may apply

the STARS/ASTARS/FAASTARS methods with little hesitation, tending to only a handful of subtle but crucial details.

The methodologies for enhancing and extending ASTARS above are outlined in the Methods section. We then present numerical results, comparing un-modified FAASTARS with the various extended versions.

3.2 Methods

We will present the following extension methods within the context of Fully Automated ASTARS (FAASTARS), where we generally assume the user has a limited amount of information about f, beyond the ability to evaluate it at various points in Λ.

3.2.1 Adaptive Thresholding and Active Subcycling

Flat-lining of FAASTARS occurs at an iteration k whenever the slope between $\hat{f}(\lambda^{(k)})$ and $\hat{f}(\lambda^{(k-1)})$ approaches zero. To make this criterion more precise, we used theoretical results from Chapter II – particularly the last term in (2.3.70) – and some numerical experimentation to write the following *flat-lining criterion*, which is

$$m_k > -\frac{P\hat{\sigma}}{\sqrt{2}}, \tag{3.2.1}$$

where m_k denotes the k-th slope of the line between $\hat{f}(\lambda^{(k)})$ and any number of previous iterates (we typically use about 10). Anytime (3.2.1) is true, we have reason to believe FAASTARS is flat-lining (due to minimizing in an incorrect/incomplete $\tilde{\mathcal{A}}$ as much as possible) or has converged to a minimizer.

We first briefly discuss adaptive thresholding. First, we assume the user may set τ to any reasonable value at the outset of FAASTARS, especially without exact knowledge of f. (Usually $\tau \in [0.9, 0.95]$ is standard.) With adaptive thresholding turned on, (3.2.1) is checked at every iteration. When the flat-lining criterion is met, we increase $\tilde{j}$ by 1 dimension and set τ equal to sum of the first $\tilde{j}$ *normalized* eigenvalues. By consequence, τ will now exactly capture $\tilde{j}+1$ active directions, increasing the space iterates step in towards a minimizer. We found mild increases in τ when flat-lining occurs forces the inclusion of more active variables which have not yet been minimized so that FAASTARS can resume a proper convergence rate towards a minimizer.

We now present active subcycling. This method will involve extensions and modifications to the third phase of FAASTARS, which we recall involves stepping in the approximated AS $\tilde{A}$ (once enough STARS steps were taken in the second phase of FAASTARS). Recall that in active subspace retraining, we supplement the set of samples with information from additional ASTARS iterates and recompute $\tilde{A}$ at

a predetermined interval. However, without the ability to explore in directions outside $\tilde{\mathcal{A}}$, FAASTARS will stop learning new information about inactive variables in some cases, even with the use of retraining.

In Algorithm 7 below, we present active subcycling, which will involve stepping in $\tilde{\mathcal{A}}$ until (3.2.1) is met, at which point we switch to performing STARS for better learning and convergence. If possible, we may switch back to $\tilde{\mathcal{A}}$ if flat-lining occurs again, and continue this process iteratively until convergence of the algorithm.

Algorithm 7: Active Subcycling

Input: $\lambda^{(k)}$; $f_k := \hat{f}(\lambda^{(k)})$; history of evaluations $\hat{f}(\lambda^{(i)})$, $i = 1, \ldots, k$; $\tilde{\mathcal{A}}$; $\tilde{\mathcal{I}}$; P; $\hat{\sigma}$; line window $l_w < k$ (default $l = 10$)
Fit a line (via least squares) between f_k and $f_{k-1}, \ldots, f_{k-l_w}$
Compute slope of this line, m_k
while $k < M$ **do**
  if (3.2.1) *is True* **then**
    Perform step of STARS in full variables
    Compute m_k
    Set $k = k + 1$
  else
    Retrain $\tilde{\mathcal{A}}$ with new f_k and perform a step of FAASTARS (Phase 3)
    Set $k = k + 1$

We see that both adaptive thresholding and active subcycling may both be used to solve the same general flat-lining issue, but in distinct ways. While adaptive thresholding slowly increases $\tilde{\mathcal{A}}$ in hopes to "cover" the true $\mathcal{A}$, active subcycling immediately switches to searches in all variables using STARS as soon as flat-lining begins to occur. Subcycling also offers the benefit of reducing the search directions back to some active set $\tilde{\mathcal{A}}$ if flat-lining continues, whereas thresholding can only steadily increase the number of search directions. We now turn our attention to alternative weighting schemes, which improves (FA)ASTARS convergence in other settings.

3.2.2 Alternative Weighting Schemes

Recall that the direction of an active step from ASTARS takes the form

$$u_{\mathcal{A}}^{(k)} = \sum_{i=1}^{j} r_i \tilde{v}^i, \quad r_i \sim N(0, \omega_i^2), \quad i = 1, \ldots, j, \tag{3.2.2}$$

which is a weighted linear combination of the active directions of $\hat{f}$. In ASTARS, we equally weight the j active directions, in the sense that $\omega_i^2 = 1$ for $i = 1, \ldots, j$, and in STARS Chen and Wild (2015), all P directions are randomly perturbed with unit variance as well.

We may consider $\omega_i = \sqrt{\tilde{q_{max}}}/\sqrt{\tilde{q_i}}$, $i = 1, \ldots, j$, where we recall $\tilde{q_i}$ denotes a nunmerical eigenvalue obtained from the eigendecomposition of $\tilde{W}^\top \tilde{W}$, which much now be carefully indexed so that $\tilde{q_i}$

corresponds to the eigenvalue in the i-th direction in Λ (and not the i-th largest!) and $q_{\max}$ is the largest eigenvalue.

We see that these weights will ensure smaller random steps in the most active directions, and progressively larger steps in less active directions. As such, we may also consider using this weighting scheme in all P directions in (3.2.2), letting $i = 1, \ldots, P$ and defining ω_i for all $i = 1, \ldots, P$ similarly to above.

It may seem counter-intuitive to use small weights in $\mathcal{A}$ and larger weights in $\mathcal{I}$. However, as we saw in the last chapter, our methods are quite sensitive to L_1 estimates – shrinking the smoothing factor and producing small steps when we overestimate – and the surrogates we obtain to form $\tilde{\mathcal{A}}$, both of which are most heavily influenced by the most active directions. As such, we propose more exploration in $\mathcal{I}$ to learn slightly more optimistic (and accurate) L_1 approximations and surrogates of higher quality, with points spread further apart in $\mathcal{I}$, possibly aiding in more accurately discovering just how inactive those directions may (or may not) be.

Note we violate STARS assumptions as soon as we take $\omega_i^2 > 1$ for any direction i. If we take $\omega_i^2 < 1$ in certain directions, we will not violate the theoretical assumptions, but we will likely witness sub-optimal convergence when our estimated eigenvalues are incorrect. Since incorrect eigenvalues will lead to incorrect assumptions about how active some directions may be, we may not take cautious enough steps in those directions.

3.2.3 FAASTARS Multi-Objective Routine

We now consider the case that f is a vector-valued function, so $\dim \mathcal{D} = D > 1$. Formally, when $\Lambda = \mathbb{R}^P$ and $\mathcal{D} = \mathbb{R}^D$, $D > 1$, we write $f(\lambda) = (f_1(\lambda), \ldots, f_D(\lambda))^\top \in \mathcal{D}$, and assume each f_i, $i = 1, \ldots, D$ is a real-valued function $\Lambda \to \mathbb{R}$. Here, we will assume all f_i are differentiable and convex in their domain Λ and each have a gradient Lipschitz constant $L_{1,i}$.

To model additive noise in the f_i evaluations, we write $\hat{f}(\lambda; \xi) = (\hat{f}_1(\lambda; \xi_1), \ldots, \hat{f}_D(\lambda; \xi_D))^\top$, where $\hat{f}_i(\lambda; \xi_i) = f_i(\lambda) + \epsilon_i(\xi_i)$. Herein we set $\epsilon_i = \epsilon$ for all $i = 1, \ldots, D$, and assume ϵ is a zero-mean random variable with variance σ^2 as in Chapter 2. Note that despite the fact that we assume $\epsilon_i = \epsilon$, each $\epsilon_i(\xi_i) = \epsilon(\xi_i)$ is a distinct realization or draw of the random variable ϵ indexed by ξ_i, where ξ is now an indexing *vector*. That is, not all component-wise $\hat{f}_i$ experience the same noise draw – each are iid.

Minimizing vector-valued functions is a field of optimization known as *Multi-Objective Optimization*, an interesting and mathematically rich area in its own right. We shall consider one of the most standard approaches, which is the method of *linear scalarization* Karimi and Karimi (2017). Linear scalarization involves transforming a multi-objective problem into a single-objective problem, where the transformed objective function is a map to a 1-dimensional data space (here, $\mathcal{D} = \mathbb{R}$). Linear scalarization in-

volves forming the transformed objective function, which is taken as a weighted linear combination of the D component-wise functions in f.

We write the unconstrained, linearly scalarized optimization under additive uncertainty problem as

$$\min_{\lambda \in \Lambda} \mathbb{E}_{\xi} \left(\sum_{i=1}^{D} w_i \hat{f}_i(\lambda) \right) \tag{3.2.3}$$

where $w_i \in \mathbb{R}$, $i = 1, \ldots, D$ are weighting coefficients. Here, we will take $w_i = 1$ to equally weight each term in the scalarized objective function. Also, it is unclear what effects $w_i \neq 1$ may have on magnifying or shrinking the noise.

The non-unique solutions to (3.2.3) are known as *Pareto points* (or *properly efficient solutions* as in Karimi and Karimi (2017)) and lie on a contour of optimal points. Indeed, without regularization, solutions to (3.2.3) are not generally unique. Multi-objective optimization is not the focus of this dissertation, and so we will not consider regularization techniques used to ensure unique properly efficient solutions – this is an interesting and open computational challenge in our setting.

We do however propose and investigate a modification to our FAASTARS routine which will determine whether or not $\hat{f}$ is a vector-valued function. If $\hat{f}$ is determined to be vector-valued, then FAASTARS will automatically define the objective function in (3.2.3) (with the equal weights $w_i = 1$) and perform FAASTARS. We again emphasize taht the obtained properly efficient solution is generally non-unique.

The scalarized objective function we use, $\hat{J}(\lambda) := \sum_{i=1}^{D} \hat{f}_i(\lambda)$ (with all $w_i = 1$) has zero-mean additive noise. However, the variance in the additive noise is $D^2\sigma^2$ (due to the sum of the D observations of ϵ). Also, one can show that the stochastic-free J has a gradient Lipchitz constant equal to $L_{1,1} + \cdots + L_{1,D}$. Since FAASTARS has the ability to learn σ^2 and L_1 only from evaluations of $\hat{J}$, we only need the closed-form variance and Lipschitz constant for diagnostics and analysis.

In the following Numerical Results section, we investigate the effectiveness of what we call the *FAASTARS Multi-Objective Routine*, which, in summary, forms $\hat{J}$, performs FAASTARS, and exits by reporting the obtained minimizer (after M maximum iterations) and its evaluation under $\hat{f}$.

3.3 Numerical Results

We present numerical results as they relate to the proposed enhancements and extensions to ASTARS-based methods. We begin by presenting the FAASTARS Multi-Objective Routine and alternative weighting schemes. Then, we consider examples of active subcycling and adaptive thresholding to address flatlining, and also investigate blending the presented techniques.

Example 1 (FAASTARS Multi-Objective Routine): *Let* $\hat{f} : \Lambda = \mathbb{R}^{20} \to \mathcal{D} = \mathbb{R}^3$ where

$$\hat{f}(\lambda;\xi) := \left(\lambda_1^2 + \epsilon(\xi_1), \quad (\lambda_1 - 1)^2 + \epsilon(\xi_2), \quad \sum_{i=2}^{5} \lambda_i^2 + \epsilon(\xi_3)\right)^\top. \tag{3.3.1}$$

$\epsilon(\cdot) \sim N(0, \sigma^2)$, *where* $\sigma^2 = 1 \times 10^{-6}$ *and each* $\hat{f}_i$ *involves draws of the additive noise* ϵ. *Note* $L_1 = 6$ *here. We see* λ_1 *is the sole active variable for* $\hat{f}_1$ *and* $\hat{f}_2$, *and* $\lambda_2, \ldots, \lambda_5$ *are active variables for* $\hat{f}_3$; *the remaining 15 directions are inactive for all* $\hat{f}_i$. *We took initial iterate* $\lambda^{(0)}$ *with components drawn from a zero-mean Gaussian with unit variance, scaled by 10. We performed 1000 trails each of STARS using exact hyperparameters, ASTARS using exact active hyperparemters as well as the exact AS, and using exact hyperparameters but learned AS, we performed FAASTARS, shown in 3.1. A maximum iteration count of* 500 *was used. Active subspaces were re-trained every* $2P = 40$ *steps using an eigenvalue threshold of 99.9%. Noise regularization of* $D^2\sigma^2$ *was used to moderately improve results (i.e., so FAASTARS can reach the level of the noise).*

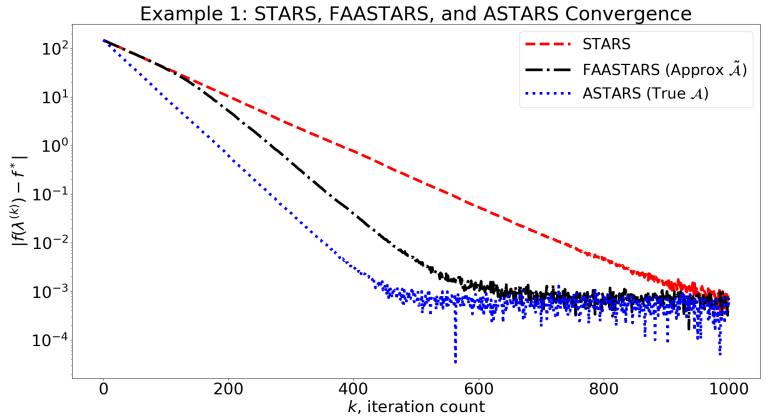


Figure 3.1: We show the convergence of STARS, FAASTARS, and ASTARS (without learning) for the multi-objective problem in Example 1.

Similarly to the single-objective functions we considered in Chapter II, we found ASTARS and FAASTARS offer computational savings when $\mathcal{A}$ is clearly defined. (The result is unsurprising since our multi-objective routine uses linear scalarization to convert the multi-object problem to a single-object problem.) The obtained minimizer is non-unique; regularization in $\mathcal{I}$ would be required to ensure a unique solution.

In the next example, we revisit the active sphere function, modified with coefficients so that certain active variables can be made more or less active. (Doing so allows us to demonstrate the effectiveness

of the weights on active variables of ranging importance.) We perform FAASTARS with regular weighting as well as FAASTARS with alternative weights and compare the results.

Example 2 (Alternative Weights): *We modify the active sphere in Chapter II, Example 2 to include coefficients* w_i *so* $\hat{f} : \Lambda = \mathbb{R}^{10} \to \mathcal{D} = \mathbb{R}$ *where*

$$\hat{f}(\lambda; \xi) := \sum_{i=1}^{j} w_i \lambda_i^2 + \epsilon(\xi), \tag{3.3.2}$$

$\epsilon(\cdot) \sim N(0, \sigma^2)$*, where* $\sigma^2 = 1 \times 10^{-6}$ *and each* $\hat{f}$ *involves draws of the additive noise* ϵ*. The coefficients* w_i *can make the* j *variables appearing in* $\hat{f}$ *either more active, or entirely inactive. We took* $j = 3$ *and* $w_1 = 100$*,* $w_2 = 10$*, and* $w_3 = 1$ *so* $\mathcal{A}$ *is 3-dimensional, but* λ^1 *is the most active variable. (*λ^2 *and* λ^3 *are both active as well.) We used standard FAASTARS (with normal weighting) and FAASTARS with the alternative weighting scheme presented here. We used* $\tau = 0.999$ *for FAASTARS with the default weights, but found* $\tau = 0.99999$ *was better for FAASTARS with alternative weights, likely since large exploration in* $\mathcal{A}$ *is slightly damped by our weighting. 100 trials of each method were performed and averaged to produce the figure below and mildly regularizing against noise in our surrogates slightly improved convergence around the noise level.*

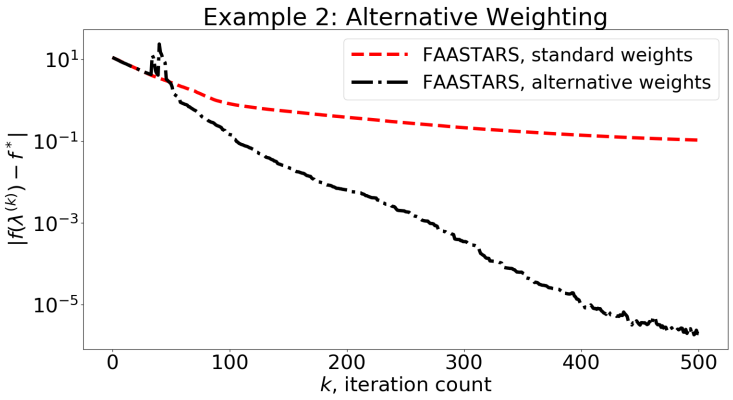


Figure 3.2: We show the convergence of FAASTARS with default weights versus FAASTARS with alternative weights in Example 2.

In 3.2, we see after a brief series of unstable iterations – directly after the AS has been trained, and active directions are less reliable – the alternatively-weighted FAASTARS offers large computational savings compared with regular FAASTARS, in this example, on average. Often, when the order of magnitude of $\hat{f}(\lambda^0)$ is small, the AS is difficult to accurately estimate, and active directions are inaccurate. Since

the weights dampen steps in $\mathcal{A}$ but allow large $\mathcal{I}$ exploration, $\tilde{\mathcal{A}}$ is better approximated, and computational savings from stepping in a more accurate $\tilde{\mathcal{A}}$ are borne out, as expected.

In the next example, we revisit Example 5 from Chapter II, in which flat-lining is a serious issue when $\tilde{j}$ is too small. We perform both active subcycling and adaptive thresholding to more obtain active sets with as many informative directions as possible, local to an iterate $\lambda^{(k)}$.

Example 3 (Addressing Flat-lining): *We revisit the objective function from Example 5 in the previous chapter where we recall we took* $\hat{f} : \Lambda = \mathbb{R}^P \to \mathbb{R} = \mathcal{D}$,

$$\hat{f}(\lambda; \xi) = \sum_{i=1}^{P} 2^{(-1)^{(i-1)}(i-1)} \lambda_i^2 + \epsilon(\xi), \tag{3.3.3}$$

$\epsilon(\cdot) \sim N(0, \sigma^2)$, *where* $\sigma^2 = 1 \times 10^{-3}$. *We note* f *is convex. We considered* $P = 10$ *and note* $D = 1$. *Recall that the minimizer of* $\hat{f}$ *is given by* $0 \in \Lambda$ *with* $f^* = 0$, $L_1 = 1024$, *and as* i *increases, terms in* $\hat{f}$ *become either more important or less important, depending on whether* i *is even or odd. We took an initial iterate similar to prior examples. Here, the determination of which variables are active depends completely on one's choice of eigenvalue threshold, which determines the number of directions to include in* $\tilde{\mathcal{A}}$. *We show convergence of STARS, regular FAASTARS (*$\tau = 0.9$*), FAASTARS with adaptive thresholding, and FAASTARS with active subcycling averaged over 100 trials in 3.3.*

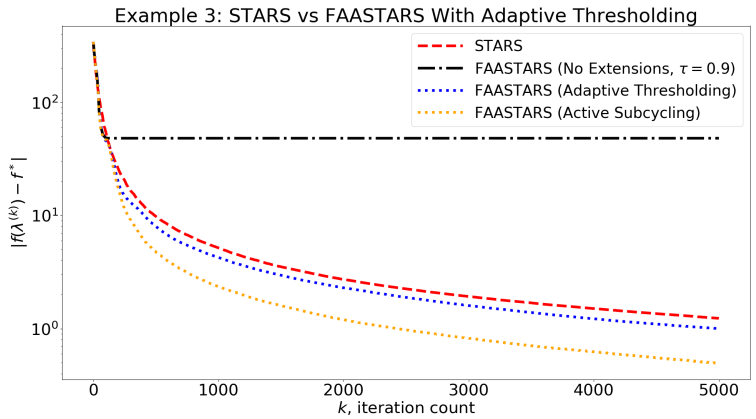


Figure 3.3: For Example 3, we compare the convergence of STARS, regular FAASTARS ($\tau = 0.9$), FAASTARS with adaptive thresholding, and FAASTARS with active subcycling to address flat-lining behavior originally demonstrated in Chapter II, Example 5.

We find that we are more or less able to reproduce the result of performing FAASTARS with fixed $j = 4$ and $j = 8$ (as in 2.5) for active subcycling and adaptive thresholding, respectively. We find that

in this example, active subcycling – behaving similarly to when we used a fixed $j = 4$ directions – offers the most computational benefit and indeed, both subcyling and thresholding prevent flat-lining and offer savings, compared to STARS. (As in Chapter II, Example 5, we did not run the trials until achievable accuracy was reached due its great computational expense.)

3.4 Discussion

Enhancements to the FAASTARS routine were presented that improve its numerical performance in certain examples – especially those that were a cause of concern in the Chapter 2 Numerical Examples. These modifications, including adaptive thresholding and active subcycling address the flat-lining issue discussed in the previous chapter.

We found alternative weighting schemes for random searches in ASTARS are somewhat unstable (when $\tilde{\mathcal{A}}$ is poorly estimated) but offers substantial computational winnings in some cases. Since the presented theoretical results do not consider alternative weights, we caution that convergence with other weighting schemes is not guaranteed, and should be researched and investigated further.

Finally, we provided a straightfoward approach for solving multi-objective optimization problems, involving vector-valued objective functions with noise. Using linear scalarization, we performed STARS, ASTARS, and FAASTARS to obtain non-unique properly efficient solutions. Here, we attempted to lay the groundwork for approaching multi-objective DFO using our novel methods. We again find computational winnings are possible when an AS is easily defined among the component-wise functions; however, once again, our numerical example possessed such an $\mathcal{A}$ by design. In practice, computational savings would be mild for less-clearly defined $\mathcal{A}$ (i.e., the spectrum of W does not contain a large drop-off) but still potentially beneficial in this computational expensive scenario, involving multiple real-valued functions with high-dimensional domains. Further research into regularizing the linearly scalarized objective functions we employed here could yield a method which could guarantee unique solutions and the potential for theoretical statements.

CHAPTER IV

SOLVING STOCHASTIC INVERSE PROBLEMS WITH ASTARS

4.1 Overview

In the first chapter, we presented equivalent deterministic optimization formulations for inverse problems – using either Bayesian inversion or *Data-Consistent Inversion (DCI)* – under the heavy assumptions that the model is linear in Λ and all associated densities are Gaussian Tarantola (2005); Wildey *et al.* (2018). We presented the Bayesian or classical formulation of the deterministic optimization problem, as well as a modified *Stochastic Inverse Problem (SIP)* solved via DCI, ensuring a data-consistent solution. Recall, DCI is a novel method used for solving a SIP in which the obtained data-consistent solution π_Λ^{up} matches the given observed data density $\pi_{\mathcal{D}}^{\text{obs}}$ exactly under the image of f.

In this chapter, we begin with the assumption that our data-to-parameter map is linear (or mildly nonlinear) and we also use Gaussian knowledge in Λ (for the initial density) and $\mathcal{D}$ (for the observed density) throughout. We assume $f(\lambda) = A\lambda$ for a $D \times P$ real-valued matrix A. When f is nonlinear, we will require a linearization of f around some point in Λ; most often, we use the mean of the initial density $\pi_\Lambda^{\text{init}}$, denoted $\bar{\lambda}$. Recall, the use of the terminology "initial density" in taken from the DCI setting; this density is called the prior density, $\pi_\Lambda^{\text{prior}}$ in the Bayesian setting. Since we often set the prior and initial densities equal for comparisons between Bayesian inversion and DCI, $\bar{\lambda}$ will denote the mean of both densities, unless otherwise specified.

We first recall the Bayesian misfit with the stochastic-free $f(\lambda) = A\lambda$,

$$S(\lambda) = \frac{1}{2}\left(\left|\left|C_{\mathcal{D}}^{-1/2}(A\lambda - \bar{d})\right|\right|_2^2 + \left|\left|C_\Lambda^{-1/2}(\lambda - \bar{\lambda})\right|\right|_2^2\right), \tag{4.1.1}$$

where $||\cdot||_2$ denotes the standard Euclidean norm (in Λ for the first term and $\mathcal{D}$ for the second term). Also, recall the DCI misfit

$$T(\lambda) = \frac{1}{2}\left(\left|\left|C_{\mathcal{D}}^{-1/2}(A\lambda - \bar{d})\right|\right|_2^2 + \left|\left|C_\Lambda^{-1/2}(\lambda - \bar{\lambda})\right|\right|_2^2 - \left|\left|C_A^{-1/2}(A(\lambda - \bar{\lambda}))\right|\right|_2^2\right), \tag{4.1.2}$$

where $C_A = AC_\Lambda A^T$; note that with $D = 1$, $C_A \in \mathbb{R}^{>0}$ and $C_A^{-1/2}$ is well-defined as long as $A \neq 0$. In Wildey *et al.* (2018), it is shown that (4.1.2) may be rewritten as

$$T(\lambda) = \frac{1}{2}\left(\left|\left|C_{\mathcal{D}}^{-1/2}(A\lambda - \bar{d})\right|\right|_2^2 + \left|\left|C_R^{-1/2}(\lambda - \bar{\lambda})\right|\right|_2^2\right), \tag{4.1.3}$$

where $C_R := C_\Lambda^{-1} - A^\top C_A^{-1} A$. This formulation of T will be particularly helpful in validating yet another alternate formulation we use in our proposed method in the next section, in using ASTARS to perform DCI.

In the misfits above, we replace $A\lambda$ with $\hat{f}(\lambda;\xi) = A\lambda + \epsilon(\xi)$, obtaining (non-deterministic) OUU problems. In the next section, we show that in this case one may re-express S and T (with $\hat{f}$ in place of A) as functions with additive noise. Since the additive noise emerging from the misfit functions in both cases are nonzero in mean, we suggest shifting the mean of the additive noise appearing in these quantities to zero.

We arrive at the problem of minimizing a convex function – which is really a composition of functions involving the noisy $\hat{f}$ – with zero-mean additive noise (when mean-shifting is performed). As the key step in our presented two-step solution process, we apply FAASTARS for the needed optimization to perform Bayesian inversion and DCI using deterministic, convex optimization in the presence of (zero-mean) additive noise. In previous chapters, we have shown ASTARS will perform more efficiently on average when a map permits meaningful dimension reduction in Λ and these computational winnings extend to the optimization approach to inversion in similar scenarios.

4.2 Methods

First, we present needed results regarding the general use of an AS in both the Bayesian and DCI settings. We then make considerations the noise in $\hat{f}$, and propose a *mean-shift* of the additive noise, which ensures the objective function we work with has zero-mean additive noise. Then, we present a two-step method for performing DCI using FAASTARS.

4.2.1 Utilizing $\mathcal{A}$ for Inversion with Linear $\hat{f}$

We begin with theoretical claims needed for our proposed two-step method to perform inversion using ASTARS. The authors in Constantine *et al.* (2016) show that the sensitivity matrix W_M associated with the AS of the data misfit function M with $D = 1$ is given by

$$W_M = \frac{1}{C_D^2} A^\top (AA^\top + \bar{d}\bar{d}^\top) A, \tag{4.2.1}$$

where $C_D > 0$ is real-valued here when $D = 1$ and A is non-stochastic. We note that in this case, W_M above will have the same eigenvectors (hence, active directions) as the general sensitivity W computed for the map A (with eigenvalues scaled by a factor of $C_{\mathcal{D}}^{-2}$). We seek to generalize this form for the case $D > 1$.

We make the standard predictability assumption that $Aw = d$ for any (fixed) $d \in \mathcal{D}$. The AS of f is given by the eigenvectors of $A^\top A$, while the AS of the general data misfit function $M(\lambda) := \frac{1}{2}\|f(\lambda) - d\|_{\mathcal{D}}^2 = \frac{1}{2}(\lambda - w)^\top A^\top C_{\mathcal{D}}^{-1} A(\lambda - w)$ is given by

$$W_M = A^\top C_{\mathcal{D}}^{-1} A \int_\Lambda (\lambda - w)(\lambda - w)^\top \rho(\lambda)\, d\Lambda\, A^\top C_{\mathcal{D}}^{-1} A. \tag{4.2.2}$$

Here the predictability assumption requires the inactive subspace (i.e., the nullspace) of A to be in the inactive subspace of M and is necessary for (4.2.2) to hold. If w is zero and the probability measure ρ is Gaussian then the integral is a constant and additionally when $D = 1$, (4.2.2) collapses to (4.2.1) (where $\bar{d} = 0$). The shift vector w only changes the constant, having the same effect as the shift by $\bar{d}\bar{d}^\top$ in (4.2.1). A change of variables with $\hat{\lambda} = \lambda - w$ completes the argument. Since $A^\top A\lambda^{\mathcal{I}} = 0$ – where $\lambda^{\mathcal{I}}$ denotes any λ projected into $\mathcal{I}$ – we also have $W_M\lambda^{\mathcal{I}} = 0$, showing that shifts by different w will not change the active directions obtained from W_M. We conclude that for linear maps A, the general data misfit function M has the same AS directions as A itself, a fact we shall leverage in our computational approach.

We now state and prove two theoretical claims crucial for justifying our two-step approach to Bayesian inversion and DCI, respectively. Let C_Λ and $C_{\mathcal{D}}$ denote covariance matrices of initial/prior and observed/data likelihood densities (depending on whether we are discussing Bayesian or Data-Consistent Inversion) respectively. Recall that $||\lambda||_\Lambda = ||C_\Lambda^{-1/2}\lambda||_2$, where $||\cdot||_2$ is the Euclidean 2-norm in Λ, and likewise for $\mathcal{D}$ and $C_{\mathcal{D}}$.

Claim 1: Let $A \in \mathbb{R}^{P\times 1}$ and $Aw = \bar{d}$ for $w \in \Lambda$ and where $\bar{d}$ is the mean of the observed. Let $\lambda^{\mathcal{A}} := P_{\mathcal{A}}(\lambda)$, and similarly for $\mathcal{I}$. The Bayesian MAP point (the minimum of S) can be obtained by instead minimizing

$$\tilde{S}(\lambda) := \frac{1}{2}\left(||A\lambda^{\mathcal{A}} - \bar{d}||_{\mathcal{D}}^2 + ||\lambda^{\mathcal{A}} - \bar{\lambda}^{\mathcal{A}}||_\Lambda^2 + ||\lambda^{\mathcal{I}} - \bar{\lambda}^{\mathcal{I}}||_\Lambda^2\right). \tag{4.2.3}$$

Proof: First, observe that if A is invertible, all variables are active ($\mathcal{A} = \Lambda$) and the third term in (4.2.3) above drops out, meaning $\tilde{S}$ is equivalent to S. Noting $\mathcal{A} = \Lambda \implies \lambda^{\mathcal{A}} = \lambda$, the minimizers of S and $\tilde{S}$ are equal since $S = \tilde{S}$ in this case.

When A is not invertible, $\mathcal{I}$ is nontrivial. Given the linear map $f(\lambda) = A\lambda$ and the standard loss function, we make the standard predictability assumption that $Aw = \bar{d}$ for some $w \in \Lambda$. Expanding the data mismatch term (the first term in (4.1.1)),

$$(A(\lambda - w))^\top C_D^{-1}(A(\lambda - w)).$$

Writing $\lambda = \lambda^{\mathcal{A}} + \lambda^{\mathcal{I}}$ in the form above and noting $A\lambda^I = 0$, we see that the data mismatch term in $\tilde{S}$ (the first term in (4.2.3)) is equivalent.

Expanding the Bayesian regularization term (term 2 in (4.1.1)) and taking $\lambda = \lambda^{\mathcal{A}} + \lambda^{\mathcal{I}}$ and $\bar{\lambda} = \bar{\lambda}^{\mathcal{A}} + \bar{\lambda}^{\mathcal{I}}$, we write

$$(\lambda^{\mathcal{A}} - \bar{\lambda}^{\mathcal{A}} + \lambda^{\mathcal{I}} - \bar{\lambda}^{\mathcal{I}})^\top C_\Lambda^{-1}(\lambda^{\mathcal{A}} - \bar{\lambda}^{\mathcal{A}} + \lambda^{\mathcal{I}} - \bar{\lambda}^{\mathcal{I}}),$$

which, by linearity, writes

$$(\lambda^{\mathcal{A}} - \bar{\lambda}^{\mathcal{A}})^\top C_\Lambda^{-1}(\lambda^{\mathcal{A}} - \bar{\lambda}^{\mathcal{A}}) + (\lambda^{\mathcal{I}} - \bar{\lambda}^{\mathcal{I}})^\top C_\Lambda^{-1}(\lambda^{\mathcal{I}} - \bar{\lambda}^{\mathcal{I}}).$$

Observe we have obtained an expansion of the second two terms in (4.2.3), showing the regularization in $\tilde{S}$ is equivalent to the regularization in S.

We have shown $\tilde{S}$ has an equivalent data mismatch and regularization term to those appearing in S. Thus, it is equivalent to minimize $\tilde{S}$ and S. ■

Claim 2: Let $A \in \mathbb{R}^{P\times 1}$ and $Aw = \bar{d}$ for $w \in \Lambda$ and where $\bar{d}$ is the mean of the observed. Let $\lambda^{\mathcal{A}} := P_{\mathcal{A}}(\lambda)$, and similarly for $\mathcal{I}$. The data-consistent MUD point (the minimum of T) can be obtained by instead minimizing

$$\tilde{T}(\lambda) := \frac{1}{2}\left(||A\lambda^{\mathcal{A}} - \bar{d}||^2_{\mathcal{D}} + ||\lambda^{\mathcal{I}} - \bar{\lambda}^{\mathcal{I}}||^2_{\Lambda}\right). \tag{4.2.4}$$

Proof: First, observe that if A is invertible, all variables are active ($\mathcal{A} = \Lambda$) and thus, all variables are informative to the data, and one can show $C_R = 0$ in (4.1.3). In this case, the unique MUD point is obtained by solving the data misfit problem only, meaning deregularization terms vanish. Again noting $\mathcal{A} = \Lambda$, $\lambda^{\mathcal{A}} = \lambda$, and the minimizers of T and $\tilde{T}$ are equal.

When A is not invertible, $\mathcal{I}$ is nontrivial, meaning there are directions in Λ which are not informing the data. Given the linear map $f(\lambda) = A\lambda$ and the standard loss function, we make the standard predictability assumption that $Aw = \bar{d}$ for some $w \in \Lambda$. Expanding the data mismatch term (the first term in (4.1.3)),

$$(A(\lambda - w))^\top C_D^{-1}(A(\lambda - w)).$$

Writing $\lambda = \lambda^{\mathcal{A}} + \lambda^{\mathcal{I}}$ in the form above and noting $A\lambda^I = 0$, we see that the data mismatch term in $\tilde{T}$ (the first term in (4.2.4)) is equivalent.

We express the DCI regularization term (the second term in (4.1.3)) as

$$(\lambda - \bar{\lambda})^\top (C_\Lambda^{-1} - A^\top (C_A)^{-1} A)(\lambda - \bar{\lambda}), \quad C_A = A C_\Lambda A^\top.$$

Note that the projector onto the column space of $C_\Lambda^{\frac{1}{2}} A^\top$ is

$$C_\Lambda^{\frac{1}{2}} A^\top (A C_\Lambda A^\top)^{-1} A C_\Lambda^{\frac{1}{2}},$$

so we may rewrite this regularization term as

$$(\lambda - \bar{\lambda})^\top C_\Lambda^{-\frac{1}{2}} \left(I - P_{C_\Lambda^{\frac{1}{2}} A^\top} \right) C_\Lambda^{-\frac{1}{2}} (\lambda - \bar{\lambda}),$$

where P denotes the projection matrix. Equivalently,

$$(\lambda - \bar{\lambda})^\top C_\Lambda^{-\frac{1}{2}} P_{\text{null}\left(A C_\Lambda^{\frac{1}{2}} \right)} C_\Lambda^{-\frac{1}{2}} (\lambda - \bar{\lambda}), \tag{4.2.5}$$

where we have used that the orthogonal complement of col(B)=null($B^\top$).

It is straightforward to show that $C_\Lambda^{-\frac{1}{2}} \lambda_\mathcal{I}$ is in null($A C_\Lambda^{\frac{1}{2}}$), and that $C_\Lambda^{-\frac{1}{2}} \lambda_\mathcal{A}$ is in its orthogonal complement. Therefore, (4.2.5) reduces to

$$(\lambda_\mathcal{I} - \bar{\lambda}_\mathcal{I})^\top C_\Lambda^{-1} (\lambda_\mathcal{I} - \bar{\lambda}_\mathcal{I}).$$

Writing $\lambda_\mathcal{I}$ as $V_\mathcal{I}^\top \lambda$, we get the regularization contribution to the inverse covariance matrix,

$$C_R^{-1} = V_\mathcal{I} C_\Lambda^{-1} V_\mathcal{I}^\top,$$

showing the covariance matrix used in the DCI regularization term (the second term in (4.1.3))

Therefore, $\tilde{T}$ has an equivalent data mismatch term to T and an equivalent regularization, where $\tilde{T}$ only regularizes solutions in directions spanned by $\mathcal{I}$. We have shown it is equivalent to minimize $\tilde{T}$ and T. ■

We will propose a two step method of minimizing $\tilde{T}$ utilizing ASTARS-based methods (specifically, phase 3 of FAASTARS). Since minimizing the second term of $\tilde{T}$ is straightforward with $\mathcal{I}$ in hand, the method focuses on minimizing the data mismatch term, leveraging dimension reduction (if possible) and employing FAASTARS phase 3, minimizing this term in learned active variables of the misfit function.

Before we present the two-step methods for using ASTARS-based DFO to perform Bayesian inversion and DCI in this Gaussian, linear case, we first consider the impact the noise in $\hat{f}$ has on the generalized data misfit term, appearing in both S and T.

4.2.1.1 Mean-Shifting Noise in Data Misfit when $D = 1$

We assume available evaluations of maps are noisy, hence misfits S and T will involve $\hat{f} = A+\epsilon$ instead of the stochastic-free A. The proposed two-step process for performing DCI using FAASTARS only requires minimizing the general form $||\hat{f}(\lambda;\xi) - \bar{d}||^2_{\mathcal{D}}$, which is also true in the Bayesian case, where regularizing does not involve evaluating the map. When $\mathcal{D} = \mathbb{R}$ – so $D = 1$ – the data covariance matrix $C_{\mathcal{D}}$ is a positive scalar (which can be factored out) so that we may minimize the general noisy data misfit function, $M_\epsilon(\lambda;\xi) := (\hat{A}(\lambda;\xi)-\bar{d})^2$. (Here we are concerned with *minimizers*, not minimum values.) Expressing the noise we introduce into M,

$$M_\epsilon(\lambda;\xi) = (A\lambda + \epsilon(\xi) - \bar{d})^2. \tag{4.2.6}$$

We re-write M_ϵ as $M_\epsilon(\lambda;\xi) = (A\lambda - \bar{d})^2 + \eta(\epsilon(\xi))$, a function with additive noise. Here, η will be a new random variable, a function of the random variable ϵ, still specified by draws ξ. We have

$$M_\epsilon(\lambda;\xi) = (A\lambda - \bar{d})^2 + \eta(\epsilon(\xi)), \quad \eta(\epsilon(\xi)) := 2(A\lambda - \bar{d})\epsilon(\xi) + \epsilon(\xi)^2. \tag{4.2.7}$$

Recalling $\mathbb{E}(\epsilon(\xi)) = 0$ for all ξ,

$$\mathbb{E}_\xi(M_\epsilon(\lambda;\xi)) = \mathbb{E}_\xi(\eta(\epsilon(\xi))) = \mathbb{E}_\xi(\epsilon(\xi)^2). \tag{4.2.8}$$

With $\mathbb{E}(\epsilon) = 0$, all we can say about $\mathbb{E}(\epsilon^2)$ is that it is finite; in particular, there is no guarantee that this moment is zero. In other words, M_ϵ exhibits additive noise with a nonzero mean. We define

$$\tilde{M}_\epsilon(\lambda;\xi) := M_\epsilon(\lambda;\xi) - \mathbb{E}_\xi(\epsilon(\xi)^2) = (A\lambda - \bar{d})^2 + \eta(\epsilon(\xi)) - \mathbb{E}_\xi(\epsilon(\xi)^2), \tag{4.2.9}$$

where we now have $\mathbb{E}_\xi\left(\tilde{M}_\epsilon(\lambda;\xi)\right) = (A\lambda - \bar{d})^2$, meaning the additive noise in $\tilde{M}_\epsilon$ is zero-mean.

If, for instance, we consider the fairly common/standard assumption that the noise $\epsilon \sim N(0,\sigma^2)$ for all independent identically distributed draws, we have $\mathbb{E}(\epsilon^2) = \sigma^2$, a moment which we already estimate via ECNoise Moré and Wild (2015) to perform STARS. So in the Gaussian noise case, we already have the necessary correction term in hand, albeit estimated. When the second moment of ϵ is unclear, one could perform *Kernel Density Estimation (KDE)* on a handful of samples of the observed variance in the noise via differencing techniques along the lines of Moré and Wild (2015).

We also present the variance of η,

$$\mathrm{Var}_\xi(\eta(\epsilon(\xi))) = \mathbb{E}_\xi(\epsilon(\xi)^4) + 2m\mathbb{E}_\xi(\epsilon(\xi)^3) + m^2\mathbb{E}_\xi(\epsilon(\xi)^2) - \mathbb{E}_\xi(\epsilon(\xi)^2)^2, \tag{4.2.10}$$

where $m = 2(A\lambda - \bar{d})$ Note that the variance of η involves higher-order moments of ϵ, and may or may not be easily specified using analytic techniques depending on the chosen noise model. As such, we propose learning the variance of η via ECNoise in our methods, to avoid direct computations of potentially unknown moments of ϵ. However, when $\epsilon \sim N(0, \sigma^2)$, we have $\mathrm{Var}_\xi(\eta(\epsilon(\xi))) = 2\sigma^4 + m^2\sigma^2$. Notice than as $\lambda \to \lambda^{\mathrm{MUD}}$ that $m \to 0$ and the variance in η approaches $2\sigma^4$.

4.2.1.2 Two-Step Bayesian Inversion via (A)STARS

Below, we present a two-step solution process for performing Bayesian inversion in our setting leveraging dimension reduction for computational winnings in minimizing $\tilde{S}$ in (4.2.3), if possible. Note our expression in (4.2.3) and claim about its equivalence to S (in (4.1.1)) are key in the proposed process. Another crucial assumption is that $\mathcal{A}$ is the same for A and the data misfit term, M.

Bayesian Inversion via ASTARS, Step 1: First, perform STARS on data mismatch term of S only and perform noise mean-shifting; i.e., minimize $||\hat{f}(\lambda;\xi) - \bar{d}||^2_{\mathcal{D}} - \mathbb{E}\left(\epsilon(\xi)^2\right)$, where the second moment of ϵ may need to be estimated. Use $\bar{\lambda}$, the mean of the prior as initial iterate for STARS. When enough iterates of STARS are taken to form $\tilde{\mathcal{A}}$ (of A, not the data misfit), perform Phase 3 of FAASTARS (approximate ASTARS using $\tilde{\mathcal{A}}$) for more efficient convergence to a non-unique minimizer of the data misfit. Optimize $||\hat{f}(\lambda^{\tilde{\mathcal{A}}};\xi) - \bar{d}||^2_{\mathcal{D}} + ||\lambda^{\tilde{\mathcal{A}}} - \bar{\lambda}^{\tilde{\mathcal{A}}}||_\Lambda - \mathbb{E}\left(\epsilon(\xi)^2\right)$ – the first two terms in (4.2.3) but with $\hat{f}$ replacing A – and call the obtained minimizer $\hat{\lambda}^{\tilde{\mathcal{A}}}$, which is unique in $\tilde{\mathcal{A}}$.

Bayesian Inversion via ASTARS, Step 2: Project the prior mean onto $\mathcal{I}$ to obtain $\bar{\lambda}^{\mathcal{I}}$. $\lambda^{\mathrm{MAP}} = \hat{\lambda}^{\mathcal{A}} + \bar{\lambda}^{\mathcal{I}}$. Use surrogate (obtained in step 1 to train $\mathcal{A}$) for $\hat{f}$ as A in (1.4.5) to compute C_{post}.

Observe that the two-step process corresponds to minimizing S over $\mathcal{A}$ (step 1) and then performing necessary regularization in $\mathcal{I}$ to form the solution (step 2). Note that using $\tilde{\mathcal{A}}$ in place of $\mathcal{A}$ introduces error into the MAP point (and corresponding covariance); for reasonable $\tilde{\mathcal{A}}$, these errors are mild but observable in our experiments. We shall use a similar process to perform DCI, which we present next.

4.2.1.3 Two-Step DCI via (A)STARS

We present a two-step process to perform DCI similar to the proposed two-step process for Bayesian inversion. Once again, our theoretical statements are key in the proposed method.

DCI via ASTARS, Step 1: First, perform STARS on data mismatch term of S only and perform noise mean-shifting; i.e., minimize $||\hat{f}(\lambda;\xi) - \bar{d}||^2_{\mathcal{D}} - \mathbb{E}\left(\epsilon(\xi)^2\right)$, where the second moment of ϵ may need to be estimated. Use $\bar{\lambda}$, the mean of the prior as initial iterate for STARS. When enough iterates of

STARS are taken to form $\tilde{\mathcal{A}}$ (of A, not the data misfit), perform Phase 3 of FAASTARS (approximate ASTARS using $\tilde{\mathcal{A}}$) for more efficient convergence to a non-unique minimizer of the data misfit, which will also minimize the first term in $\tilde{T}$.

DCI via ASTARS, Step 2: Project the prior mean onto $\mathcal{I}$ to obtain $\bar{\lambda}^{\mathcal{I}}$. $\lambda^{\text{MUD}} = \hat{\lambda}^{\mathcal{A}} + \bar{\lambda}^{\mathcal{I}}$. Use surrogate (obtained in step 1 to train $\mathcal{A}$) for $\hat{f}$ as A in (1.4.8) to compute C_{up}.

This two-step method is conceptually quite similar to the Bayesian two-step method, in which we dichotomize the problem in $\mathcal{A}$ and $\mathcal{I}$ to leverage the possible computational savings. We now turn our attention to the nontrivial case that $\hat{f}$ is nonlinear.

4.2.2 Considerations for Nonlinear $\hat{f}$

If the noise-free model $f(\lambda)$ is nonlinear, one typically linearizes around a point in Λ; in this setting, linearizing around $\bar{\lambda}$ is standard. Then, one might proceed with one of the methods outlined above using a linear surrogate A in place f. For strongly nonlinear $\hat{f}$ (where a linearization about $\bar{\lambda}$ cannot capture its behavior) this approach will fail, and more sophisticated considerations are needed, which we outline here.

When f is nonlinear, one can show the eigenvectors related to the sensitivity of the misfit will not generally match the eigenvectors of the map. As a consequence, we must store all collected values of $f(\lambda)$ (which are formed in the computation of the data misfit function) so that an AS can be trained for f. We suggest minimizing the data misfit as before – potentially using the AS of the misfit for computational winnings – and also train an AS from (noisy) observed $f(\lambda) + \epsilon$ values. In other words, step one of our general approach for inversion with ASTARS would remain almost completely unmodified, except for the necessary storage of evaluations of the map (not just the misfit). Step two would involve using the (approximated) inactive subspace of f – not the misfit – to perform the regularization in $\mathcal{I}$ step.

First, we present general formulations of the MAP and MUD points (and corresponding covariance matrices) in the nonlinear setting. We assume f permits meaningful dimension reduction and as usual denote its active set with $\mathcal{A}$ and the complimentary inactive set as $\mathcal{I}$.

We begin with the presentation of the DCI results, which we are able to cleanly present here. (The Bayesian case is more algebraically involved, and details are provided in an appendix, whereas the DCI case is naturally de-coupled, as we shall see.) Let λ^* be a minimizer $T(\lambda)$, equal to $\lambda^*_{\mathcal{A}} + \bar{\lambda}_{\mathcal{I}}$, the generalized MUD point. We linearize f around λ^* and substitute into T to obtain

$$(\lambda - \lambda^*)^\top \nabla_{\mathcal{A}} f(\lambda^*)^\top C_{\mathcal{D}}^{-1} \nabla_{\mathcal{A}} f(\lambda^*)(\lambda - \lambda^*) + (\lambda - \lambda^*)^\top V_{\mathcal{I}} C_{\Lambda}^{-1} V_{\mathcal{I}}^\top (\lambda - \lambda^*),$$

where $\nabla_{\mathcal{A}} f(\lambda^*) := \nabla f(\lambda^*) V_{\mathcal{A}}^\top$ and note that the second term is a regularization in the inactive subspace, independent of f.

If $\dim(\mathcal{D}) = D = 1$ and $\dim(\mathcal{A}) = j \geq 1$, then the updated covariance takes on the following form:

$$C_{up}^{-1} = V_{\mathcal{A}} D^\dagger(\nabla_{\mathcal{A}} f(\lambda^*)) C_{\mathcal{D}}^{-1} D^\dagger(\nabla_{\mathcal{A}} f(\lambda^*)) V_{\mathcal{A}}^\top + V_{\mathcal{I}} V_{\mathcal{I}}^\top C_\Lambda^{-1} V_{\mathcal{I}} V_{\mathcal{I}}^\top, \tag{4.2.11}$$

where $D^\dagger(\nabla_{\mathcal{A}} f)$ is a $j \times j$ diagonal matrix with the entries of $\nabla_{\mathcal{A}} f$ along the diagonal. Note that if $j = D = 1$ (which we consider in our numerical example with nonlinear f) C_{up}^{-1} collapses to

$$\nabla f(\lambda^*) C_D^{-1} \nabla f(\lambda^*)^\top + V_{\mathcal{I}} V_{\mathcal{I}}^\top C_\Lambda^{-1} V_{\mathcal{I}} V_{\mathcal{I}}^\top.$$

For the Bayesian regularization, let now λ^* minimize $S(\lambda)$. Let $S = S_1 + S_2$, where S_1 denotes the first two terms in $\tilde{S}$ (see (4.2.3)) (data misfit and regularization in $\mathcal{A}$) and S_2 denotes the last term in $\tilde{S}$ (regularization in $\mathcal{I}$, independent of f). We take it as the data-misfit in the active variables of f plus the projection of the prior onto the inactive variables. Taylor expanding at this point and substituting into S_1 yields

$$\left(f(\lambda^*) + \nabla f(\lambda^*)^\top (\lambda - \lambda^*) - \bar{d}\right) C_{\mathcal{D}}^{-1} \left(f(\lambda^*) + \nabla f(\lambda^*)^\top (\lambda - \lambda^*) - \bar{d}\right)^\top + (\lambda - \bar{\lambda}) V_{\mathcal{A}} C_\Lambda^{-1} V_{\mathcal{A}}^\top (\lambda - \bar{\lambda})^\top.$$

The next step involves a great deal of algebra to involve $\nabla_{\mathcal{A}}$ in place of ∇ – and is given in Appendix B – but mainly requires using the fact that at λ^*, the gradient of S_1 is zero. The result is

$$C_{\text{post}}^{-1} = \nabla_{\mathcal{A}} f(\lambda_*) C_{\mathcal{D}}^{-1} \nabla_{\mathcal{A}} f(\lambda_*)^\top + V_{\mathcal{A}} C_\Lambda^{-1} V_{\mathcal{A}}^\top + V_{\mathcal{I}} C_\Lambda^{-1} V_{\mathcal{I}}^\top. \tag{4.2.12}$$

Note that the inactive regularization term above and S_2 are the same. That is,

$$(\lambda - \lambda^*)^\top V_{\mathcal{I}} C_\Lambda^{-1} V_{\mathcal{I}}^\top (\lambda - \lambda^*).$$

is the same as

$$(\lambda_{\mathcal{I}} - \bar{\lambda_{\mathcal{I}}})^\top C_\Lambda^{-1} (\lambda_{\mathcal{I}} - \bar{\lambda_{\mathcal{I}}}),$$

and similarly for the active regularization part of S_1.

In the next section, we experiment with similar two step methods proposed for the linear case, which involves iteratively stepping towards a potentially non-unique solution to the data mismatch prob-

lem, leveraging (FA)ASTARS for computational savings in this process, and forming a surrogate for f in order to estimate ∇f to form the MAP/MUD points and covariances above. Besides requiring a reasonable linear approximation around the obtained MAP/MUD points, another consequence of our approach is that we also must assume the corresponding posterior/updated densities are roughly Gaussian (with a single mode). In situations where these assumptions are inappropriate, one may need more sophisticated surrogates, as well as Gaussian mixture models to approximate multi-modal solutions, neither of which we consider here.

4.3 Numerical Results

In this section, we present results of using the ASTARS-based two-step approach to perform DCI in example problems, considering both linear and nonlinear noisy maps and high-dimensional Λ. Here, we will consider $\mathcal{D} = \mathbb{R}$.

Example 1: *Let* $\Lambda = \mathbb{R}^{25}$, $\hat{A}\lambda = A\lambda + \epsilon$, $A \in \mathbb{R}^{25\times 1}$, $A = [10 \quad 0 \quad 0 \quad \cdots \quad 0]$ *and* $\epsilon \sim N(0, 10^{-6})$. *We assume the initial distribution* $\lambda \sim N(\bar{\lambda} = 0, C_\Lambda = I_P)$ *and observed data* $d \sim N(\bar{d} = 10, C_\mathcal{D} = 1)$. *Here, the solution is* $\lambda^{MUD} = (1 \quad 0 \quad 0 \quad \cdots \quad 0)^\top$ *since* $A\lambda^{MUD} = \bar{d}$ *and* $\lambda_i^{MUD} = \bar{\lambda}_i = 0$ *in the inactive directions* $i = 2, \ldots, 25$. *We performed 100 trials of the two-step method (outlined in detail below). We also show the convergence of using STARS only to minimize the data misfit in step one, for comparisons to the more efficient FAASTARS, showing the computational savings we achieve by stepping in the AS to obtain the minimizer in step one.*

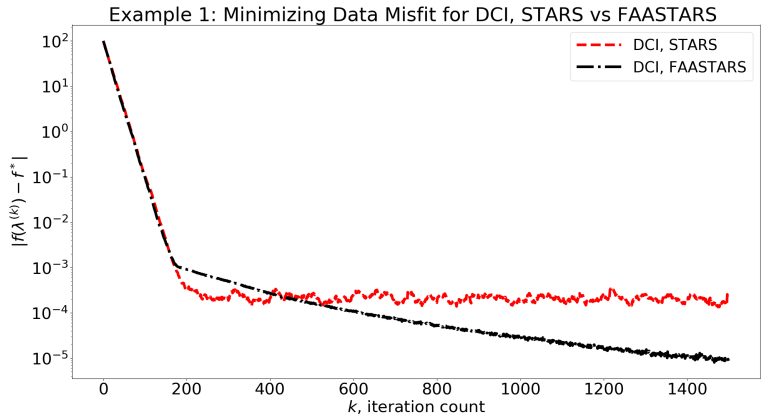


Figure 4.1: We show the average convergence of STARS versus FAASTARS for Step 1 of the DCI solution process. Note that in the vertical axis labeling $f = M$, the data misfit, and $f^* = M^* = 0$, the minimum value of the misfit.

Our solution process was as follows. We minimized $(A\lambda + \epsilon - 10)^2 - 10^{-6}$ using STARS with $\bar{\lambda} = 0$ as initial iterate for enough iterations to fit a surrogate – a little less than 200 iterations with $P = 25$.

Recall the subtraction -10^{-6} comes from our suggested mean-shifting, where we subtract by the variance of ϵ since ϵ is normally distributed (see last subsection). We obtained $\tilde{\mathcal{A}}$, the learned AS of the *misfit* – which, in this case, is equal to the AS of the map A – using a quadratic surrogate fit to the taken STARS samples and used only 75% of the eigenvalues by weight (i.e., $\tau = 0.75$) to take ASTARS steps until convergence to a (non-unique) minimizer of the misfit function is found. In both STARS and FAASTARS, we used the correct L_1 (of the misfit) but we estimated the variance in η (not ϵ) using ECNoise. We used active subspace retraining every $2P = 50$ iterations for FAASTARS and noise regularization is unused. The results of performing step one over 100 trials is shown in 4.1.

We then updated data misfit minimizer λ^* obtained from ASTARS using the closed-form solution to minimum of the regularization of the learned inactive variables, setting $\lambda_i^* = \bar{\lambda}_i$, $i = \tilde{j}+1, \ldots, 25$, where $\tilde{j}+1$ should be 2, but may be different depending on the quality of $\tilde{\mathcal{A}}$ (i.e., if the AS is poorly approximated) as well as the threshold τ. We observed mixed qualities in the various obtained λ^{MUD} here, where regularization was sometimes performed in too few variables, meaning some inactive components were not reset back to the initial mean of 0. In particular, we found almost always $\tilde{j} \leq 3$ near the end of each FAASTARS trial, but only some trials achieve $\tilde{j} = j = 1$ (many get stuck at $\tilde{j} = 2$ here).

Notice, as well, the counter-intuitive benefit of FAASTARS' slow but steady convergence, once it learns $\tilde{\mathcal{A}}$ and begins taking steps in this smaller (usually between 1- and 3-dimensional) space. By 200 iterations, the data misfit has been driven to values approaching – but not quite reaching – the level of observable noise. STARS continues to step in Λ, unable to perturb λ_1 at a fine enough level to drive the misfit down to the noise level. FAASTARS, however, zeros in on minimizing the active variable λ_1 (and sometimes, incorrectly, a small handful of other directions) so that we eventually reach the noise level that STARS cannot. We say this result is counter-intuitive since we are used to seeing the benefits of a faster convergence rate from FAASTARS. However, here, computational savings are not achieved from a faster convergence rate than what we see in the first 200 iterations; instead, we achieve savings from reaching the level of the noise as efficiently as we can.

Example 2: *Let $\Lambda = \mathbb{R}^{40}$, $\hat{f}(\lambda) = \lambda_1^3 + \epsilon$ and $\epsilon \sim N(0, 10^{-6})$. We assume the initial distribution $\lambda \sim N(\bar{\lambda} = \mathbf{1}, C_\Lambda = I_P)$ (where $\mathbf{1}$ is the vector of all 1's in Λ). We considered two observed densities, $d \sim N(\bar{d} = 8, C_\mathcal{D} = 1)$ and $d \sim N(\bar{d} = 27, C_\mathcal{D} = 0.1)$. Here, the solution is $\lambda^{MUD} = (m^* \quad 1 \quad 1 \quad \cdots \quad 1)^\top$ – where $m^* = 2$ when $\bar{d} = 8$ and $m^* = 3$ when $\bar{d} = 27$ – ensuring $f(\lambda^{MUD}) = \bar{d}$ and $\lambda_i^{MUD} = \bar{\lambda}_i = 1$ in the inactive directions $i = 2, \ldots, 40$. We performed 100 trials of the step one of our two-step approach to solve the data mismatch problem when $d \sim N(\bar{d} = 8, C_\mathcal{D} = 1)$. (We obtained almost identical results when $d \sim N(\bar{d} = 27, C_\mathcal{D} = 0.1)$.) We chose to complete solve the problem in the case $d \sim N(\bar{d} = 27, C_\mathcal{D} = 0.1)$ by*

performing step two of our two-step process, and analyze various obtained updated densities in several plots below.

To solve the data mismatch problem via step one of our two-step process with $d \sim N(\bar{d} = 8, C_{\mathcal{D}} = 1)$, our solution process was as follows. We minimized $(\lambda^3 + \epsilon - 8)^2 - 10^{-6}$ using STARS with $\bar{\lambda}$ as the initial iterate for enough iterations to fit a surrogate. We obtained $\tilde{\mathcal{A}}$, the learned AS of the *misfit* – which, in this case, is equal to the AS of the map A – using a quadratic surrogate fit to the taken STARS samples and used 90% of the eigenvalues by weight (i.e., $\tau = 0.9$) to take ASTARS steps until convergence to a (non-unique) minimizer of the misfit function is found. In both STARS and FAASTARS, we used the correct L_1 (of the misfit) but we estimated the variance in η (not ϵ) using ECNoise. We used active subspace retraining every $2P = 80$ iterations for FAASTARS and noise regularization is unused. We see the convergence to the single active component of λ^{MUD} is more efficient (averaged over 100 trials, shown in 4.2) using FAASTARS, leveraging dimension reduction in this nonlinear case. Note the immediately faster convergence of FAASTARS – here we used alternative weighting to obtain better convergence.

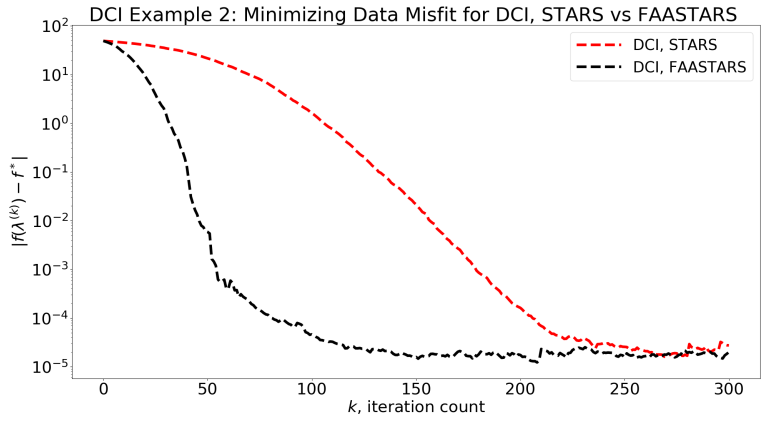


Figure 4.2: We show the average convergence of STARS versus FAASTARS for Step 1 of the DCI solution process in this nonlinear case and $d \sim N(\bar{d} = 8, C_{\mathcal{D}} = 1)$. Note that in the vertical axis labeling $f = M$, the data misfit, and $f^* = M^* = 0$, the minimum value of the misfit.

We note that we obtained nearly identical results for solving the data misfit problem when $d \sim N(\bar{d} = 27, C_{\mathcal{D}} = 0.1)$. We now present the results of applying step two in this case (with $d \sim N(\bar{d} = 27, C_{\mathcal{D}} = 0.1)$) and discuss the ramifications of using approximate information – namely, $\tilde{\mathcal{A}}$ – in this setting. We used the approximated $\tilde{\mathcal{A}}$ and $\nabla f(\lambda^{\text{MUD}})$ (from surrogates of various qualities) to form the MUD point and corresponding covariance C_{up} using the regularization step (step 2 in our process) and (4.2.11).

As shown in 4.3, we found that even when the subspace distance between $\tilde{\mathcal{A}}$ and $\mathcal{A}$ was small – even as low as 2% – the pushforwards of the corresponding (approximated) updated solutions does not

match the observed data to an acceptable extent. One will notice that the approximated solutions appear to have incorrect means, slightly larger than the correct mean of $\bar{d} = 27$; as well, even with small subspace distances, the covariance (in this case the real-valued variance) of the pushfoward is clearly too large.

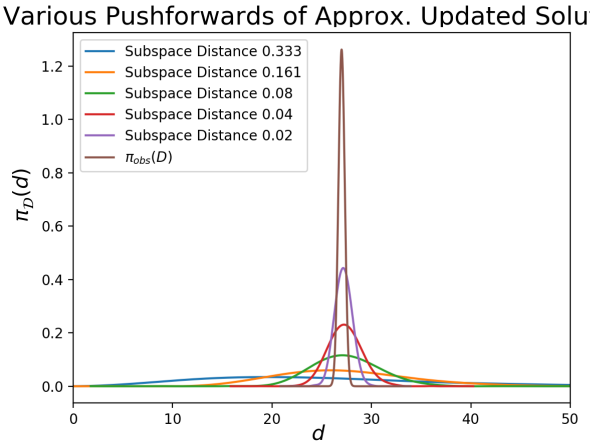


Figure 4.3: We show in this nonlinear example that the pushforwards of solutions $\pi_{\Lambda}^{\text{up}}$ – formed with inexact $\tilde{\mathcal{A}}$ – are inaccurate, even if the subspace distance between $\tilde{\mathcal{A}}$ and $\mathcal{A}$ is small.

We found that by using the exact $\mathcal{A}$, our two-step process succeeds in producing an updated solution with a pushforward roughly equal to the observed data, as we see in 4.4. The pushfoward does not match the data exactly, mainly due to the additive noise in $\hat{f}$. Despite slight inaccuracy, we do obtain an acceptably data-consistent updated solution with our approach. We conclude by showing the great cost reduction our approach may offer – requiring only a few hundred $\hat{f}$ evaluations.

Despite great sensitivity to $\tilde{\mathcal{A}}$ in our approach, we note that obtaining the DCI solution in this example via analytic methods is either expensive, inaccurate, or both, where the difficulty lies in approximating the pushforward of $\pi_{\Lambda}^{\text{init}}$, denoted $\pi_{\mathcal{D}}^{f(\Lambda)}$. (Recall, we assume exact knowledge of $\pi_{\Lambda}^{\text{init}}$ and $\pi_{\mathcal{D}}^{\text{obs}}$ – the other two terms appearing in the analytic DCI updated density.)

From analytic approaches – which are still possible in this example, but often would not be when f is nonlinear – we know $\pi_{\mathcal{D}}^{f(\Lambda)}$ is and *indeterminate distribution*, undefined at $d = 0$. Using a million samples and KDE, we obtain a good approximation to $\pi_{\mathcal{D}}^{f(\Lambda)}$. However, a million $\hat{f}$ evaluations in our setting would be prohibitively expensive. Using the same number of $\hat{f}$ evaluations as our ASTARS approach (300 evaluations), the accuracy is poor. And even with a million samples, we obtain the most inaccurate pushforward by using quadratic surrogates. These results are shown in 4.5.

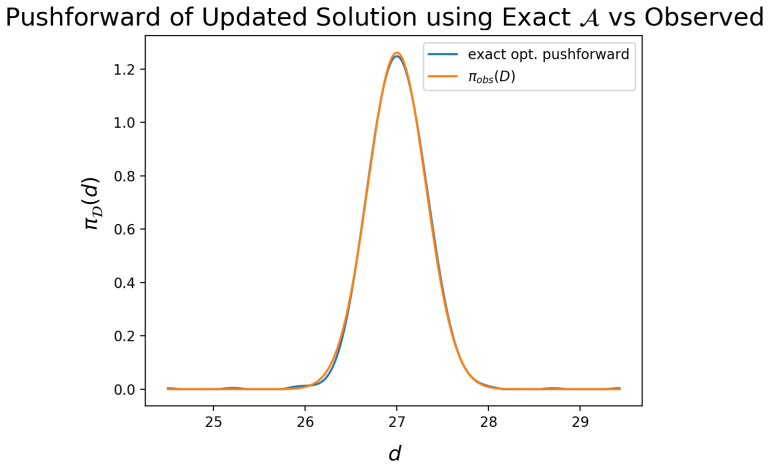


Figure 4.4: We show in this nonlinear example that the pushforward of a solution π_Λ^{up} – formed with $\mathcal{A}$ exact – greatly improves solution accuracy, compared to using fairly-well approximated $\tilde{\mathcal{A}}$.

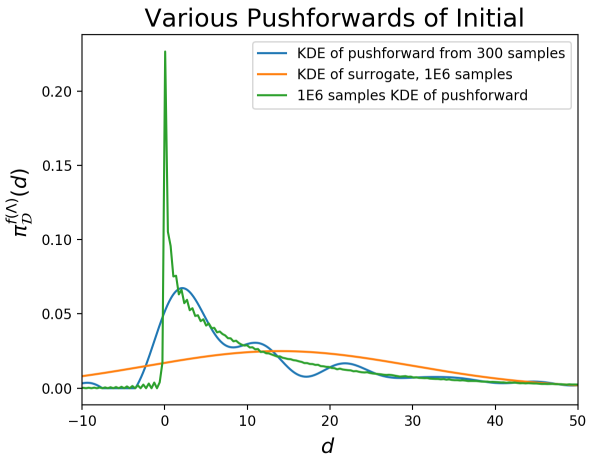


Figure 4.5: We show in this nonlinear example that the pushforward of $\pi_\Lambda^{\text{init}}$ is expensive to obtain via KDE and is inaccurately obtained using surrogates – even when trained on a million samples.

The trade-off between our approach and standard sampling approaches is illustrated in this example: we exchange the error from surrogates or KDE for *linearization* errors, which in other examples, may be more extreme.

4.4 Discussion

In our numerical examples with both linear and nonlinear models, we demonstrated the computational benefit in stepping in the AS instead of in the full variables in solving certain SIPs using DCI via

optimization, not dissimilar to what the authors find in Constantine *et al.* (2016) in the Bayesian setting and what we proposed here in our two-step method which accelerates Bayesian inversion. Step one – minimizing the data misfit and employing FAASTARS once possible – is identical in our prosposed Bayesian and DCI solution processes. Step two – which is not the same for the Bayesian and DCI approaches – involved regularizing using $\mathcal{A}$ and $\mathcal{I}$, which we have shown are equivalent to the different regularization techniques used in the optimization formulations of Bayesian inversion and DCI with linear f and Gaussian PDF's.

We proposed a two-step method in the case that f is nonlinear and found the theoretical MAP and MUD points and their corresponding covariances by performing our two step process with the nonlinear signal replacing the linear A in our objective functionals. We obtained a surrogate for f which allows for the approximation of MAP/MUD points and the covariances via the closed-form gradient of the surrogate. We also assume that all PDF's are Gaussian in our approach, which is not generally true for nonlinear problems. Thus, our solutions are Gaussian approximations to the true solutions, which may be non-Gaussian. Future work could involve considerations for more closely approximating solutions which may be multi-modal, likely via mixture models.

Since the hyperparameters in STARS are dimension-dependent, anytime an AS resolves the map well – which occurred in the above example by obvious design – we expect ASTARS in the exact active variables to converge more quickly than STARS in full variables, as in Chapter II and III. In numerical experiments, we observed that the quality of the estimated $\tilde{A}$ often negatively impacted step two of our proposed process, where inactive variables are reset to the mean of the initial density, due to incorrectly-found dimensions $\tilde{j}$ of $\tilde{\mathcal{A}}$. More upfront samples for higher-quality surrogates (used to form $\tilde{\mathcal{A}}$) would surely improve results, but are avoided in our setting, where we evaluate $\hat{f}$ as little as possible. In some applications, the MAP and MUD points we obtained may be acceptable; in other applications, they may not.

REFERENCES

Bardsley JM (2018). *Computational Uncertainty Quantification for Inverse Problems.* SIAM, Philadelphia, PA.

Beck A (2014). *Introduction to Nonlinear Optimization: Theory, Algorithms, and Applications with MATLAB.* SIAM, Philadelphia, PA.

Bertsekas DP (2016). *Nonlinear Programming.* Athena Scientific, Nashua, NH.

Butler T, Jakeman J, Wildey T (2018). "Combining Push-Forward Measures and Bayes' Rule to Construct Consistent Solutions to Stochastic Inverse Problems." *SIAM Journal on Scientific Computing*, **40**(2), A984–A1011.

Calliess JP (2017). "Lipschitz optimisation for Lipschitz interpolation." In *2017 American Control Conference (ACC 2017).* Seattle, WA, USA.

Carey V (2020). "active_subspaces _py3." https://github.com/variscarey/active_subspaces_py3.

Chen R, Wild SM (2015). "Randomized Derivative-Free Optimization of Noisy Convex Functions." ArXiv:1507.03332. Funded by the Deparetment of Energy. Unpublished paper.

Constantine, Dow, Wang (2014). "Active Subspace Methods in Theory and Practice: Applications to Kriging Surfaces." *SIAM Journal on Scientific Computing*, **36**(4), A1500–A1524.

Constantine, Eftekhari, Wakin (2015). "Computing Active Subspaces Efficiently with Gradient Sketching." In *2015 IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP).*

Constantine P (2020). "active_subspaces." https://github.com/paulcon/active_subspaces.

Constantine PG (2015). *Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies.* SIAM, Philadelphia, PA.

Constantine PG, Diaz P (2017). "Global sensitivity metrics from active subspaces." *Reliability Engineering & System Safety*, **162**, 1–13.

Constantine PG, Kent C, Bui-Thanh T (2016). "Accelerating Markov Chain Monte Carlo with Active Subspaces." *SIAM Journal on Scientific Computing*, **38**, A2779–A2805.

Doering CR, Gibbon JD, Holm DD, Nicolaenko B (1988). "Low-dimensional behaviour in the complex Ginzburg-Landau equation." *Nonlinearity*, **1**, 279–309.

Gorbunov E, Dvurechensky P, Gasnikov A (2019). "An Accelerated Method for Derivative-Free Smooth Stochastic Convex Optimization." ArXiv:1802.09022. Unpublished paper.

Hall J, Carey V (2020). "ASTARS." https://github.com/jordanrhall/ASTARS.

Hastie T, Tibshirani R, Friedman J (2001). *The Elements of Statistical Learning.* Springer Series in Statistics. Springer New York Inc., New York, NY, USA.

Karimi M, Karimi B (2017). "Linear and conic scalarizations for obtaining properly efficient solutions in multiobjective optimization." *Math Sci*, **11**, 319–325.

Kvasov, Sergeyev (2012). "Lipschitz gradients for global optimization in a one-point-based partitioning scheme." *Journal of Computational and Applied Mathematics*, **236**(16), 4042–4054.

Lea DJ, Allen MR, Haine TWN (2000). "Sensitivity analysis of the climate of a chaotic system." *Tellus A*, **52**(5), 523–532.

Moré JJ, Wild SM (2015). "Estimating Computational Noise." *SIAM Journal on Scientific Computing*, **33**(3), 1292–1314. doi:10.1137/100786125. Funded by the Department of Energy.

Moré JJ, Wild SM (2020). "ECNoise." https://www.mcs.anl.gov/~wild/cnoise/. Accessed Sep. 25, 2020.

Murray J (1989). *Mathematical Biology*. Springer-Verlag.

Nesterov Y (2005). "Smooth minimization of non-smooth functions." *Math. Program.*, **103**, 127–152.

Nesterov Y, Spokoiny V (2017). "Random Gradient-Free Minimization of Convex Functions." *Foundations of Computational Mathematics*, **17**, 527–566.

Papaioannou I, Ehre M, Straub D (2019). "PLS-based adaptation for efficient PCE representation in high dimensions." *Journal of Computational Physics*, **387**, 186–204.

Pilosov M (2020). *Computational Advances in Data-Consistent Inversion: Measure-Theoretic Methods for Improving Predictions*. Ph.D. thesis. University of Colorado Denver.

Queipo NV, Haftka RT, Shyy W, Goel T, Vaidyanathan R, Tucker PK (2005). "Surrogate-based analysis and optimization." *Progress in Aerospace Sciences*, **41**, 1–28.

Rosenbrock HH (1960). "An Automatic Method for finding the Greatest or Least Value of a Function." *The Computer Journal*, **3**, 175–184.

Rosipal R, Krämer N (2006). "Overview and recent advances in partial least squares." *Lecture Notes in Computer Science*, **3940**, 34–51.

Russi TM (2010). *Uncertainty Quantification with Experimental Data and Complex System Models*. Ph.D. thesis. University of California Berkeley.

Sauer T (2011). *Numerical Analysis*. 2nd edition. Addison-Wesley Publishing Company, USA.

Shin SJ, Wu Y, Zhang HH, Liu Y (2017). "Principal weighted support vector machines for sufficient dimension reduction in binary classification." *Biometrika*, **104**, 67–81.

Smith R (2013). *Uncertainty Quantification: Theory, Implementation, and Applications*. SIAM, Philadelphia, PA.

Stuart A (2010). "Inverse problems: A Bayesian perspective." *Acta Numerica*, **19**, 451–559.

Tarantola A (2005). *Inverse Problem Theory and Methods for Model Parameter Estimation*. SIAM, Philadelphia, PA.

Trefethen LN, Bau D (1997). *Numerical Linear Algebra*. SIAM, Philadelphia, PA.

Wang Q, Hu R, Blonigan P (2014). "Least Squares Shadowing sensitivity analysis of chaotic limit cycle oscillations." *Journal of Computational Physics*, **267**, 210–224.

Wildey T, Butler T, Jakeman J, Marvin B (2018). "Consistent Bayesian Inference with Push-forward Measures: Scalable Implementations and Applications." Slideshow, 2018 SIAM Annual Meeting, Portland, OR, USA. July 9-13. SAND2018-7239C.

Zhou X (2017). "Lipschitz continuous gradient." https://xingyuzhou.org/blog/notes/Lipschitz-gradient. Unpublished paper.

APPENDIX A

Chapter II Appendix

We present a handful of interesting but technical numerical results we observed in computations from Chapter II Examples 1-3. We also present an extra numerical result from Chapter II Example 1, where we allowed for multiplicative noise in $\hat{f}$.

1. Average Initial and Final $\tilde{j}$'s in Examples 1-3

We present the average of the initial and final average obtained dimensions $\tilde{j}$ of the approximated $\tilde{\mathcal{A}}$ obtained using FAASTARS from Examples 2 and 3 in Chapter II, without hyperparameter learning. Recall, 100 trials were performed in Example 2 (without learning) and 50 trials were performed in Example 3 (without learning). Recall that in Example 2, $\dim(\mathcal{A}) = j = 10$ (with 10 equally-active directions) but in Example 3, $j = 5$. We note that all trials in Example 1 start with fairly small $\tilde{j}$ (usually between 2 and 5) and finish with $\tilde{j} = j = 1$.

Table A.1: Average Initial and Final $\tilde{j}$ in Chapter II Examples 2 and 3

Example	**Average Initial $\tilde{j}$**	**Average Final $\tilde{j}$**	**True j**
Example 2	14.06	11.41	10
Example 3	34.46	5.56	5

Note that in both cases, the average $\tilde{j}$ is greatly reduced by the time termination is reached by using active subspace retraining in FAASTARS, described in Chapter II. However, both average $\tilde{j}$ are overestimated by the final iteration – more noticeably in Example 2 – indicating a slightly larger τ may be necessary. (One may choose to apply adaptive thresholding or active subcyling in this case, as well.)

2. Discussion and Example of Multiplicative Noise

Recall, we call the noise in $\hat{f}$ *multiplicative* when $\hat{f} = f(1 + \epsilon)$. We did not consider multiplicative noise in $\hat{f}$ in the theoretical nor numerical results presented in Chapter II. The STARS manuscript Chen and Wild (2015) does contain theoretical results regarding the convergence, complexity, and achievable accuracy of STARS in the case that $\hat{f}$ exhibits multiplicative noise. Extensions to those results to obtain theoretical statements (similar to those we derived and presented here in the additive noise case) are an item for future work.

However, we obtained numerical results in the multiplicative noise setting by making a trivial modification to ASTARS and FAASTARS to use the correct smoothing factor μ_k^* – see (1.3.4) in Chapter I. (The smoothing factor must be re-computed at each iteration, which is importantly scaled by the last evaluation of $\hat{f}$ due to the term $\epsilon\hat{f}$ appearing in evaluations of $\hat{f}$ with multiplicative noise.) Note that

the user only needs to turn on a logical flag in the case of multiplicative noise (i.e., set the flag equal to "`True`") for our implementations of STARS, ASTARS, and FAASTARS to use the correct smoothing factor in its computations.

We present the results of introducing multiplicative noise into Example 1 from Chapter II. No hyperparameter learning was used. In particular, we let $\hat{f}(\lambda;\xi) = (w^\top\lambda)^2(1+\epsilon(\xi))$ where $\epsilon \sim N(0, 1\times 10^{-12})$ (as in the no-learning case for this example). We used an initial start drawn from $N(0, I_P)$ not scaled by 10, as in Example 1 from Chapter II. We performed 100 trials. All other specifications were left unchanged. Finally, note that the ECNoise estimates to $\hat{\sigma}^2$ must also be scaled in the multiplicative setting (i.e., divided by the square of the first $\hat{f}$ evaluation taken – see Moré and Wild (2015)) which is also automatically handled in our code. (Though, we do not present hyperparameter learning here.)

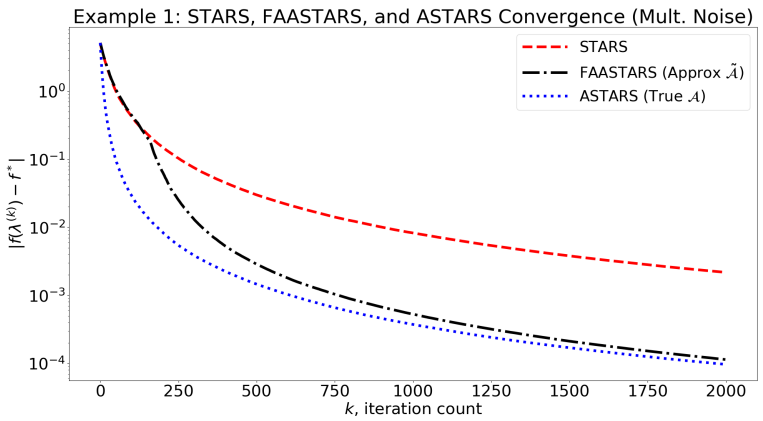


Figure A.1: We show the convergence of STARS, ASTARS, and FAASTARS using the objective function in Chapter II Example 1, but with multiplicative noise.

Notice that whereas STARS (and ASTARS and FAASTARS) converged in 800 iterations with additive noise, we see convergence is far from achieved even after 2000 iterations in the multiplicative noise case. We also notice that this rate appears nonlinear, whereas convergence rates in Chapter II Example 1 appear linear; also, interestingly, FAASTARS convergence mirrors ASTARS more closely than any of our examples. (Recall that we have even scaled down the initial iterate from what was used in Chapter II Example 1.) These results are no surprise; indeed, multiplicative noise introduces noise scaled by $\hat{f}(\lambda)$, which may be large, causing great magnification in these values. Future work will involve improving numerical performance of our methods and developing theoretical results for the case of multiplicative noise.

APPENDIX B

Chapter IV Appendix

We present the technical (and somewhat lengthy) details in verifying the presented form of the Bayesian posterior covariance, C_{post} in the case that f is nonlinear, which was presented in Chapter IV without a full justification.

1. Verifying C_{post} for Nonlinear f

We address the case when $\dim(\mathcal{A}) = j \leq D = \dim(\mathcal{D})$, and recalling $\nabla_{\mathcal{A}} f(\lambda^*) := \nabla f(\lambda^*) V_{\mathcal{A}}^\top$, we assume $\nabla_{\mathcal{A}} f(\lambda^*)$ has full rank. As before, let λ^* minimize $S(\lambda_{\mathcal{A}})$ and recall $S = S_1 + S_2$, where S_1 denotes the first two terms in $\tilde{S}$ (see (4.2.3)) (data misfit and regularization in $\mathcal{A}$) and S_2 denotes the last term in $\tilde{S}$ (regularization in $\mathcal{I}$, independent of f). For compactness, let $C_{\mathcal{A}}^{-1} := V_{\mathcal{A}} C_\Lambda^{-1} V_{\mathcal{A}}^\top$, the covariance in $\mathcal{A}$. Taylor expanding S_1 at λ^* and using the notation above yields

$$\begin{aligned}&(f(\lambda^*) + \nabla_{\mathcal{A}} f(\lambda^*)^\top (\lambda_{\mathcal{A}} - \lambda^*) - \bar{d}))^\top C_{\mathcal{D}}^{-1} (f(\lambda^*) + \nabla_{\mathcal{A}} f(\lambda^*)^\top (\lambda_{\mathcal{A}} - \lambda^*) - \bar{d}) + (\lambda_{\mathcal{A}} - \bar{\lambda}_{\mathcal{A}})^\top C_{\mathcal{A}}^{-1} (\lambda_{\mathcal{A}} - \bar{\lambda}_{\mathcal{A}}) \\ &=: S_{1,1}(\lambda_{\mathcal{A}}) + S_{1,2}(\lambda_{\mathcal{A}}),\end{aligned}$$

where we let $S_{1,1}$ denote the first term in the expansion of S_1 above and $S_{1,2}$ denote the second term of S_1 (expanded). Recognizing that the quadratic form $S_{1,2}(\lambda_{\mathcal{A}})$ can be rewritten as the inner product $(\lambda_{\mathcal{A}} - \bar{\lambda_{\mathcal{A}}}, \lambda_{\mathcal{A}} - \bar{\lambda_{\mathcal{A}}})_{C_{\mathcal{A}}^{-1}}$, we may add and subtract λ^* from both sides of the inner product to obtain

$$(\lambda_{\mathcal{A}} - \lambda^*, \lambda_{\mathcal{A}} - \lambda^*)_{C_{\mathcal{A}}^{-1}} + (\lambda^* - \bar{\lambda}_{\mathcal{A}}, \lambda_{\mathcal{A}} - \lambda^*)_{C_{\mathcal{A}}^{-1}} + (\lambda_{\mathcal{A}} - \bar{\lambda}_{\mathcal{A}}, \lambda^* - \bar{\lambda}_{\mathcal{A}})_{C_{\mathcal{A}}^{-1}} + (\lambda^* - \bar{\lambda}_{\mathcal{A}}, \lambda^* - \bar{\lambda}_{\mathcal{A}})_{C_{\mathcal{A}}^{-1}}. \quad (2.0.1)$$

At λ^*, the (active) gradient of $S(\lambda_{\mathcal{A}})$ is zero, so

$$\nabla_{\mathcal{A}} S_1(\lambda^*) = \nabla_{\mathcal{A}} f(\lambda^*) C_{\mathcal{D}}^{-1} (f(\lambda^*) - \bar{d}), \quad \nabla_{\mathcal{A}} S_2(\lambda^*) = C_{\mathcal{A}}^{-1} (\lambda^* - \bar{\lambda}_{\mathcal{A}}), \quad \text{and} \quad \nabla_{\mathcal{A}} (S_1(\lambda^*) + S_2(\lambda^*)) = 0. \quad (2.0.2)$$

Combining our linearization term with the inner products in (2.0.1) yields

$$\begin{aligned}&(\lambda_{\mathcal{A}} - \lambda^*)^\top \nabla_{\mathcal{A}} f(\lambda^*) C_{\mathcal{D}}^{-1} \nabla_{\mathcal{A}} f(\lambda^*)^\top (\lambda_{\mathcal{A}} - \lambda^*) + (\lambda_{\mathcal{A}} - \lambda^*)^\top C_{\mathcal{A}}^{-1} (\lambda_{\mathcal{A}} - \lambda^*) \\ &+ (\lambda_{\mathcal{A}} - \lambda^*)^\top \left(\nabla_{\mathcal{A}} f(\lambda^*) C_{\mathcal{D}}^{-1} (f(\lambda^*) - d) + C_{\mathcal{A}}^{-1} (\lambda^* - \bar{\lambda}_{\mathcal{A}}) \right) \\ &+ \left(C_{\mathcal{D}}^{-1} \nabla_{\mathcal{A}} f(\lambda^*)^\top (f(\lambda^*) - \bar{d}) + (\lambda^* - \bar{\lambda}_{\mathcal{A}}) C_{\mathcal{A}}^{-1} \right)^\top (\lambda_{\mathcal{A}} - \lambda^*) \\ &+ (f(\lambda^*) - \bar{d})^\top C_{\mathcal{D}}^{-1} (f(\lambda^*) - d) + (\lambda^* - \bar{\lambda}_{\mathcal{A}})^\top C_{\mathcal{A}}^{-1} (\lambda^* - \bar{\lambda}_{\mathcal{A}}).\end{aligned} \quad (2.0.3)$$

The second and third lines in (2.0.3) are zero from the gradient conditions in (2.0.2), while the fourth line is independent of $\lambda_{\mathcal{A}}$. In particular, the fourth line is equal to $S(\lambda^*)$; i.e., it is a constant with respect to $\lambda_{\mathcal{A}}$. We have shown

$$\begin{aligned}\pi_{\Lambda}^{\text{post}} &\sim \exp\left(-\frac{1}{2}(S_1(\lambda)+S_2(\lambda))\right)\\ &= \exp\left(-\frac{1}{2}\left((\lambda_{\mathcal{A}}-\lambda^*)^\top\left(\nabla_{\mathcal{A}}f(\lambda^*)C_{\mathcal{D}}^{-1}\nabla_{\mathcal{A}}f(\lambda^*)^\top+C_{\mathcal{A}}^{-1}\right)(\lambda_{\mathcal{A}}-\lambda^*)+(\lambda_{\mathcal{I}}-\lambda^*)^\top C_{\mathcal{I}}^{-1}(\lambda_{\mathcal{I}}-\lambda^*)\right)\right)\\ &\sim \mathcal{N}(\lambda^*, C_{\text{post}}),\end{aligned}$$

where

$$C_{\text{post}}^{-1} = \nabla_{\mathcal{A}}f(\lambda^*)C_{\mathcal{D}}^{-1}\nabla_{\mathcal{A}}f(\lambda^*)^\top + C_{\mathcal{A}}^{-1} + C_{\mathcal{I}}^{-1}, \tag{2.0.4}$$

and $C_{\mathcal{I}}^{-1} := V_{\mathcal{I}}C_{\Lambda}^{-1}V_{\mathcal{I}}^\top$, the covariance in $\mathcal{I}$. (2.0.4) is equivalent to (4.2.12) in Chapter IV.