On the Effects of Noise on Parameter Identification Optimization Problems

Kay E. Vugrin

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Mathematics

Jeffrey T. Borggaard, Chair
Eugene M. Cliff
Terry Herdman
Traian Iliescu
Shu-Ming Sun

April 4, 2005
Blacksburg, Virginia

Keywords: Optimization, Numerical Noise, Parameter Identification, Nonlinear Least–Squares, Inverse Problem, Nelder–Mead Algorithm, Shuffled Complex Evolution Method

On the Effects of Noise on Parameter Identification Optimization Problems

by
Kay E. Vugrin

ABSTRACT

The calibration of model parameters is an important step in model development. Commonly, system output is measured, and model parameters are iteratively varied until the model output is a good match to the measured system output. Optimization algorithms are often used to identify the model parameter values. The presence of noise is difficult to avoid when physical processes are used to calibrate models due to measurement error, model structure error, and errors arising from numerical techniques and approximate solutions. Our study focuses on the effects of noise in parameter identification optimization problems. We generate six test problems, including five perturbations of a smooth problem. A previously studied groundwater parameter identification problem serves as our seventh test problem. We test the Nelder–Mead Algorithm, a combination of the Nelder–Mead Algorithm and Simulated Annealing, and the Shuffled Complex Evolution Method on these test problems. Comparison of optimization results for these problems reveals the effects of noise on optimization performance, including an increase in fitness values and a decrease in the number of fit evaluations. We vary the values of the internal algorithmic parameters to determine the effects of different values and present numerical results that indicate that changing the values of the algorithmic parameters can cause profound differences in optimization results for all three algorithms. A variation of the generally accepted parameter values for the Nelder–Mead Algorithm is recommended, and we determine that the Nelder–Mead/Simulated Annealing Hybrid and Shuffled Complex Evolution Method are too problem dependent for general recommendations for parameter values. Finally, we prove new convergence results for the Nelder–Mead/Simulated Annealing Hybrid in both smooth and noisy cases.

Sandia is a multiprogram laboratory operated by Sandia Corporation for the Department of Energy's National Nuclear Security Administration under Contract DE-AC04-94AL85000. This research is funded in part by WIPP programs administered by the U.S. Department of Energy.

Dedication

I dedicate this dissertation to members of two generations who have inspired me to appreciate the past and to look forward to the future:

In loving memory of my grandfather, Kenneth Phillip White, Senior, for his lifelong examples of dedication to family and perseverance in the face of difficulty;

and,

To my daughter, Melina Kay Vugrin, in hope and anticipation of the marvelous experiences ahead of her.

Acknowledgments

This dissertation would not have been possible without the support of many individuals. The people at ICAM and the Mathematics Department at Virginia Tech have all been very pleasant to work and learn with, and I thank all of them for a wonderful graduate experience. I especially appreciate the professors that taught my courses and encouraged me to continue learning. I thank my committee members Drs. Gene Cliff, Terry Herdman, Traian Iliescu, and Shu-Ming Sun for improving my dissertation and providing invaluable technical guidance. I also thank them for their warm support over the past five years. I thank my advisor, Dr. Jeff Borggaard, for five years of teaching, mentoring, and challenging me. Thank you for looking out for my future, for working so hard for my benefit, and for many hours of entertaining debates. I cannot imagine a better advisor.

I thank Misty Linkous, Amy Rein, and Margaret Vugrin for locating the information I needed during the course of this dissertation. I thank Hannah Swiger and the Graduate School staff at Virginia Tech for easing the difficulties of a long–distance student. I also thank Dan Eno and Tom Kirchner for discussing statistics with me, and thank Tom for proofreading.

Several Sandia employees and contractors have been instrumental in the completion of this project. I thank Randy Roberts and Rick Beauheim for providing me with a very interesting and enjoyable project for my first job, and for their patience and forbearance with my many questions. I thank John Avis for his patient support, and for his amazingly accurate and fast programming abilities. I particularly thank my managers Dave Kessel, Mark Rigali, and Paul Shoemaker for their complete support of this project in both word and deed. This dissertation is a result of their willingness to combine Sandia's priorities with my graduate work, and I am grateful for the opportunities they provided. I also thank Sandia National Laboratories Tuition Assistance Program for support during the completion of this project.

I thank my friends and family members for their support and encouragement throughout my years as a student. My parents Ken and Barbara have always encouraged my quest for learning, and I believe my love for education is a direct result of their encouragement from a very young age. My greatest debt of thanks, however, belongs to my husband Eric. Thank you for encouraging me in all of my goals. Thank you for listening when I needed to talk, for supporting me in the decisions I made, and for disregarding those decisions when I got them wrong. In short, thank you for being the perfect husband and father.

Contents

1 Introduction **1**

2 The Inverse Problem and Noisy Parameter Identification Optimization Problems **4**

2.1 Inverse Problem 5

2.2 Nonlinear Least–Squares Optimization Problem 6

2.3 Optimization in the Presence of Noise 8

2.3.1 Sources of Noise in Parameter Identification Problems 8

2.3.2 Optimization Algorithms in the Presence of Noise 8

3 Optimization Algorithms and Theory **13**

3.1 The Nelder–Mead Downhill Simplex Method 13

3.1.1 The Nelder–Mead Algorithm 14

3.1.2 Convergence Behavior For Non–Noisy Functions 15

3.1.3 Convergence Behavior for Noisy Functions 19

3.2 The Nelder–Mead and Simulated Annealing Hybrid 20

3.2.1 The Nelder–Mead and Simulated Annealing Hybrid Algorithm 21

3.2.2 Convergence Behavior for Non–Noisy Functions 22

3.2.3 Convergence Behavior for Noisy Functions 28

3.3 The Shuffled Complex Evolution Method 32

4 Test Problems **36**

4.1 Polynomial Test Problems 36

4.1.1 Test Problem 1: Fifth Degree Polynomial Model 37
4.1.2 Test Problem 2: Third Degree Polynomial Model 39
4.2 Groundwater Parameter Identification Test Problem 40
4.2.1 Governing Equations 40
4.2.2 Numerical Simulation Techniques 42
4.2.3 Test Problem 3: Groundwater Parameter Identification Problem . . . 43

5 Optimization Testing Methods **46**
5.1 Comparative Performance Analysis 46
5.2 Nelder–Mead Algorithm 47
5.2.1 Space Sampling Technique 47
5.2.2 Testing Procedure 48
5.3 Nelder–Mead/Simulated Annealing Hybrid 49
5.4 Shuffled Complex Evolution Method 50

6 Numerical Results: The Nelder–Mead Algorithm **52**
6.1 Test Problem 1 52
6.1.1 Test Problem 1*a* 52
6.1.2 Test Problem 1*b* 53
6.1.3 Test Problem 1*c* 55
6.1.4 Effects Due to Noise 56
6.2 Test Problem 2 58
6.2.1 Test Problem 2*a* 58
6.2.2 Test Problem 2*b* 60
6.2.3 Test Problem 2*c* 61
6.2.4 Effects Due to Noise 63
6.3 Test Problem 3 64
6.3.1 Parameter Selection 64
6.3.2 Numerical Results 65

6.4 Selecting NMA Parameters for Noisy Optimization 66

7 Numerical Results: The Nelder–Mead/Simulated Annealing Hybrid 69

7.1 Test Problem 3 69

7.1.1 Cooling Schedule 1 69

7.1.2 Cooling Schedule 2 71

7.2 Comparison of Cooling Schedules 72

8 Numerical Results: The Shuffled Complex Evolution Method 73

8.1 Test Problem 1 73

8.1.1 Test Problem 1*a* 73

8.1.2 Test Problem 1*b* 75

8.1.3 Test Problem 1*c* 76

8.1.4 Effects Due to Noise 78

8.2 Test Problem 2 79

8.2.1 Test Problem 2*a* 80

8.2.2 Test Problem 2*b* 81

8.2.3 Test Problem 2*c* 82

8.2.4 Effects Due to Noise 83

8.3 Test Problem 3 84

8.3.1 Parameter Selection 84

8.3.2 Numerical Results 85

8.4 Selecting SCEM Parameters for Noisy Optimization 86

9 Comparison of Algorithms 88

9.1 Test Problem 1 88

9.2 Test Problem 2 89

9.3 Test Problem 3 90

9.4 Selecting an Algorithm 91

10 Conclusions and Future Work **93**

Bibliography **96**

A Nelder Mead Parameter Samples **107**

B Shuffled Complex Evolution Parameter Samples **110**

C Nelder Mead Numerical Results **113**

D Shuffled Complex Evolution Numerical Results **129**

Vita **144**

List of Figures

3.1 The Nelder–Mead Downhill Simplex Algorithm 15
3.2 The Shuffled Complex Evolution Method 34
3.3 The Competitive Complex Evolution Strategy 35

4.1 Fifth Order Polynomial and Matching Points 38
4.2 Fifth Order Polynomial and Matching Points With Low Level of Noise . . . 38
4.3 Fifth Order Polynomial and Matching Points With High Level of Noise . . . 39
4.4 Test Problem 3: Measured Pressure Values 44
4.5 Test Problem 3: Measured Flowrate Values 45
4.6 Test Problem 3: Matching Points 45

List of Tables

5.1 Nelder–Mead Parameter Ranges 47
5.2 Starting Points for Test Problem 1 48
5.3 Starting Points for Test Problem 2 49
5.4 NMA/SA: Parameter Samples for Cooling Schedule 1 50
5.5 NMA/SA: Parameter Samples for Cooling Schedule 2 50
5.6 SCEM Parameter Values 51

6.1 Test Problem 1*a*: Best NMA Parameter Combinations 53
6.2 Test Problem 1*a*: Worst NMA Parameter Combinations 53
6.3 Test Problem 1*b*: Best NMA Parameter Combinations 54
6.4 Test Problem 1*b*: Worst NMA Parameter Combinations 54
6.5 Test Problem 1*b*: Parameter Combinations Yielding Lowest True Fit Values for the NMA 55
6.6 Test Problem 1*c*: Best NMA Parameter Combinations 56
6.7 Test Problem 1*c*: Worst NMA Parameter Combinations 56
6.8 Test Problem 1: Comparison of Noise Effects on the NMA 57
6.9 Test Problem 2*a*: Best NMA Parameter Combinations 59
6.10 Test Problem 2*a*: Worst NMA Parameter Combinations 59
6.11 Test Problem 2*b*: Best NMA Parameter Combinations 60
6.12 Test Problem 2*b*: Worst NMA Parameter Combinations 61
6.13 Test Problem 2*c*: Best NMA Parameter Combinations 62
6.14 Test Problem 2*c*: Worst NMA Parameter Combinations 62

6.15 Test Problem 2: Comparison of Noise Effects on the NMA 63
6.16 Test Problem 3: NMA Results 66
6.17 Competitive NMA Parameter Combinations 67
6.18 Poor NMA Parameter Combinations 68

7.1 Test Problem 3: NMA/SA and Cooling Schedule 1 Results 70
7.2 Test Problem 3: NMA/SA and Cooling Schedule 2 Results 71

8.1 Test Problem 1*a*: Best SCEM Parameter Combinations 74
8.2 Test Problem 1*a*: Worst SCEM Parameter Combinations 74
8.3 Test Problem 1*b*: Best SCEM Parameter Combinations 75
8.4 Test Problem 1*b*: Worst SCEM Parameter Combinations 76
8.5 Test Problem 1*b*: Parameter Combinations Yielding Lowest True Fit Values for the SCEM 76
8.6 Test Problem 1*c*: Best SCEM Parameter Combinations 77
8.7 Test Problem 1*c*: Worst SCEM Parameter Combinations 77
8.8 Test Problem 1*c*: Parameter Combinations Yielding Lowest True Fit Values for the SCEM 78
8.9 Test Problem 1: Comparison of Noise Effects on the SCEM 79
8.10 Test Problem 2*a*: Best SCEM Parameter Combinations 80
8.11 Test Problem 2*a*: Worst SCEM Parameter Combinations 80
8.12 Test Problem 2*b*: Best SCEM Parameter Combinations 81
8.13 Test Problem 2*b*: Worst SCEM Parameter Combinations 81
8.14 Test Problem 2*c*: Best SCEM Parameter Combinations 82
8.15 Test Problem 2*c*: Worst SCEM Parameter Combinations 83
8.16 Test Problem 2: Comparison of Noise Effects on the SCEM 84
8.17 Test Problem 3: SCEM Results 85
8.18 Competitive SCEM Parameter Values 87
8.19 Poor SCEM Parameter Values 87

9.1 Test Problem 1: Comparison of the NMA and SCEM 89

9.2 Test Problem 2: Comparison of the NMA and SCEM 90
9.3 Test Problem 3: Comparison of the NMA, NMA/SA Hybrid, and SCEM . . 91

A.1 Nelder–Mead Parameter Samples 107

B.1 SCEM Parameter Samples 110

C.1 NMA Numerical Results: Test Problem 1a 113
C.2 NMA Numerical Results: Test Problem 1b 116
C.3 NMA Numerical Results: Test Problem 1c 118
C.4 NMA Numerical Results: Test Problem 2a 120
C.5 NMA Numerical Results: Test Problem 2b 123
C.6 NMA Numerical Results: Test Problem 2c 125

D.1 SCEM Numerical Results: Test Problem 1a 129
D.2 SCEM Numerical Results: Test Problem 1b 131
D.3 SCEM Numerical Results: Test Problem 1c 134
D.4 SCEM Numerical Results: Test Problem 2a 136
D.5 SCEM Numerical Results: Test Problem 2b 139
D.6 SCEM Numerical Results: Test Problem 2c 141

Chapter 1

Introduction

Parameter identification is a fundamental process in the preparation of models for predictive calculations. One popular technique for the solution of nonlinear parameter identification problems is the application of optimization algorithms to the pertinent inverse problems. Model parameters are iteratively varied until the model output "fits" or matches an experimentally collected data set. The objective function for optimization is a metric quantifying the difference between simulated and measured results, and the optimized variables are the model parameters. The iterative optimization process is challenged by mathematical issues inherent to inverse problems, such as non–uniqueness of optimal parameters and possible numerical instability. However, there is another potential complication for optimization algorithms when applied to parameter identification problems: the presence of noise.

Noise can affect parameter identification problems through several possible entry points. Data collection methods are frequently corrupted by noise. The model used to represent the physical system under study may not completely describe the system. Thus, the discrepancy between the model's predictions and the true system response is a source of error or noise. Finally, numerical approximation techniques can add noise to the solution of the model, thereby adding noise to the optimization objective function. Numerical approximations to differential equations are often quite noisy. Adaptive methods, partial convergence, and stabilization strategies are all useful techniques in approximating solutions to differential equations, but these approaches create noise and difficulties for many optimization algorithms [22]. We denote the noise generated by numerical approximations as *numerical noise*. In this study, we focus on the difficulties encountered by optimization algorithms when they are used to solve noisy parameter identification problems.

In Chapter 2, we first discuss inverse problems and formulate the optimization problem used in parameter identification problems. We discuss the causes of noise in parameter identification problems. We outline three broad categories of optimization algorithms. Our investigations are limited to three specific algorithms: the Nelder–Mead Algorithm, a combination of the Nelder–Mead Algorithm and simulated annealing theory, and the Shuffled

Complex Evolution Method. In Chapter 3, we describe each algorithm, including the rules and control parameters that dictate algorithm behavior. We also summarize existing convergence results both for smooth and noisy cases. We present and prove several new theorems related to convergence behavior of the Nelder–Mead/Simulated Annealing hybrid algorithm.

In Chapter 4, we introduce seven test problems. Six of these test problems are based on analytically generated data sets. A nominal data set is generated by evaluating a fifth degree polynomial at 101 evenly spaced points. Two additional data sets are generated by adding normally distributed random data to the nominal data at two different standard deviation values. We attempt to match these three data sets through solution of the inverse problem. In the first three test problems, each of the three data sets is modeled with a fifth degree polynomial. The parameters that we attempt to identify are the polynomial coefficients. For the next three test problems, we model each of the three data sets with a third degree polynomial. A third degree polynomial is unable to embody the curvature of a fifth degree polynomial. These test problems reflect a common reality– many numerical models are unable to accurately represent the real world. Our final test problem is a groundwater parameter identification problem that contains noise from all three possible sources. The data set to be matched is a previously examined set of experimentally collected data.

Chapter 5 outlines our testing techniques for each of the three algorithms. We design our numerical study to shed light on the answers to the following three questions:

1. How does the presence of noise affect each of the optimization algorithms?
2. How do different values for optimization algorithm control parameters affect optimization, both with and without noise?
3. Do algorithmic parameter values need to be adjusted in order to better accommodate noisy optimization problems?

In order to answer these questions, we apply the Nelder–Mead Algorithm and Shuffled Complex Evolution Method to the first six test problems. We test 81 different combinations of optimization algorithm parameters for each of the algorithms. A subset of these 81 parameter samples for each algorithm is applied to the groundwater identification test problem. The Nelder–Mead/Simulated Annealing hybrid is applied to the groundwater identification test problem using two different cooling schedules and eighteen algorithmic parameter samples. Chapters 6 through 8 discuss the results of our optimization runs, and Chapter 9 is a comparison of the three algorithms. Through comparison of our generated test problems, we are able to isolate some of the effects of noise on the optimization algorithms. We find that adjustment of algorithmic parameters has a profound effect on all three algorithms. Although we are able to make some recommendations for choosing algorithmic parameters for the Nelder–Mead algorithm, we find that performance of the Nelder–Mead/Simulated Annealing hybrid and Shuffled Complex Evolution Method are too problem dependent to

make general recommendations. Finally, we present our conclusions and a brief discussion of future work.

Chapter 2

The Inverse Problem and Noisy Parameter Identification Optimization Problems

Engineering and scientific disciplines frequently use models to represent and predict physical processes. Often, these models are developed based on observations of the process under certain conditions. For example, a model describing the level of medication in the bloodstream may be based on measured concentrations at different time intervals after exposure. Or, a model of the cooling schedule of a heated metal could be based on temperature readings at different time intervals [10]. Development of these types of models is a complicated process. We present the following definitions from [118] to aid in understanding of the model development process.

Definition 2.0.1 System parameters *characterize the geometry and/or physical nature of the system.*

Definition 2.0.2 Subsidiary conditions *consist of initial conditions, which define the initial state of the system, and boundary conditions, which define the system's relationships of exchanging quantities of interest with its neighboring systems.*

Definition 2.0.3 Control variables *represent the excitation to the system. Control variables may appear in the subsidiary conditions.*

Definition 2.0.4 State variables *describe the state of the system. These variables are often measurable.*

Model development requires four steps [70, 119]. The first step is identification of the relevant system parameters. Ideally, these parameters will accurately describe the system in its entirety. The second step is determination of the physical principles that govern system

behavior. These principles specify the relationship between system parameters, subsidiary conditions, and control variables (input) and the state variables (output). The third step is determination of the values of the system parameters. This step is known as *parameter identification.*

Definition 2.0.5 Parameter identification *is the problem of determining model parameters from observed system states and other available information [118].*

When applied in an iterative manner, these three steps comprise model calibration. According to [118], "model calibration involves adjustment of the model structure and model parameters of a simulation model simultaneously or sequentially so as to make the input–output relation of the model fit any observed excitation–response relation of the real system." When model calibration is complete, the fourth and final step of model development is verification and validation of the model. After all four steps are completed, the model can be used to predict system behavior (values of state variables) for different combinations of system parameters, subsidiary conditions, and control variables.

When the system parameters, subsidiary conditions, and control variables have known values, solution of the model for the value of the state variables is the so–called *forward problem* [118]. The forward problem can be solved directly using analytical or numerical methods. Solutions of the forward problem predict system behavior under differing initial conditions. Thus, the information provided through solution of a properly calibrated forward problem can be used to predict system behavior.

2.1 Inverse Problem

System parameters generally cannot be measured directly, so another process is needed to determine their appropriate values. State variables can be measured directly from the physical process. After state variables are measured, the parameter values that reproduce the observed data can be found [118].

Parameter identification is an inverse problem. The observed system state and appropriate model structure are known, and the problem is to find the parameter values that best approximate the observed state. Although state variables can be calculated directly using known parameter values, unknown parameter values can only be inferred from state variables. An iterative trial and error process is required to determine parameter values [119].

Inverse problems are challenging for several reasons. The existence of a solution is not guaranteed, particularly if the observed data contains errors or if the model is grossly incorrect [119]. It is rare that any parameter set can exactly match given data. Additionally, the parameter values that can give rise to a given state are generally not unique. Oftentimes, not enough field data has been collected to determine a unique value for each parameter.

Finally, stability is also a concern for inverse problems. The final solution must be evaluated to ensure continuous dependence upon the inputs [118]. Sensitivity analysis can be applied to inverse problems to evaluate whether or not a problem is stable.

A popular method for parameter identification is to formulate the problem as an optimization problem. Although there are several possible objective functions, the most common is the least–squares function [83]. If $\mathbf{x} \in \mathbb{R}^n$ represents parameter values, $\mathbf{d} \in \mathbb{R}^m$ represents the measured state variables, and $\mathbf{y}(\cdot) : \mathbb{R}^n \rightarrow \mathbb{R}^m$ represents the simulated solution to the forward problem for parameter values $\mathbf{x}$, then

$$f(\mathbf{x}) = \sum_{j=1}^{m} (y_j(\mathbf{x}) - d_j)^2 \tag{2.1}$$

is the least–squares objective function. This expression quantifies the fit between the simulated state variables and the measured state variables. If P represents the set of all possible parameter values, the optimization problem for parameter identification is to find some $\mathbf{x}_* \in P$ such that

$$f(\mathbf{x}_*) \leq f(\mathbf{x}) \quad \text{for all} \quad \mathbf{x} \in P. \tag{2.2}$$

Then, the solution $\mathbf{x}_*$ is a candidate for the true system parameters. Ideally, there is some $\mathbf{x}_* \in P$ such that $f(\mathbf{x}_*) = 0$, implying that the simulation results exactly match the observed data. In practice, problems using real data rarely meet this zero–residual criteria [118].

The presence of noise in the inverse problem can cause difficulties for the optimization process. This issue is discussed in detail in Section 2.3. In data fitting applications where the objective function has the form of equation (2.1), the number of data points m is usually much greater than the number of variables n. Thus, the nonlinear least–squares problem can be viewed as an overdetermined system of equations [44]. Because more observation points exist than parameters, there are more equations than unknowns. If the optimal point $\mathbf{x}_*$ exists, it may not be unique, thus implying the existence of many combinations of parameters that result in the lowest observed value for f [118]. The presence of infinite global minima can complicate optimization decisions during each iteration, termination behavior, and performance analysis.

2.2 Nonlinear Least–Squares Optimization Problem

The general optimization problem Σ is as follows:

Σ: Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$.
Find $\mathbf{x}_* \in \mathbb{R}^n$ such that

$$f(\mathbf{x}_*) \leq f(\mathbf{x}) \quad \text{for all} \quad \mathbf{x} \in \mathbb{R}^n. \tag{2.3}$$

The following notation is used in order to facilitate readability. For any vector or matrix G and subscript b, G_b indicates $G(\mathbf{x}_b)$. Further, unless specified otherwise, all vector norms in this paper refer to the l^2 norm. That is, if $\mathbf{x} \in \mathbb{R}^n$,

$$\|\mathbf{x}\| = \|\mathbf{x}\|_2 = \left(\sum_{i=1}^{n} x_i^2 \right)^{\frac{1}{2}}. \tag{2.4}$$

Additionally, all matrix norms refer to the induced l^2 norm. That is, if $A \in \mathbb{R}^{m \times n}$, then

$$\|A\| = \|A\|_2 = \sup_{\|\mathbf{x}\|_2 = 1} \|A\mathbf{x}\|. \tag{2.5}$$

In a nonlinear least–squares optimization problem, the objective function has a specific structure [94]. The objective function has the form

$$f(\mathbf{x}) = \frac{1}{2} \sum_{j=1}^{m} r_j^2(\mathbf{x}), \tag{2.6}$$

where $\mathbf{x} \in \mathbb{R}^n$ and r_j is a continuous nonlinear mapping from $\mathbb{R}^n$ to $\mathbb{R}$ for each value of j. The functions r_j are called residual functions and have the form indicated in equation (2.1) in data–fitting applications. Clearly, $f : \mathbb{R}^n \to \mathbb{R}$, and $f(\mathbf{x})$ is always nonnegative. As outlined in [44], the residual functions can be expressed in the vector form $R(\mathbf{x}) = (r_1(\mathbf{x}), r_2(\mathbf{x}), \ldots, r_m(\mathbf{x}))^T$. Thus, $R : \mathbb{R}^n \to \mathbb{R}^m$, and the objective function f can be rewritten as

$$f(\mathbf{x}) = \frac{1}{2} R(\mathbf{x})^T R(\mathbf{x}) = \frac{1}{2} ||R(\mathbf{x})||^2. \tag{2.7}$$

When each r_j is twice continuously differentiable, the gradient and Hessian of f can be expressed in terms of the first and second derivatives of r_j. The first derivative of $R(\mathbf{x})$ is called the Jacobian of R. The Jacobian is an $m \times n$ matrix defined by

$$J_{ij}(\mathbf{x}) = \frac{\partial r_j(\mathbf{x})}{\partial x_i} \quad \text{for} \quad i = 1, \ldots n, \quad j = 1, \ldots m. \tag{2.8}$$

The gradient of f can be expressed in terms of J. That is,

$$\nabla f(\mathbf{x}) = \sum_{j=1}^{m} r_j(\mathbf{x}) \nabla r_j(\mathbf{x}) = J(\mathbf{x})^T R(\mathbf{x}). \tag{2.9}$$

The Hessian matrix of f can also be expressed in terms of the Jacobian of R:

$$\nabla^2 f(\mathbf{x}) = \sum_{j=1}^{m} \nabla r_j(\mathbf{x}) \nabla r_j(\mathbf{x})^T + \sum_{j=1}^{m} r_j(\mathbf{x}) \nabla^2 r_j(\mathbf{x}) = J(\mathbf{x})^T J(\mathbf{x}) + S(\mathbf{x}) \tag{2.10}$$

where

$$S(\mathbf{x}) = \sum_{j=1}^{m} r_j(\mathbf{x}) \nabla^2 r_j(\mathbf{x}). \tag{2.11}$$

When $f(\mathbf{x})$ has the form in equation (2.6), Σ is a nonlinear least–squares optimization problem. The nonlinear least–squares optimization problem (NLSOP) has a unique feature. Since $f(\mathbf{x}) \geq 0$ for all $\mathbf{x} \in \mathbb{R}^n$, $f(\mathbf{x}_*) \geq 0$. Therefore, if $f(\mathbf{x}) = 0$, then $\mathbf{x}$ must be a global minimizer. Knowledge of the value of f at the global minimum/minima can be very useful during theoretical investigations and determination of termination criteria.

2.3 Optimization in the Presence of Noise

2.3.1 Sources of Noise in Parameter Identification Problems

Parameter identification is an inherently noisy process. There are several unavoidable sources of error, including observation error, model structure error, and forward solution error [118]. *Observation error* can arise from inconsistencies in data collection, equipment malfunction or precision limitations, and human error. Additionally, the number and distribution of the observed data points will affect the accuracy of any parameters calculated using these data points. Observation error is generally assumed to have a random distribution [38]. *Model structure error* occurs when the mathematical model does not completely represent the physics of the system under study. Most real systems must be modeled through the use of simplifying assumptions, since the physical realities are often too complex to model exactly. Additionally, modelers may not know every feature of the system under study, so certain characteristics may be overlooked altogether. The model structure error cannot be quantified or bounded and is usually the dominating error term [12, 118]. Finally, *forward solution error* occurs when a numerical method is used to solve the forward problem. The numerical simulation is not an exact solution to the model, particularly when time constraints dictate the level of accuracy in the solution [118]. Generally, the error from the simulation of the forward problem can be bounded, though the magnitude can vary for each $\mathbf{x}$ value [22]. Due to these unavoidable errors, the results of the inverse problem must be carefully analyzed. In particular, if a model can be calibrated using one set of data and then *verified* by comparing its results against another set of data, modelers can have more confidence in the model and the calculated parameters [59].

2.3.2 Optimization Algorithms in the Presence of Noise

Though parameter identification optimization problems are subject to several unavoidable sources of noise, any optimization problem that depends upon numerical approximation can

also be prone to the presence of noise. Optimization in the presence of numerical noise is a popularly studied topic that merits further discussion, even in the context of an investigation of other types of noise. Many practical problems depend upon numerical approximation techniques in order to evaluate the objective function [23]. For instance, conduction and phase change problems [98], aerodynamic design problems [16, 18, 19, 25, 57], fluid modeling problems [22, 26, 28, 127, 128, 129, 133], and forced convection problems [21] are all examples of optimization problems where the objective function depends upon numerical approximations to complicated differential equations. Examples found in [64, 68, 116] are optimization problems that combine numerical solutions to differential equations with the least–squares objective function. Techniques such as mesh adaptation and adaptive time stepping strategies and phenomena such as partial convergence cause noise in the objective function [22]. Because the numerical approximations are not exact solutions to the differential equations, the true value of the objective function is not known. Additionally, this noise can cause spurious local minima and discontinuities [22, 133].

Often, accuracy of the numerical approximation and objective function evaluations can be improved through additional computational effort [22, 26, 28, 133]. However, there are many applications that require hours or days of computational time or manpower [50, 51]. The amount of time an analyst is willing to expend on objective function evaluations will cause a practical limit on the accuracy of objective function evaluations. Since noise cannot be eradicated in every case, optimization algorithms must be developed or modified in order to improve their performance in the presence of noise. A few examples of algorithms designed specifically for noisy optimization can be found in [50, 51, 52, 131].

Although there are many sources of noise, we focus on sources that greatly impact parameter identification problems. Specifically, we investigate the effects of errors in the matching data, and the effects of inadequate models. We also indirectly consider the impact of inexact numerical techniques. However, Sachs contends that errors introduced by various numerical approximations (i.e. finite difference techniques, finite element techniques, and quadrature) "can be interpreted as a perturbation of the data of the problem" [108]. Thus, the results of analysis of the effects of perturbations in problem data may be representative of other sources of noise.

Gradient–Based Methods in the Presence of Numerical Noise

Gradient–based optimization methods are characterized by the use of first and/or second derivatives to identify search directions. Newton's method and the steepest descent method are two of the most fundamental gradient–based methods. These methods have crippling convergence issues, however. Newton's method is not convergent for arbitrary starting points. Convergence results depend upon a starting point that is "close enough" to the minimum. The steepest descent method is known for "zigzagging" as the minimum is approached, thus extending the number of iterations required for convergence [11]. More effective gradient–

based methods include the Levenberg–Marquardt Method [84, 88], the trust–region method [35], the Davidon–Fletcher–Powell Method [56], and the Broyden–Fletcher–Goldfarb–Shanno updates [24, 55, 65, 113]. Many optimizers have attempted to develop new gradient–based algorithms specifically to exploit the structure of equations (2.9) and (2.10) [41, 42, 43, 44, 45, 68, 111].

Numerical noise in the objective function causes problems that are unique to gradient–based methods. Specifically, the gradient of the objective function might be approximated based on corrupted function values. The gradient approximation is usually a far worse approximation to the true gradient than the objective function approximation is to the true objective function. The gradient approximation of a noisy function can have the wrong value by many orders of magnitude and can even have the wrong sign, causing the algorithm to search in non–decrease directions with large step sizes [22, 133]. Numerical noise can cause many points to have a gradient value equal to zero, and gradient–based algorithms cannot distinguish these points from true global or local minima. If the noise causes discontinuities in the objective function, gradient approximations are not possible at these points. Incorrect or nonexistent gradient information can invalidate convergence theory and behavior that apply in the smooth case [108]. When convergence can be proven for noisy gradient–based optimization, more stringent assumptions are often required than in the smooth case [32, 33, 53, 72].

The method of gradient approximation can be varied in order to improve optimization algorithm performance for noisy functions. The finite difference method is very common [17], but when it is used to approximate the gradient of a noisy function, the gradient itself is very noisy and possibly discontinuous. Furthermore, the magnitude and sign of the finite difference gradient approximation depend upon the step size used [22, 62, 64, 133]. A trust–region method using finite difference gradient approximations has been shown to converge to a point farther from the true minimum than the starting point when applied to a noisy shape optimization problem [22, 133]. Other possible gradient approximation techniques include direct differentiation [120, 121], the discrete sensitivity system [26, 27], adjoint variable methods [75], automatic differentiation [64], and the hybrid sensitivity equation method [116]. The continuous sensitivity equation method in particular is a very competitive gradient approximation technique in the presence of numerical noise [15, 16, 19, 20, 22, 98, 125, 126, 133].

When gradient–based methods work, they often require fewer function evaluations than other methods [4] and have more robust convergence theory. However, "standard numerical optimization methods do not perform well unless the function to be minimized can be evaluated to a fairly high precision; this is necessary to have good estimates of numerical derivatives" [77]. Since gradient–based algorithms are challenged by the presence of numerical noise, we believe they may be challenged by other sources of noise in parameter identification problems. Due to the applicability of direct search methods and metaheuristics in the presence of varying levels of noise, we will confine our analysis to methods that do not require gradient information.

Direct Search Methods in the Presence of Numerical Noise

The first major class of optimization methods that we investigate are direct search methods. Direct search methods search for successively improving iterates without explicit or approximated gradient information. These methods make decisions based on function comparison [63]. Although a definitive characterization of direct search methods does not exist [80], Hooke and Jeeves propose the following [74]:

> We us the phrase "direct search" to describe sequential examination of trial solutions involving comparison of each trial solution with the "best" obtained up to that time together with a strategy for determining (as a function of earlier results) what the next trial solution will be.

Classical direct search methods include Rosenbrock's Method [107], the pattern search algorithm of Hooke and Jeeves [74], Powell's Method [101], Nelder and Mead's Downhill Simplex Method [93], and the Zangwill Method [139]. More recent investigations have yielded the Dennis and Torczon Algorithm [46], the Anderson and Ferris Algorithm [4], the Lucidi and Sciandrene algorithms [86], and the García–Palomares and Rodríguez algorithms [60]. In recent years, there has been a resurgence of interest in developing general convergence theory for a subset of direct search methods [80, 123].

At first glance, direct search methods may seem less attractive and robust than gradient–based methods since they do not use gradient information to identify decrease directions. However, direct search methods can be very useful for minimizing functions that are not differentiable, or even discontinuous. The presence of numerical noise in an optimization problem can result in nondifferentiability and discontinuities, which are obvious pitfalls for gradient–based methods. Direct search methods have been applied to noisy objective functions with good results [4, 6, 23, 77, 85, 124]. In cases such as [22], where finite difference gradients vary widely in magnitude and sign, there are many points where the gradient approximation is near zero. Gradient–based algorithms may use $\nabla f \approx 0$ as termination criteria. When noise adds spurious local minima to an objective function, gradient–based methods may terminate at these points. Direct search methods do not consider gradient information in their termination criteria, so this particular pitfall is avoided. Thus, direct search methods may be better able to handle the presence of numerical and other types of noise than gradient–based methods.

Metaheuristics in the Presence of Numerical Noise

The second major class of optimization algorithms that we study are the so–called "metaheuristics" (also known as intelligent optimization techniques).

Definition 2.3.1 *A* metaheuristic *is an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space. . . in order to find efficiently near–optimal solutions [97].*

Metaheuristics search solution spaces in an intelligently random manner. They were initially developed to search solution spaces of combinatorial or discrete problems [105]. Trial points at each iteration are not selected based on gradient information or a model of the objective function about the current iterate. Instead, new points to evaluate are chosen through a random process that varies by algorithm. Metaheuristics have demonstrated good results for a variety of problems, though performance is highly problem dependent [105, 138]. Since gradients are not computed or considered, metaheuristics can be applied to problems with discontinuous objective functions or discrete solution spaces. Although problem size and complexity affect the performance of heuristics, these factors do not preclude their use. Metaheuristics can return good results for complicated problems since simplifying models are not generated in the course of decision making. However, due to the loose guidelines that determine new iterates, neither feasibility, optimality, nor convergence can be proven for the category at large [105]. Popular metaheuristics include genetic algorithms, tabu searches, simulated annealing algorithms, evolutionary algorithms, artificial neural networks, and ant colony optimization [96, 99].

Metaheuristics can be modified to optimize continuous problems. When the metaheuristic cannot be applied directly, the guidelines or theory behind the method can be applied to selection of iterates for continuous problems. Investigations of metaheuristics applied to continuous optimization problems are relatively new and have generated a great deal of interest. For examples of tabu searches, genetic algorithms, and evolution strategies applied to continuous problems with competitive results, see [58, 81, 114, 117, 137].

As with other approaches to optimization, the performance of metaheuristics is affected by numerical noise. As in the case of direct search methods, the absence of gradient approximations reduces the effect of noise on metaheuristics. Spurious local minima or discontinuities caused by noise will not preclude the use of metaheuristics. Noise can affect the decisions made during exploration of the search space, however, thereby affecting the outcome of the algorithm [54]. Evolution strategies in particular have been hailed as effective in the presence of numerical noise [6]. For examples of metaheuristics applied to problems with numerical noise, see [61, 67, 140]. The promising results when metaheuristics are applied to problems with numerical noise may indicate a robust performance in the presence of other types of noise.

Chapter 3

Optimization Algorithms and Theory

Many methods exist for finding or approximating $\mathbf{x}_*$. The techniques discussed in Section 2.3 have all been applied to Σ and the NLSOP. We investigate the performance of three optimization methods when applied to noisy parameter identification optimization problems. The algorithms are discussed in detail in the following sections.

3.1 The Nelder–Mead Downhill Simplex Method

The first algorithm we consider is the Nelder–Mead Downhill Simplex Method, a direct search optimization method [93]. The Nelder–Mead Downhill Simplex Method (or Nelder–Mead Algorithm [NMA]) is widely used for nonlinear optimization problems such as Σ. The algorithm is so pervasive that pre–coded implementations appear in *Numerical Recipes in C: The Art of Scientific Computing* [102] and MATLAB's `fminsearch` function [91].

The NMA has several advantages. It is simple to program and implement. The algorithm is not hindered when derivative information is prohibitively expensive, completely unavailable, or even non-existent. When function evaluations are expensive, the NMA is quicker than many other direct search methods due to the small number of function evaluations needed for most iterations. Finally, the method is known for a noticeable reduction in objective function value in early iterations [82].

However, the NMA has drawbacks. The algorithm has been used for many years, but stringent convergence theory is sparse [82]. (Existing convergence results are discussed Sections 3.1.2 and 3.1.3). The NMA has a well–documented tendency to converge or terminate very slowly [4]. Finally, algorithm performance tends to decline as the problem dimension increases [80]. Due to these issues, efforts have been made to modify the NMA to improve its convergence behavior [29, 78, 103].

The NMA shares many of the advantages of direct search methods for problems with noisy

objective functions. Specifically, poor gradient approximations are not an issue. Continuity and differentiability of the objective function are not necessary. However, the convergence issues of the NMA are aggravated when numerical noise is present [2]. One suggestion to overcome this difficulty is to terminate the algorithm when the change in objective function between iterations is on the order of the magnitude of the noise in the problem [23, 134].

3.1.1 The Nelder–Mead Algorithm

The NMA moves through the solution space in the guise of a *simplex*.

Definition 3.1.1 *A* simplex *is a geometric figure that has a number of vertices (corners) equal to one more than the number of dimensions in the parameter space [134].*

A simplex is a set of $n+1$ points, or vertices, in $\mathbb{R}^n$. A nondegenerate simplex defines a convex hull in n-dimensions [82]. Thus, a nondegenerate simplex in one dimension corresponds to a line segment, while a nondegenerate simplex in two dimensions corresponds to a triangle. A nondegenerate simplex in three dimensions has four vertices and thus corresponds to a tetrahedron [134].

The NMA requires a user–defined starting point. This starting point is perturbed n times to generate the $n + 1$ linearly independent points of the initial simplex. The starting point is added to a multiple of the first n unit vectors in order to generate the new points. During execution of the NMA, objective function values are calculated for each vertex of the simplex. At each iteration, the algorithm replaces at least one vertex, thus changing the shape and location of the simplex. There are four ways the simplex can deform or move: reflection, expansion, contraction, or shrinkage. Figure 3.1 details the procedure the NMA follows at each iteration as applied to Σ [82, 93].

During a *reflection* step, the vertex $\mathbf{x}_{n+1}$ with the highest objective function value is projected through $\bar{\mathbf{x}}$ (the average of the other n vertices). The resulting point replaces $\mathbf{x}_{n+1}$ if it is an improvement over $\mathbf{x}_{n+1}$. When the reflection parameter α is equal to one, a reflection move preserves the hypervolume of the simplex [82]. During a simplex *expansion*, $\mathbf{x}_{n+1}$ is replaced by a point along the trajectory of the reflection point that is farther away from the simplex. In this case, simplex volume is not conserved. A *contraction* is similar to an expansion. However, the point that replaces $\mathbf{x}_{n+1}$ is a projection of $\mathbf{x}_{n+1}$ through $\bar{\mathbf{x}}$ scaled by a reducing factor. Clearly, the simplex volume is reduced during a contraction [134]. In a *shrinkage* step, every vertex except the one with the lowest fit value is replaced by a reduced reflection through its current value. See [82] for graphical illustrations of each simplex manipulation.

Select $\mathbf{x}_1$, $\mathbf{x}_2$, ..., $\mathbf{x}_{n+1}$ to form the initial simplex. Initialize $\alpha > 0$ (the reflection parameter), $\gamma > 1$ (the expansion parameter), $0 < \beta < 1$ (the contraction parameter), and $0 < \sigma < 1$ (the shrinkage factor).
Repeat until "convergence":

1. Order the elements in the simplex such that $f(\mathbf{x}_1) = \min_{1 \le j \le n+1} f(\mathbf{x}_j)$ and $f(\mathbf{x}_{n+1}) = \max_{1 \le j \le n+1} f(\mathbf{x}_j)$ and $f(\mathbf{x}_1) \le f(\mathbf{x}_2) \le \cdots \le f(\mathbf{x}_n) \le f(\mathbf{x}_{n+1})$. Define $\bar{\mathbf{x}} = \frac{1}{n} \sum_{j=1}^{n} \mathbf{x}_j$. Test termination criteria (3.1).

2. Let $\mathbf{x}_r = \bar{\mathbf{x}} + \alpha(\bar{\mathbf{x}} - \mathbf{x}_{n+1})$. If $f(\mathbf{x}_1) > f(\mathbf{x}_r)$, let $\mathbf{x}_e = \bar{\mathbf{x}} + \gamma(\mathbf{x}_r - \bar{\mathbf{x}})$ and go to step 3. If $f(\mathbf{x}_1) \le f(\mathbf{x}_r) < f(\mathbf{x}_{n+1})$, replace $\mathbf{x}_{n+1}$ by $\mathbf{x}_r$ and go to step 1. Otherwise, go to step 4.

3. Replace $\mathbf{x}_{n+1}$ by $\mathbf{x}_e$ (expansion) if $f(\mathbf{x}_r) > f(\mathbf{x}_e)$ and by $\mathbf{x}_r$ (reflection) if $f(\mathbf{x}_r) \le f(\mathbf{x}_e)$ to yield a new simplex. Return to step 1.

4. Let $\mathbf{x}_c = \bar{\mathbf{x}} + \beta(\mathbf{x}_{n+1} - \bar{\mathbf{x}})$. If $f(\mathbf{x}_c) < f(\mathbf{x}_{n+1})$, replace $\mathbf{x}_{n+1}$ with $\mathbf{x}_c$ (contraction) and go to step 1. Otherwise, go to step 5.

5. Replace $\mathbf{x}_j$ by $\mathbf{x}_1 + \sigma(\mathbf{x}_j - \mathbf{x}_1)$ for $j = 2, \ldots, n+1$ (shrinkage) and return to step 1.

Figure 3.1: The Nelder–Mead Downhill Simplex Algorithm

The termination criteria that we enforce for the NMA is based upon comparison of objective function values over the simplex. When

$$\frac{[f(\mathbf{x}_{n+1}) - f(\mathbf{x}_1)]}{\frac{1}{2}[f(\mathbf{x}_{n+1}) + f(\mathbf{x}_1)]} < 1 \times 10^{-5}, \tag{3.1}$$

the algorithm will terminate. (Note that the termination criteria is always non–negative, due to the ordering of the simplex vertices). As the simplex evolves to contain $n+1$ points with similar objective function values, termination will follow. Equation (3.1) does not imply that the simplex points are close together. Once the termination criteria (3.1) is true, the vertex with the lowest objective function value is returned.

3.1.2 Convergence Behavior For Non–Noisy Functions

Before convergence theory can be discussed, the following notation and definitions from [78] are required. Unless specified otherwise, subscripts indicate the element number of the simplex and superscripts indicate the iteration counter.

Definition 3.1.2 *Let S represent the simplex vertices $\{\mathbf{x}_j\}_{j=1}^{n+1}$ numbered according to increasing objective function value such that*

$$f_1 \leq f_2 \leq \ldots \leq f_{n+1}, \tag{3.2}$$

where f_j denotes $f(\mathbf{x}_j)$.

Definition 3.1.3 *Let the average objective function value $\underline{f}$ be calculated as*

$$\underline{f} = \frac{1}{n+1} \sum_{j=1}^{n+1} f(\mathbf{x}_j), \tag{3.3}$$

and let $\underline{f}^k$ denote $\underline{f}$ at the k^{th} iteration.

Definition 3.1.4 *The $n \times n$ matrix of simplex directions is denoted by*

$$V(S) = (\mathbf{x}_2 - \mathbf{x}_1, \mathbf{x}_3 - \mathbf{x}_1, \ldots, \mathbf{x}_{n+1} - \mathbf{x}_1) = (\mathbf{v}_1, \ldots, \mathbf{v}_n). \tag{3.4}$$

The term V^k denotes $V(S^k)$, the matrix of simplex directions at the k^{th} iteration.

Definition 3.1.5 *The simplex diameter is computed as*

$$d(S) = \max_{1 \leq i,j \leq n+1} \|\mathbf{x}_i - \mathbf{x}_j\| . \tag{3.5}$$

Definition 3.1.6 *The simplex condition number $\kappa(V)$ is the l^2 condition number of V:*

$$\kappa(V) = \|V\| \left\|V^{-1}\right\| = \left\|V^T\right\| \left\|V^{-T}\right\| . \tag{3.6}$$

Definition 3.1.7 *The term $\delta(f : S)$ is the vector of objective function differences and is computed as*

$$\delta(f : S) = (f(\mathbf{x}_2) - f(\mathbf{x}_1), f(\mathbf{x}_3) - f(\mathbf{x}_1), \ldots, f(\mathbf{x}_{n+1}) - f(\mathbf{x}_1))^T. \tag{3.7}$$

Definition 3.1.8 *The simplex gradient is defined as*

$$D(f : S) = V^{-T} \delta(f : S). \tag{3.8}$$

The term $D^k f$ denotes $D(f : S^k)$.

Definition 3.1.9 *The two oriented lengths $\sigma_+(S)$ and $\sigma_-(S)$ are computed as*

$$\sigma_+(S) = \max_{2 \leq j \leq n+1} \|\mathbf{x}_1 - \mathbf{x}_j\| \quad \textit{and} \tag{3.9}$$

$$\sigma_-(S) = \min_{2 \leq j \leq n+1} \|\mathbf{x}_1 - \mathbf{x}_j\| . \tag{3.10}$$

Lagarias et al. have proven limited convergence results for the original NMA, and we restate several of their results. The original statements and proofs of Lemmas 3.1.1 and 3.1.2 and Theorems 3.1.1, 3.1.2, and 3.1.3 can be found in [82].

Lemma 3.1.1 *If the initial simplex S^0 is nondegenerate, so are all subsequent NMA simplices.*

Lemma 3.1.2 *Assume f is strictly convex on $\mathbb{R}^n$ and the NMA is applied to f beginning with a nondegenerate simplex S^0. Then, no shrink steps will be taken.*

Theorem 3.1.1 (Convergence of the NMA in one dimension) *Let f be a strictly convex function on $\mathbb{R}^1$ with bounded level sets. Assume that the NMA is applied to f with algorithmic parameters satisfying $\alpha > 0$, $\gamma > 1$, $\gamma > \alpha$, $\alpha\gamma \geq 1$, and $0 < \beta < 1$, beginning with a nondegenerate initial simplex S^0. Then, both points of the Nelder–Mead simplex converge to x_*.*

Theorem 3.1.2 (Convergence of NMA vertex function values for $n = 2$) *Let f be a strictly convex function on $\mathbb{R}^2$ with bounded level sets. Assume that the NMA is applied to f with reflection coefficient $\alpha = 1$ and contraction coefficient $\beta = \frac{1}{2}$, beginning with a nondegenerate initial simplex S^0. Then, the three limiting vertex function values are equal. That is, if $f_i^* = \lim_{k\to\infty} f_i^k$, then $f_1^* = f_2^* = f_3^*$.*

Theorem 3.1.3 (Convergence of NMA simplex diameters to zero for $n = 2$) *Let f be a strictly convex function on $\mathbb{R}^2$ with bounded level sets. Assume that the NMA is applied to f with reflection coefficient $\alpha = 1$, expansion coefficient $\gamma = 2$, and contraction coefficient $\beta = \frac{1}{2}$, beginning with a nondegenerate initial simplex S^0. Then the sequence of simplices $\{S^k\}$ generated by the algorithm satisfy $\lim_{k\to\infty} d(S^k) = 0$.*

Note that Theorems 3.1.2 and 3.1.3 do not imply convergence to x_*. The authors admit their results seem "weak" but theorize that these results reflect the limits of the convergence behavior of the NMA [82]. A number of examples have been constructed to demonstrate the "weak" convergence properties of the NMA. In one example, the NMA is actually shown to converge to a nonstationary point (a point with a nonzero gradient) when used to optimize certain two–dimensional, continuous, strictly convex functions with three continuous derivatives [89]. Another example of convergence to a nonstationary point for a two–dimensional nonconvex function is found in [135].

Kelley has proven convergence results for the NMA under certain assumptions regarding the behavior of the simplex [78]. Several supporting lemmas must be presented prior to the convergence theorem. Lemma 3.1.3 is referenced without proof in [78]. Since the result is important and not immediately apparent, our proof is presented here.

Lemma 3.1.3 $\sigma_+(S) \leq \|V\|$.

Proof
Let x_j^i indicate the i^{th} entry in vertex j of the simplex. By definition,

$$\begin{aligned}\sigma_+(S) &= \max_{2\leq j\leq n+1} \|\mathbf{x}_1 - \mathbf{x}_j\| = \max_{2\leq j\leq n+1} \sqrt{(x_1^1 - x_j^1)^2 + \cdots + (x_1^1 - x_j^{n+1})^2} \\ &= \max_{1\leq j\leq n} \left[\sum_{i=1}^{n} (v_j^i)^2\right]^{\frac{1}{2}} = \max_{1\leq j\leq n} \|\mathbf{v}_j\|.\end{aligned}$$

Choose $1 \leq k \leq n$ such that

$$\|\mathbf{v}_k\| = \max_{1\leq j\leq n} \|\mathbf{v}_j\|. \tag{3.11}$$

Then,

$$\sigma_+(S) = \|\mathbf{v}_k\| = \|V\mathbf{e}_k\|, \tag{3.12}$$

where $\mathbf{e}_k$ is the unit vector in direction k. By definition,

$$\|V\mathbf{e}_k\| \leq \sup_{\|\mathbf{x}\|=1} \|V\mathbf{x}\| = \|V\|. \tag{3.13}$$

Thus, $\sigma_+(S) \leq \|V\|$. □

Another supporting lemma from [40] is needed.

Lemma 3.1.4 *For* $A \in \mathbb{R}^{n\times n}$, $\|A\| = \left\|A^T\right\|$.

Lemma 3.1.5 is proven in [78]. This lemma is a key component of Kelley's convergence proof for the NMA.

Lemma 3.1.5 *Let S be a simplex meeting the conditions of Definition 3.1.2. Let ∇f be Lipschitz continuous in a neighborhood of S with Lipschitz constant $2K_f$. Then for $K = \sqrt{n}K_f$,*

$$\|\nabla f(\mathbf{x}_1) - D(f:S)\| \leq K\kappa(V)\sigma_+(S). \tag{3.14}$$

The convergence result also requires the following assumptions.

Assumption 3.1.1 *V^0 is nonsingular, the vertices of the simplex S^k satisfy Definition 3.1.2 for each k, and $\underline{f}^{k+1} < \underline{f}^k$ for all but a finite number of k.*

These assumptions are satisfied by the sequence of simplices generated by the NMA if the initial vertices are linearly independent and if only a finite number of shrink steps occur [78]. As evidenced by Lemma 3.1.2, there are cases where the assumption of no shrink steps is valid. Additionally, numerical investigations have shown that shrink steps are very uncommon [82, 122]. However, the assumption is not generally enforceable for the unmodified NMA. In addition to these assumptions, a sufficient decrease condition is required. Every iteration must satisfy

$$\underline{f}^{k+1} - \underline{f}^{k} < -\omega \left\| D^k f \right\|^2 \tag{3.15}$$

where

$$\omega = \omega_0 \frac{\sigma_+(S^0)}{\|D^0 f\|}, \tag{3.16}$$

and ω_0 is generally on the order of 10^{-4} [78]. Again, this condition is unenforceable in the general case.

We now state Kelley's convergence result. The proof is found in [78].

Theorem 3.1.4 (Convergence of the NMA to stationary points) *Let a sequence of simplices satisfy Assumption 3.1.1 and let the assumptions of Lemma 3.1.5 hold, with the Lipschitz constants K^k uniformly bounded. Assume that $\left\{\underline{f}^k\right\}$ is bounded from below. If the sufficient decrease condition* (3.15) *holds for all but finitely many k and the product $\sigma_+(S^k)\kappa(V^k) \to 0$ as $k \to \infty$, then any accumulation point of the simplices is a stationary point of f.*

For the NLSOP, $\underline{f}$ is always bounded below by 0.

Although practical implementations of the NMA do not generally meet Assumption 3.1.1 and the decrease condition (3.15), the theory resulting from these assumptions is useful when applying the algorithm. Additionally, the convergence theory can be used as a guideline when developing convergent variants of the NMA.

3.1.3 Convergence Behavior for Noisy Functions

In this section, we follow the example in [78] and consider objective functions of the form

$$f(\mathbf{x}) = g(\mathbf{x}) + \phi(\mathbf{x}), \tag{3.17}$$

where $g(\mathbf{x})$ is a smooth function with a Lipschitz continuous gradient and ϕ is the noise term. Although ϕ is assumed to be bounded and defined for all $\mathbf{x} \in \mathbb{R}^n$, ϕ is not required to be continuous, or even deterministic [78]. The noise term ϕ can be a random variable, and the value of $\phi^{\,k}(\mathbf{x})$ may not be the same value as $\phi^{\,k+1}(\mathbf{x})$.

A new term from [78] is defined to quantify the magnitude of ϕ over a simplex.

Definition 3.1.10 *Let* $\|\phi\|_S = \max_{1 \le j \le n+1} \|\phi(\mathbf{x}_j)\|$.

Lemma 3.1.6 below is an analogue to Lemma 3.1.5 for the noisy case, and the proof is found in [78].

Lemma 3.1.6 *Let S be a simplex as described in Definition 3.1.2. Let f satisfy equation* (3.17) *and let ∇g be Lipschitz continuous in a neighborhood of S with Lipschitz constant $2K_g$. Then, for $K = (K_g + 2)\sqrt{n}$,*

$$\|\nabla g(\mathbf{x}_1) - D(f : S)\| \le K\kappa(V) \left(\sigma_+(S) + \frac{\|\phi\|_S}{\sigma_+(S)} \right). \tag{3.18}$$

The following theorem (stated and proven in [78]) is a direct result of Lemma 3.1.6 and several additional assumptions.

Theorem 3.1.5 (Convergence of the NMA in the presence of noise) *Let a sequence of simplices satisfy Assumption 3.1.1, and let the assumptions of Lemma 3.1.6 hold with the Lipschitz constants K_g^k uniformly bounded. Assume that $\left\{\underline{f}^k\right\}$ is bounded from below. If* (3.15) *holds for all but finitely many k and*

$$\lim_{k \to \infty} \kappa(V^k) \left(\sigma_+(S^k) + \frac{\|\phi\|_{S^k}}{\sigma_+(S^k)} \right) = 0, \tag{3.19}$$

then any accumulation point of the simplices is a stationary point of g.

Note that equation (3.19) requires that the magnitude of the noise term approaches zero at stationary points in a sufficiently rapid manner [78].

3.2 The Nelder–Mead and Simulated Annealing Hybrid

The second algorithm we consider in the presence of noise is a hybrid of the NMA and Simulated Annealing theory (NMA/SA) suggested by [102]. The simulated annealing algorithm is a very popular metaheuristic. (See [97] for a complete description.) It has returned competitive results for a variety of problems [97], does not require the computational cost of gradient calculations or maintaining populations or lists of solutions, and is simple to understand and implement. Though the simulated annealing algorithm was originally intended for combinatorial problems, it has recently been applied successfully to continuous problems [37, 66].

Several combinations of simulated annealing and the NMA have been suggested [71, 102, 105] and outperform simulated annealing alone on test functions and practical problems [3]. A variant of the NMA/SA hybrid for constrained optimization has problems generated a great deal of interest and encouraging optimization results [30, 31]. Finally, simulated annealing is very popular in geophysical parameter identification problems [112].

3.2.1 The Nelder–Mead and Simulated Annealing Hybrid Algorithm

The NMA/SA is composed of the NMA described in Section 3.1.1 combined with random perturbations scaled by a temperature value [102]. Although each new simplex is generated in the manner detailed in Figure 3.1, the objective function values are perturbed in order to impact the progress of the algorithm. Every function value in the current simplex is perturbed in the following way:

$$\hat{f}(\mathbf{x}_j^k) = f(\mathbf{x}_j^k) + \left| T^k \log(\psi_j^k) \right|, \tag{3.20}$$

where T^k is the current temperature and ψ_j^k is a random number chosen from a uniform distribution between $[l_{SA}, 1]$ where $l_{SA} > 0$. (The recommended interval from [102] is actually $(0, 1)$, but we enforce strict upper and lower limits for the benefit of theoretical convergence analysis and numerical stability.) The effect of the perturbation is to increase f for each vertex of the simplex. Conversely, for each potential point $\mathbf{x}_p^k$ considered during the course of the algorithm, the function value is decreased:

$$\hat{f}(\mathbf{x}_p^k) = f(\mathbf{x}_p^k) - \left| T^k \log(\psi_p^k) \right|. \tag{3.21}$$

When the NMA is applied to the perturbed function values, the increased simplex function values (3.20) are compared with the decreased function values of potential points (3.21). Therefore, when $f(\mathbf{x}_p^k)$ is an improving value, the hybrid algorithm will always accept it, just as the NMA. However, when $\mathbf{x}_p^k$ is an uphill step, it will be randomly accepted based on the values of the temperature parameter and the random variables. When T^k is large, the probability of accepting an uphill step is greater. As T^k decreases, uphill steps are less likely to be accepted. When $T = 0$, the hybrid is exactly the NMA [102].

The temperature parameter T^k is iteratively decreased. The equation controlling T^k is referred to as a cooling schedule. We consider two cooling schedules suggested by [102].

Cooling Schedule 1: Fixed Decrease

$T^{k+1} = T^k(1 - \epsilon)$ after every 5 iterations, where T^0 is the initial temperature, and $\epsilon \in (0, 1)$.

Cooling Schedule 2: Accelerated Decrease

$T^k = T^0(1 - \frac{k}{K_{SA}})$ after every 5 iterations, where K_{SA} is the predetermined number of perturbed iterations and k is the current iteration number.

When Cooling Schedule 2 is used, $T^k \equiv 0$ after K_{SA} iterations. If the termination criteria has not been met after K_{SA} iterations, the algorithm continues exactly as the NMA, since the perturbation term is equal to 0.

The termination criteria for the NMA is also used for the NMA/SA hybrid. Equation (3.1) is applied to the unperturbed objective function values for the simplex. In our implementation, the current value of the temperature parameter is not a factor in the termination criteria. Termination may occur at large or small values for T^k. When the termination criteria is met, the algorithm returns the point found with the smallest unperturbed objective function value even if the point is not in the current simplex.

The goal of this hybrid is to improve upon performance of the NMA in the presence of local minima and noise. Although the NMA does not terminate prematurely at points where $\nabla f = 0$ (as can be the case for gradient–based optimization), the NMA is still a greedy algorithm. If the entire simplex rests within a true or noise–induced set of local minimizers, the simplex may not be able to escape. The acceptance of increasing function values by the NMA/SA hybrid will allow escape from neighborhoods of local minima [102]. A potential drawback of the NMA/SA hybrid, however, is an increase in computational time. Acceptance of uphill steps could result in a decrease in efficiency. Also, since the perturbed function values are ordered and compared, the vertex with the highest objective function value may not be replaced during simplex operations. Algorithm performance depends heavily on the initial value for T^0, the cooling schedule, and problem characteristics [130]. The cooling schedule and its parameters dictate the outcome of optimization results [1, 34, 69, 95, 109, 115].

3.2.2 Convergence Behavior for Non–Noisy Functions

We extend Kelley's convergence results for the NMA [78] to the NMA/SA hybrid. We accomplish this by treating the perturbation term as a noise term. We first consider convergence in the non–noisy case. In the following convergence analysis, we consider functions of the form

$$f(\mathbf{x}) = g(\mathbf{x}) + \phi_{SA}(\mathbf{x}). \tag{3.22}$$

As previously, we assume g has a Lipschitz continuous gradient. The term $\phi_{SA}(\mathbf{x})$ represents the perturbation term from equations (3.20) and (3.21). For our implementation of the NMA/SA, the random number ψ is contained within $[l_{SA}, 1]$. Thus, there is a bound on the magnitude of ϕ_{SA}:

$$\|\phi_{SA}\|_S \leq |T^0 \log(l_{SA})|. \tag{3.23}$$

Since T^0 is the largest possible value for T^k, this bound applies for each iteration of the NMA/SA. However, a stricter bound will hold at each specific iteration of the NMA/SA.

That is,

$$\|\phi_{SA}\|_{S^k} \leq \left| T^k \log(l_{SA}) \right|. \tag{3.24}$$

Most cooling schedules enforce monotonically decreasing values for T^k. (See [1] for counterexamples.) If $\{T^k\}$ is monotonically decreasing, then equation (3.24) clearly implies that the largest possible value for $\|\phi_{SA}\|_{S^k}$ is monotonically decreasing as well. Further, if $\lim_{k\to\infty} T^k = 0$, then $\lim_{k\to\infty} \|\phi_{SA}\|_{S^k} = 0$.

In order to prove convergence of the NMA/SA, we first prove the following lemma. The lemma is analogous to Lemma 3.1.5, and we proceed in the spirit of Kelley's proof.

Lemma 3.2.1 *Let S be a simplex as described in Definition 3.1.2. Let f satisfy equation (3.22) and let ∇g be Lipschitz continuous in a neighborhood of S with Lipschitz constant $2K_g$. Then, there exists $K > 0$, depending only on K_g, such that*

$$\|\nabla g(\mathbf{x}_1) - D(f:S)\| \leq K\kappa(V)\left(\sigma_+(S) + \frac{\|\phi_{SA}\|_S}{\sigma_+(S)}\right). \tag{3.25}$$

Proof

By definition,

$$\|\delta(\phi_{SA}:S)\| = \left[(\phi_{SA}(\mathbf{x}_2) - \phi_{SA}(\mathbf{x}_1))^2 + \cdots + (\phi_{SA}(\mathbf{x}_{n+1}) - \phi_{SA}(\mathbf{x}_1))^2\right]^{\frac{1}{2}}.$$

Since $|\,\phi_{SA}(\mathbf{x}_j) - \phi_{SA}(\mathbf{x}_1)| \leq 2\,\|\phi_{SA}\|_S$ for all $j \in [1, n+1]$,

$$\|\delta(\phi_{SA}:S)\| \;\leq\; \left[n(2\,\|\phi_{SA}\|_S)^2\right]^{\frac{1}{2}} = 2\sqrt{n}\,\|\phi_{SA}\|_S. \tag{3.26}$$

It follows from the definition of $D(f:S)$, norm properties, and Equation (3.26), that

$$\begin{aligned}
\|D(f:S) - D(g:S)\| &= \left\|V^{-T}(\delta(f:S) - \delta(g:S))\right\| \\
&\leq \left\|V^{-T}\right\| \|\delta(f:S) - \delta(g:S)\| \\
&= \left\|V^{-T}\right\| \|\delta(\phi_{SA}:S)\| \\
&\leq 2\sqrt{n}\left\|V^{-T}\right\| \|\phi_{SA}\|_S \\
&= 2\sqrt{n}\left\|V^{-T}\right\| \left\|V^T\right\| \left[\frac{\|\phi_{SA}\|_S}{\|V^T\|}\right] \\
&= 2\sqrt{n}\kappa(V)\left[\frac{\|\phi_{SA}\|_S}{\|V^T\|}\right].
\end{aligned}$$

Consequently, by Lemmas 3.1.3 and 3.1.4,

$$\|D(f:S) - D(g:S)\| \;\leq\; 2\sqrt{n}\kappa(V)\left[\frac{\|\phi_{SA}\|_S}{\sigma_+(S)}\right]. \tag{3.27}$$

Applying Lemma 3.1.5 to g yields

$$\|\nabla g(\mathbf{x}_1) - D(g:S)\| \leq \sqrt{n} K_g \kappa(V) \sigma_+(S). \tag{3.28}$$

Finally, by applying the triangle inequality to equations (3.27) and (3.28),

$$\|\nabla g(\mathbf{x}_1) - D(f:S)\| \leq \sqrt{n} K_g \kappa(V) \sigma_+(S) + 2\sqrt{n}\kappa(V) \left[\frac{\|\phi_{SA}\|_S}{\sigma_+(S)} \right]. \tag{3.29}$$

Since all of the terms in the inequality are non–negative, we can add quantities greater than zero to the multiplicative coefficients, and

$$\begin{aligned} \|\nabla g(\mathbf{x}_1) - D(f:S)\| &\leq \sqrt{n}(K_g + 2)\kappa(V)\sigma_+(S) + \sqrt{n}(2 + K_g)\kappa(V) \left[\frac{\|\phi_{SA}\|_S}{\sigma_+(S)} \right] \\ &= (\sqrt{n}K_g + 2\sqrt{n})\kappa(V) \left(\sigma_+(S) + \frac{\|\phi_{SA}\|_S}{\sigma_+(S)} \right). \end{aligned} \tag{3.30}$$

Thus, the proof is complete for $K = K_g\sqrt{n} + 2\sqrt{n}$. □

The cooling schedule is a critical component of the convergence proof. In particular, the rate of decrease in T^k influences the convergence properties of the NMA/SA. Thus, we need an additional assumption regarding the cooling schedule for the NMA/SA convergence proof.

Assumption 3.2.1 *Assume that the cooling schedule generates* $\{T^k\}$ *such that*

$$\lim_{k\to\infty} \kappa(V^k) \left(\frac{T^k}{\sigma_+(S^k)} \right) = 0. \tag{3.31}$$

Assumption 3.2.1 requires that the cooling schedule reduce T^k at a sufficient rate. It is not enough that $T^k \to 0$; T^k must approach zero "fast enough." There are only a few cases where Assumption 3.2.1 can be verified as true. These cases are discussed after the convergence proof. When Assumption 3.2.1 is true, the NMA/SA will converge under the same conditions that are sufficient for the NMA to converge.

We must address one more point before presenting our convergence result. Lemma 3.1.1 applies to the NMA/SA because the NMA/SA uses the NMA to generate new simplices. Although the NMA/SA may accept uphill steps, the geometric manipulations of the simplex vertices are the same techniques as used in the NMA. Thus, every possible simplex alteration in the NMA/SA will reduce or increase simplex volume by a finite multiplier. When a nondegenerate initial simplex is used, the simplex volume will not degenerate to zero as the NMA/SA progresses.

Theorem 3.2.1 (Convergence of the NMA/SA Hybrid to stationary points) *Let a sequence of simplices satisfy Assumption 3.1.1, and let the assumptions of Lemma 3.2.1 hold with the Lipschitz constants K_g^k uniformly bounded. Assume that $\left\{\underline{f}^k\right\}$ is bounded from below. If the cooling schedule satisfies Assumption 3.2.1, the sufficient decrease condition* (3.15) *holds for all but finitely many k, and*

$$\lim_{k\to\infty} \kappa(V^k)\sigma_+(S^k) = 0, \tag{3.32}$$

then any accumulation point of the NMA/SA simplices is a stationary point of g.

Proof

By Assumption 3.1.1 and Lemma 3.1.1, V^k is nonsingular for all $k > 0$, and $D^k f$ is defined for all $k > 0$. The decrease condition (3.15) and the boundedness of $\left\{\underline{f}^k\right\}$ imply that $\left\{\underline{f}^k\right\}$ converges to a constant. Assumption 3.1.1 and the decrease condition (3.15) thus imply that

$$\lim_{k\to\infty} \left\|D^k f\right\| = 0. \tag{3.33}$$

The triangle inequality applied to equation (3.27) from Lemma 3.2.1 implies that for $k = 1, 2, \ldots,$

$$\begin{aligned} \left\|D^k g\right\| &\leq \left\|D^k f\right\| + 2\sqrt{n}\kappa(V^k)\left[\frac{\|\phi_{SA}\|_{S^k}}{\sigma_+(S^k)}\right] &(3.34)\\ &\leq \left\|D^k f\right\| + K^k\kappa(V^k)\left[\sigma_+(S^k) + \frac{\|\phi_{SA}\|_{S^k}}{\sigma_+(S^k)}\right], &(3.35)\end{aligned}$$

where $K^k = K_g^k\sqrt{n} + 2\sqrt{n}$ is the term defined in Lemma 3.2.1. Let M be the uniform bound for the Lipschitz constants K_g^k. Then

$$K^k \leq \sqrt{n}(2 + M)$$

for all k. Thus, K^k is bounded for all k. The bound on K^k and the assumption outlined in equation (3.32) imply that

$$\lim_{k\to\infty} K^k\kappa(V^k)\sigma_+(S^k) = 0. \tag{3.36}$$

The bound on K^k, Assumption 3.2.1, and equation (3.24) imply that

$$\lim_{k\to\infty} K^k\kappa(V^k)\left[\frac{\|\phi_{SA}\|_{S^k}}{\sigma_+(S^k)}\right] = 0. \tag{3.37}$$

Thus, equations (3.33), (3.35), (3.36), and (3.37) imply that $\left\|D^k g\right\| \to 0$ as $k \to \infty$.

We apply Lemma 3.1.5 and norm manipulation to g to reveal that, for each $k > 0$,

$$\left\|\nabla g(\mathbf{x}_1^k)\right\| \leq \left\|D^k g\right\| + \sqrt{n} K_g^k \kappa(V^k)\sigma_+(S^k) \tag{3.38}$$

$$\leq \left\|D^k g\right\| + K^k \kappa(V^k)\left[\sigma_+(S^k) + \frac{\|\phi_{SA}\|_{S^k}}{\sigma_+(S^k)}\right]. \tag{3.39}$$

Since each of the terms on the right hand side of equation (3.39) approach 0,

$$\lim_{k\to\infty} \left\|\nabla g(\mathbf{x}_1^k)\right\| = 0. \tag{3.40}$$

Therefore, $\lim_{k\to\infty} \mathbf{x}_1^k$ is a stationary point of g.

Since $\kappa(V^k) \geq 1$ for all k (see [79] for proof), equation (3.32) implies that

$$\lim_{k\to\infty} \sigma_+(S^k) = 0. \tag{3.41}$$

Thus, all vertices have a common accumulation point. Therefore, if $\mathbf{x}_*$ is any accumulation point of the sequence $\left\{\mathbf{x}_1^k\right\}$, then $\mathbf{x}_*$ is an accumulation point of the simplices. Furthermore, $\nabla g(\mathbf{x}_*) = 0$. Therefore, any accumulation point of the simplices is a stationary point of g, and the proof is complete. □

Two subsets of cooling schedules give rise to corollaries to Theorem 3.2.1. Many cooling schedules are based on a predetermined number of iterations. The temperature is decreased as the algorithm progresses, and after a preset number of iterations have occurred, the temperature remains at a constant value of zero. Cooling Schedule 2 is an example of this type of cooling schedule, and [87] contains several other examples. For every cooling schedule of this type, Theorem 3.2.1 holds.

Corollary 3.2.1 *Let a sequence of simplices satisfy Assumption 3.1.1, and let the assumptions of Lemma 3.2.1 hold with the Lipschitz constants K_g^k uniformly bounded. Assume that $\left\{\underline{f}^k\right\}$ is bounded from below. Assume that the cooling schedule enforces a finite limit on the number of iterations where $T^k \neq 0$. If the sufficient decrease condition* (3.15) *holds for all but finitely many k and*

$$\lim_{k\to\infty} \kappa(V^k)\sigma_+(S^k) = 0, \tag{3.42}$$

then any accumulation point of the NMA/SA simplices is a stationary point of g.

Proof

Due to our assumption regarding the cooling schedule, there exists a k_T such that $T^k = 0$ for all $k \geq k_T$. Therefore,

$$\kappa(V^k)\left[\frac{T^k}{\sigma_+(S^k)}\right] = 0$$

for all $k \geq k_T$, and

$$\lim_{k\to\infty} \kappa(V^k) \left[\frac{T^k}{\sigma_+(S^k)}\right] = 0. \tag{3.43}$$

Therefore, the cooling schedule satisfies Assumption 3.2.1, and the result follows from Theorem 3.2.1. □

Another class of cooling schedules will always result in convergence of the NMA/SA under the same conditions that are sufficient for the NMA to converge. For these cooling schedules, the sequence $\{T^k\}$ must decrease in relation to the size of certain simplex properties.

Corollary 3.2.2 *Let a sequence of simplices satisfy Assumption 3.1.1, and let the assumptions of Lemma 3.2.1 hold with the Lipschitz constants K_g^k uniformly bounded. Assume that $\{\underline{f}^k\}$ is bounded from below. Assume that the cooling schedule generates $\{T^k\}$ such that*

$$T^k = o\left(\frac{\sigma_+(S^k)}{\kappa(V^k)}\right) \quad as \quad \sigma_+(S^k) \to 0. \tag{3.44}$$

If the sufficient decrease condition (3.15) *holds for all but finitely many k and*

$$\lim_{k\to\infty} \kappa(V^k)\sigma_+(S^k) = 0, \tag{3.45}$$

then any accumulation point of the NMA/SA simplices is a stationary point of g.

Proof

By equation (3.44),

$$\lim_{k\to\infty} \frac{T^k\kappa(V^k)}{\sigma_+(S^k)} = 0. \tag{3.46}$$

Thus, Assumption 3.2.1 is met, and the result follows from Theorem 3.2.1. □

An example of a cooling schedule that meets the terms of Corollary 3.2.2 is

$$T^k = T^0\frac{\sigma_+^2(S^k)}{\kappa(V^k)}. \tag{3.47}$$

3.2.3 Convergence Behavior for Noisy Functions

Now, we prove similar convergence results for optimization of noisy problems. In the following convergence analysis, we consider functions of the form

$$f(\mathbf{x}) = g(\mathbf{x}) + \phi(\mathbf{x}) + \phi_{SA}(\mathbf{x}) \tag{3.48}$$

where g and ϕ are as described in equation (3.17) and $\phi_{SA}(\mathbf{x})$ represents the perturbation term from Equations (3.20) and (3.21). Bounds (3.23) and (3.24) still apply to $\phi_{SA}(\mathbf{x})$.

In order to prove convergence of the NMA/SA, we first prove the following lemma. The lemma is analogous to Lemma 3.2.1, and we proceed in the spirit of our previous proof.

Lemma 3.2.2 *Let S be a simplex as described in Definition 3.1.2. Let f satisfy equation* (3.48) *and let ∇g be Lipschitz continuous in a neighborhood of S with Lipschitz constant $2K_g$. Then, there exists $K > 0$, depending only on K_g, such that*

$$\|\nabla g(\mathbf{x}_1) - D(f:S)\| \leq K\kappa(V)\left(\sigma_+(S) + \frac{\|\phi\|_S}{\sigma_+(S)} + \frac{\|\phi_{SA}\|_S}{\sigma_+(S)}\right). \tag{3.49}$$

Proof

By definition,

$$\|\delta(\phi:S)\| = \left[(\phi(\mathbf{x}_2) - \phi(\mathbf{x}_1))^2 + \cdots + (\phi(\mathbf{x}_{n+1}) - \phi(\mathbf{x}_1))^2\right]^{\frac{1}{2}}.$$

Since $|\,\phi(\mathbf{x}_j) - \phi(\mathbf{x}_1)| \leq 2\,\|\phi\|_S$ for all $j \in [1, n+1]$,

$$\begin{aligned}\|\delta(\phi:S)\| &\leq \left[n(2\,\|\phi\|_S)^2\right]^{\frac{1}{2}} \\ &= 2\sqrt{n}\,\|\phi\|_S\,.\end{aligned}$$

Thus,

$$\|\delta(\phi:S)\| \leq 2\sqrt{n}\,\|\phi\|_S\,. \tag{3.50}$$

By similar reasoning,

$$\|\delta(\phi_{SA}:S)\| \leq 2\sqrt{n}\,\|\phi_{SA}\|_S\,. \tag{3.51}$$

It follows from the definition of $D(f:S)$, norm properties, and equations (3.50) and (3.51) that

$$\begin{aligned}
\|D(f:S) - D(g:S)\| &= \left\|V^{-T}(\delta(f:S) - \delta(g:S))\right\| \\
&\leq \left\|V^{-T}\right\| \|\delta(f:S) - \delta(g:S)\| \\
&= \left\|V^{-T}\right\| \|\delta(\phi:S) + \delta(\phi_{SA}:S)\| \\
&\leq \left\|V^{-T}\right\| \left[\, \|\delta(\phi:S)\| + \|\delta(\phi_{SA}:S)\| \,\right] \\
&\leq 2\sqrt{n} \left\|V^{-T}\right\| \left[\, \|\phi\|_S + \|\phi_{SA}\|_S \,\right] \\
&= 2\sqrt{n} \left\|V^{-T}\right\| \|V^T\| \left[\frac{\|\phi\|_S + \|\phi_{SA}\|_S}{\|V^T\|}\right] \\
&= 2\sqrt{n}\kappa(V) \left[\frac{\|\phi\|_S + \|\phi_{SA}\|_S}{\|V^T\|}\right].
\end{aligned}$$

Then, by Lemmas 3.1.3 and 3.1.4,

$$\|D(f:S) - D(g:S)\| \leq 2\sqrt{n}\kappa(V) \left[\frac{\|\phi\|_S}{\sigma_+(S)} + \frac{\|\phi_{SA}\|_S}{\sigma_+(S)}\right]. \tag{3.52}$$

Applying Lemma 3.1.5 to g yields

$$\|\nabla g(\mathbf{x}_1) - D(g:S)\| \leq \sqrt{n} K_g \kappa(V) \sigma_+(S). \tag{3.53}$$

Finally, by applying the triangle inequality to equations (3.52) and (3.53),

$$\|\nabla g(\mathbf{x}_1) - D(f:S)\| \leq \sqrt{n} K_g \kappa(V) \sigma_+(S) + 2\sqrt{n}\kappa(V) \left[\frac{\|\phi\|_S}{\sigma_+(S)} + \frac{\|\phi_{SA}\|_S}{\sigma_+(S)}\right]. \tag{3.54}$$

Since all of the terms in the inequality are non–negative, we can add values greater than zero to the multiplicative coefficients, and

$$\begin{aligned}
\|\nabla g(\mathbf{x}_1) - D(f:S)\| &\leq \sqrt{n}(K_g + 2)\kappa(V)\sigma_+(S) + \sqrt{n}(2 + K_g)\kappa(V) \left[\frac{\|\phi\|_S}{\sigma_+(S)} + \frac{\|\phi_{SA}\|_S}{\sigma_+(S)}\right] \\
&= (\sqrt{n}K_g + 2\sqrt{n})\kappa(V) \left(\sigma_+(S) + \frac{\|\phi\|_S}{\sigma_+(S)} + \frac{\|\phi_{SA}\|_S}{\sigma_+(S)}\right).
\end{aligned} \tag{3.55}$$

Thus, the proof is complete for $K = K_g\sqrt{n} + 2\sqrt{n}$. □

The following Theorem is the analogue to Theorem 3.2.1 for the noisy case. As in the proof of Theorem 3.2.1, our proof of convergence in the presence of noise applies Lemma 3.1.1, and we only consider cooling schedules that satisfy Assumption 3.2.1.

Theorem 3.2.2 (Convergence of the NMA/SA Hybrid in the presence of noise) *Let a sequence of simplices satisfy Assumption 3.1.1, and let the assumptions of Lemma 3.2.2 hold with the Lipschitz constants K_g^k uniformly bounded. Assume that $\left\{\underline{f}^k\right\}$ is bounded from below. If the cooling schedule satisfies Assumption 3.2.1, the sufficient decrease condition* (3.15) *holds for all but finitely many k, and*

$$\lim_{k\to\infty} \kappa(V^k)\left(\sigma_+(S^k) + \frac{\|\phi\|_{S^k}}{\sigma_+(S^k)}\right) = 0, \tag{3.56}$$

then any accumulation point of the NMA/SA simplices is a stationary point of g.

Proof

By Assumption 3.1.1 and Lemma 3.1.1, V^k is nonsingular for all $k > 0$, and $D^k f$ is defined for all $k > 0$. The decrease condition (3.15) and the boundedness of $\left\{\underline{f}^k\right\}$ imply that $\left\{\underline{f}^k\right\}$ converges to a constant. Assumption 3.1.1 and the decrease condition (3.15) thus imply that

$$\lim_{k\to\infty} \left\|D^k f\right\| = 0. \tag{3.57}$$

The triangle inequality applied to equation (3.52) from Lemma 3.2.2 implies that for $k = 1, 2, \ldots,$

$$\begin{aligned} \left\|D^k g\right\| &\leq \left\|D^k f\right\| + 2\sqrt{n}\kappa(V^k)\left[\frac{\|\phi\|_{S^k}}{\sigma_+(S^k)} + \frac{\|\phi_{SA}\|_{S^k}}{\sigma_+(S^k)}\right] &(3.58)\\ &\leq \left\|D^k f\right\| + K^k\kappa(V^k)\left[\sigma_+(S^k) + \frac{\|\phi\|_{S^k}}{\sigma_+(S^k)} + \frac{\|\phi_{SA}\|_{S^k}}{\sigma_+(S^k)}\right], &(3.59)\end{aligned}$$

where $K^k = K_g^k\sqrt{n} + 2\sqrt{n}$ is the term defined in Lemma 3.2.2. Let M be the uniform bound for the Lipschitz constants K_g^k. Then

$$K^k \leq \sqrt{n}(2 + M)$$

for all k. Thus, K^k is bounded for all k. The bound on K^k and the assumption outlined in equation (3.56) imply that

$$\lim_{k\to\infty} K^k\kappa(V^k)\left[\sigma_+(S^k) + \frac{\|\phi\|_{S^k}}{\sigma_+(S^k)}\right] = 0. \tag{3.60}$$

The bound on K^k, Assumption 3.2.1, and equation (3.24) imply that

$$\lim_{k\to\infty} K^k\kappa(V^k)\left[\frac{\|\phi_{SA}\|_{S^k}}{\sigma_+(S^k)}\right] = 0. \tag{3.61}$$

Thus, equations (3.57), (3.59), (3.60), and (3.61) imply that $\left\|D^k g\right\| \to 0$ as $k \to \infty$.

Since all the terms added together in equation (3.56) are positive, each individual term must approach zero. Thus,

$$\lim_{k\to\infty} \kappa(V^k)\sigma_+(S^k) = 0. \tag{3.62}$$

We apply Lemma 3.1.5 and norm manipulation to g to reveal that, for each $k > 0$

$$\begin{aligned} \left\|\nabla g(\mathbf{x}_1^k)\right\| &\leq \left\|D^k g\right\| + \sqrt{n}K_g^k \kappa(V^k)\sigma_+(S^k) & (3.63)\\ &\leq \left\|D^k g\right\| + K^k \kappa(V^k)\left[\sigma_+(S^k) + \frac{\|\phi\|_{S^k}}{\sigma_+(S^k)} + \frac{\|\phi_{SA}\|_{S^k}}{\sigma_+(S^k)}\right]. & (3.64)\end{aligned}$$

Since all of the terms on the right hand side of equation (3.64) approach 0, $\lim_{k\to\infty}\left\|\nabla g(\mathbf{x}_1^k)\right\| = 0$. Therefore, $\lim\limits_{k\to\infty} \mathbf{x}_1^k$ is a stationary point of g.

Since $\kappa(V^k) \geq 1$ for all k, equation (3.62) implies that

$$\lim_{k\to\infty} \sigma_+(S^k) = 0. \tag{3.65}$$

Thus, all vertices have a common accumulation point. Therefore, if $\mathbf{x}_*$ is any accumulation point of the sequence $\left\{\mathbf{x}_1^k\right\}$, then $\mathbf{x}_*$ is an accumulation point of the simplices. Furthermore, $\nabla g(\mathbf{x}_*) = 0$. Therefore, any accumulation point of the simplices is a stationary point of g, and the proof is complete. □

As in the non–noisy case, two corollaries to Theorem 3.2.2 are true for certain kinds of cooling schedules.

Corollary 3.2.3 *Let a sequence of simplices satisfy Assumption 3.1.1, and let the assumptions of Lemma 3.2.2 hold with the Lipschitz constants K_g^k uniformly bounded. Assume that $\left\{\underline{f}^k\right\}$ is bounded from below. Assume that the cooling schedule enforces a finite limit on the number of iterations where $T^k \neq 0$. If the sufficient decrease condition* (3.15) *holds for all but finitely many k and*

$$\lim_{k\to\infty} \kappa(V^k)\left(\sigma_+(S^k) + \frac{\|\phi\|_S}{\sigma_+(S^k)}\right) = 0, \tag{3.66}$$

then any accumulation point of the NMA/SA simplices is a stationary point of g.

Proof

Due to our assumption regarding the cooling schedule, there exists a k_T such that $T^k = 0$ for all $k \geq k_T$. Therefore,

$$\kappa(V^k)\left[\frac{T^k}{\sigma_+(S^k)}\right] = 0$$

for all $k \geq k_T$, and

$$\lim_{k \to \infty} \kappa(V^k) \left[\frac{T^k}{\sigma_+(S^k)} \right] = 0. \tag{3.67}$$

Therefore, the cooling schedule satisfies Assumption 3.2.1, and the result follows from Theorem 3.2.2. □

Corollary 3.2.4 *Let a sequence of simplices satisfy Assumption 3.1.1, and let the assumptions of Lemma 3.2.2 hold with the Lipschitz constants K_g^k uniformly bounded. Assume that $\{\underline{f}^k\}$ is bounded from below. Assume that the cooling schedule generates $\{T^k\}$ such that*

$$T^k = o\left(\frac{\sigma_+(S^k)}{\kappa(V^k)} \right) \quad as \quad \sigma_+(S^k) \to 0. \tag{3.68}$$

If the sufficient decrease condition (3.15) *holds for all but finitely many k and if*

$$\lim_{k \to \infty} \kappa(V^k) \left(\sigma_+(S^k) + \frac{\|\phi\|_S}{\sigma_+(S^k)} \right) = 0, \tag{3.69}$$

then any accumulation point of the NMA/SA simplices is a stationary point of g.

Proof

By equation (3.68),

$$\lim_{k \to \infty} \frac{T^k \kappa(V^k)}{\sigma_+(S^k)} = 0. \tag{3.70}$$

Thus, Assumption 3.2.1 is met, and the corollary follows from Theorem 3.2.2. □

3.3 The Shuffled Complex Evolution Method

The final algorithm we consider is a hybrid of the NMA and evolutionary algorithms. The Shuffled Complex Evolution Method (SCEM) was developed by Duan, Gupta, and Sorooshian [47, 48]. Although it can be applied to the general optimization problem Σ, the algorithm was designed specifically for hydraulic parameter identification problems. The SCEM is a hybrid of four types of optimization approaches. It consists of "a combination of random and deterministic approaches..., the concept of clustering..., the concept of a systematic evolution of a complex of points spanning the space in the direction of global improvement..., and the concept of competitive evolution" [47]. It is largely based on a controlled random search algorithm developed by Price [104].

Evolutionary algorithms are metaheuristics that emulate the process of evolution. A population of solutions is created, and new solutions are found through probabilistic combination and mutation of solutions in the population [96]. There are many different implementations with varying rules for generation of new solutions. The SCEM expands on this concept by dividing the solution space into several "complexes." New solution points in each complex are generated using NMA reflections and contractions in combination with evolutionary methods. The addition of random search elements aids the algorithm in identifying new neighborhoods of interest and escaping neighborhoods of local minima. In keeping with the "competitive" nature of evolution, better solutions are more likely to be used as parents in the creation of new solutions. New complexes are periodically formed through recombination and shuffling of all current solutions. Figures 3.2 and 3.3 summarize the method as described in [47].

Our termination criteria for the SCEM is similar to the NMA termination criteria. The number of points per complex (m) is analogous to the size of the simplex in the NMA. Thus, when

$$\frac{f(\mathbf{x}_m) - f(\mathbf{x}_1)}{f(\mathbf{x}_m) + f(\mathbf{x}_1)} < 1 \times 10^{-5}, \tag{3.71}$$

the algorithm terminates. When the algorithm terminates, the point with the lowest objective function value ever found is returned, even if it is not in the current population. There is a second termination criteria for the SCEM. If the algorithm has reached a maximal number of fitness evaluations (5000 in our implementation), then it will automatically terminate and return the point with the lowest fit value ever found. When this termination criteria is employed, we refer to the run as having "prematurely terminated."

Evolution strategies are widely regarded as promising methods to use in the presence of noisy objective functions [6]. Indeed, some optimizers speculate that the consideration of populations rather than individual solutions results in a type of averaging that reduces the effects of the noise [5]. Numerical investigations of evolution strategies applied to noisy functions include [5, 6, 7, 13, 14]. The promising results suggest that the SCEM might also perform well for noisy parameter identification optimization problems. The SCEM has outperformed several other methods (including the NMA, controlled random searches, SA algorithms, and genetic algorithms) in test problems [36, 47, 48, 49, 132]. Although evolution strategies are widely used, convergence theory is sparse. As far as we can determine, there is no existing convergence theory for the SCEM.

Initialize $p \geq 1$ (number of complexes), $m \geq n + 1$ (number of points per complex), and $s = p * m$ (sample size). Let $\Omega \subset \mathbb{R}^n$ be the set of feasible points.

1. Generate a sample of s points from Ω assuming a uniform distribution. Evaluate $f(\mathbf{x}_i)$ for $i = 1, \ldots, s$. Renumber the points in order of ascending function values so that $f(\mathbf{x}_1) \leq f(\mathbf{x}_2) \leq \ldots f(\mathbf{x}_s)$. Let $D = \{(\mathbf{x}_i, f(\mathbf{x}_i)) : i = 1, \ldots s\}$.
2. Partition D into p complexes $A^1, \ldots, A^p$ of size m using the following equation:

$$A^k = \{(\mathbf{x}_j^k, f(\mathbf{x}_j^k)) : \mathbf{x}_j^k = \mathbf{x}_{j+p(k-1)}, j = 1, \ldots, m\}.$$

3. Evolve each complex using the CCE algorithm found in Figure 3.3.
4. Recreate D by combining all points from all complexes and order as in step 1.
5. Determine if termination criteria are met. If not, return to step 2.

Figure 3.2: The Shuffled Complex Evolution Method

Initialize integers $2 \leq q \leq m$, $\alpha_s \geq 1$, and $\beta_s \geq 1$. The parameter q determines the number of parent solutions used in each reproduction step.

1. Create a triangular probability distribution for the elements of A^k. Let $\text{prob}(\mathbf{x}_i) = 2(m+1-i)/m(m+1), i = 1, \ldots, m$. Thus, the point $\mathbf{x}_1^k$ has the lowest function value and highest probability, while $\mathbf{x}_m^k$ has the highest function value and lowest probability. (This distribution ensures the competitive aspect of the evolution process).

2. Sample q different points $\mathbf{u}_1, \ldots, \mathbf{u}_q$ from A^k following the triangular probability distribution. Store these points in $B = \{(\mathbf{u}_i, f(\mathbf{u}_i)) : i = 1, \ldots, q\}$.

3. (a) Renumber B in order of ascending function value. Compute

$$\mathbf{u}_g = \frac{1}{q-1} \sum_{j=1}^{q-1} \mathbf{u}_j.$$

   (b) Compute $\mathbf{u}_r = 2\mathbf{u}_g - \mathbf{u}_q$. (This is analogous to the reflection point in the NMA).

   (c) If $\mathbf{u}_r \in \Omega$, compute $f(\mathbf{u}_r)$ and go to step (d). Otherwise, generate the smallest hypercube $H \subset \mathbb{R}^n$ that contains A^k. Sample a point $\mathbf{z} \in H$, evaluate $f(\mathbf{z})$, and set $\mathbf{u}_r = \mathbf{z}$.

   (d) If $f(\mathbf{u}_r) < f(\mathbf{u}_q)$, replace $\mathbf{u}_q$ with $\mathbf{u}_r$ and go to step (f). Otherwise, let $\mathbf{u}_c = \frac{\mathbf{u}_g + \mathbf{u}_q}{2}$ and evaluate $f(\mathbf{u}_c)$.

   (e) If $f(\mathbf{u}_c) < f(\mathbf{u}_q)$, replace $\mathbf{u}_q$ with $\mathbf{u}_c$ and go to step (f). (This is analogous to an NMA contraction). Otherwise, sample a random $\mathbf{z} \in H$, compute $f(\mathbf{z})$, and set $\mathbf{u}_q = \mathbf{z}$. (This is a mutation step).

   (f) Repeat (a) through (e) α_s times.

4. Insert the members of B into A^k by replacing the q points sampled in step 2. Renumber A^k in order of ascending function values.

5. Repeat steps 1 through 4 β_s times.

Figure 3.3: The Competitive Complex Evolution Strategy

Chapter 4

Test Problems

We test the three optimization algorithms described in Chapter 3 on seven test problems. All test problems are parameter identification problems formulated as nonlinear least–squares optimization problems. Each problem requires that the optimization algorithms find parameter combinations that minimize the difference between a model and a set of data. In the first six test problems, the data that is used for comparison is numerically generated as opposed to experimentally collected. In these six problems, various levels of noise are added to the data. In the seventh test problem, experimentally collected data is used.

4.1 Polynomial Test Problems

The first set of test problems are based on polynomial functions. The fifth degree polynomial $y(x) = 0.25x + 0.05x^2 + 0.15x^3 + 0.1x^4 - 0.01x^5$ is the function we try to identify. The polynomial is evaluated at 101 uniformly spaced points on the interval $x \in [0, 10]$. We denote the vector of matching points by $\mathbf{d} \in \mathbb{R}^{101}$. Thus,

$$d_i = y(x_i) = 0.25x_i + 0.05x_i^2 + 0.15x_i^3 + 0.1x_i^4 - 0.01x_i^5, \tag{4.1}$$

where $x_1 = 0$, $x_2 = 0.1$, ..., $x_{100} = 9.9$, and $x_{101} = 10$. The simulated polynomial attempts to match these 101 points.

The software we use to simulate the polynomials is paCalc, a parameter identification framework developed by INTERA Engineering [76]. The forward problem is solved using paCalc both for data generation purposes and for optimization simulations. We use a Dell Precision Workstation 360 with a Microsoft Windows XP Professional Operating System for all of the computations.

4.1.1 Test Problem 1: Fifth Degree Polynomial Model

In Test Problem 1a, we use a fifth degree polynomial to model the data set $\mathbf{d}$. The optimized parameters are the five polynomial coefficients. (For the sake of simplicity, the constant term for every polynomial we study is set equal to zero.) Thus, the simulated polynomial for polynomial coefficients $\mathbf{p} \in \mathbb{R}^5$ is

$$y_5(\mathbf{p}; x) = p_1 x + p_2 x^2 + p_3 x^3 + p_4 x^4 + p_5 x^5, \tag{4.2}$$

and the objective function f for Test Problem 1a is

$$f(\mathbf{p}) = \sum_{i=1}^{101} (y_5(\mathbf{p}; x_i) - d_i)^2. \tag{4.3}$$

The formal optimization problem for Test Problem 1a is to find some $\mathbf{p}_* \in \mathbb{R}^5$ such that

$$f(\mathbf{p}_*) \leq f(\mathbf{p}) \quad \text{for all} \quad \mathbf{p} \in \mathbb{R}^5. \tag{4.4}$$

Test Problem 1b uses the same model to simulate the objective function. However, a noise term is added to the to 101 data points. Thus,

$$d_i^L = 0.25x_i + 0.05x_i^2 + 0.15x_i^3 + 0.1x_i^4 - 0.01x_i^5 + \phi_L(x_i), \tag{4.5}$$

where ϕ_L is a random noise term with a normal distribution of zero mean and a standard deviation of five. Thus, the least–squares objective function for Test Problem 1b is

$$f(\mathbf{p}) = \sum_{i=1}^{101} (y_5(\mathbf{p}; x_i) - d_i^L)^2. \tag{4.6}$$

In Test Problem 1c, a noise term ϕ_H is added to the data points. This noise term has a standard deviation of ten. Thus,

$$d_i^H = 0.25x_i + 0.05x_i^2 + 0.15x_i^3 + 0.1x_i^4 - 0.01x_i^5 + \phi_H(x_i). \tag{4.7}$$

The noise term in equation (4.7) causes a greater distortion of the data points than the noise term in equation (4.5). The objective function for Test Problem 1c is

$$f(\mathbf{p}) = \sum_{i=1}^{101} (y_5(\mathbf{p}; x_i) - d_i^H)^2. \tag{4.8}$$

Clearly, Test Problems 1b and 1c are noisy variations of Test Problem 1a.

For Test Problems 1a through 1c, the parameter vector $\mathbf{p} = [0.25, 0.05, 0.15, 0.1, -0.01]$ is known to be a global minimum of the unperturbed problem. The existence of other global minima or possible local minima is not known. Figure 4.1 is a plot of the true polynomial and the 101 data points. Figures 4.2 and 4.3 are plots of the data points with the two different levels of noise added.

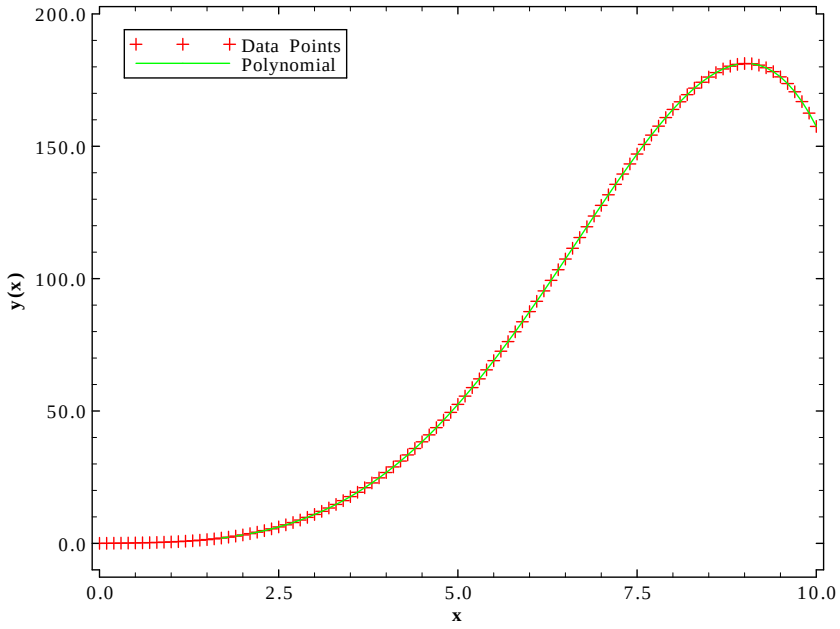

Figure 4.1: Fifth Order Polynomial and Matching Points

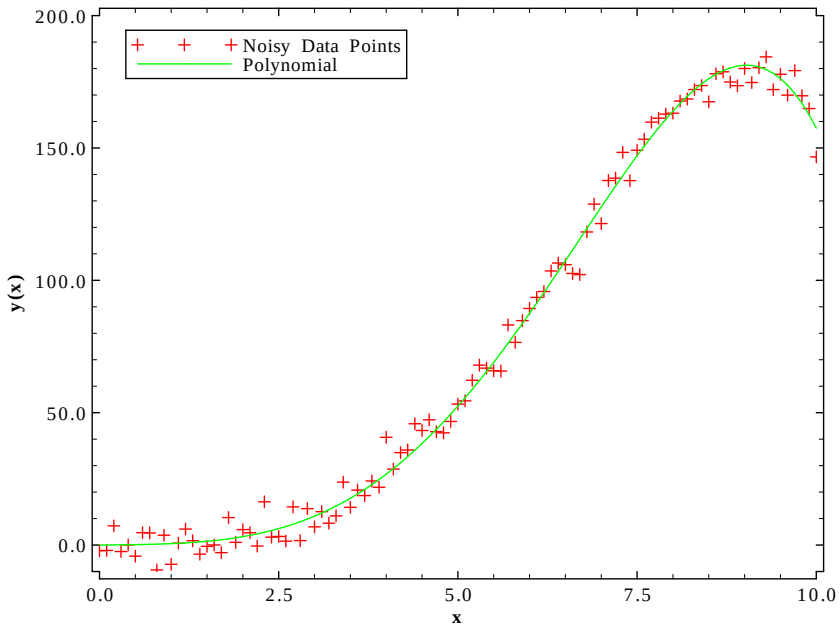

Figure 4.2: Fifth Order Polynomial and Matching Points With Low Level of Noise

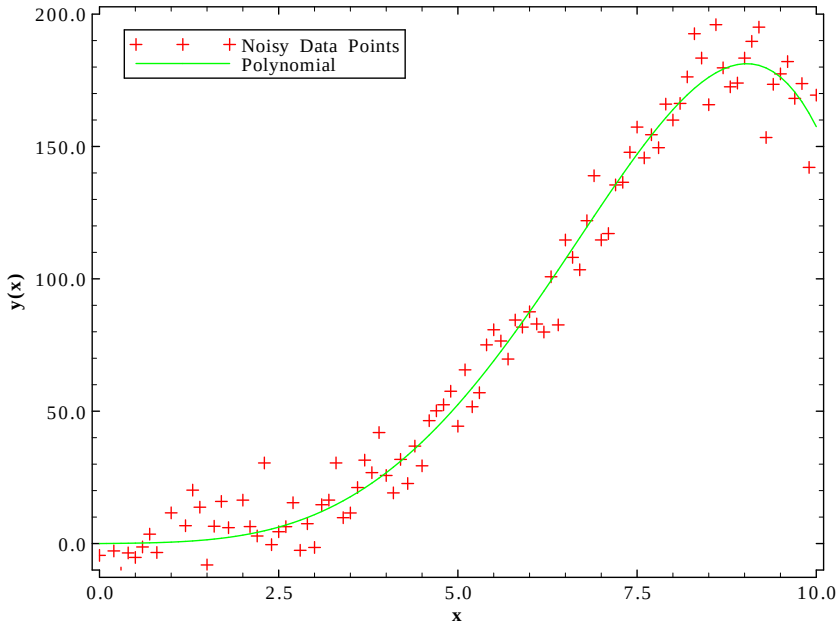

Figure 4.3: Fifth Order Polynomial and Matching Points With High Level of Noise

4.1.2 Test Problem 2: Third Degree Polynomial Model

Test Problem 2*a* is the best approximation of the unperturbed fifth degree polynomial pictured in Figure 4.1 using a third degree polynomial. In Test Problem 2*a*, the fifth degree polynomial is modeled with a third degree polynomial, and the optimized parameters are the three polynomial coefficients. Thus, the simulated polynomial for polynomial coefficients $\mathbf{p} \in \mathbb{R}^3$ is

$$y_3(\mathbf{p}; x) = p_1 x + p_2 x^2 + p_3 x^3, \tag{4.9}$$

and the objective function f for Test Problem 2*a* is

$$f(\mathbf{p}) = \sum_{i=1}^{101} (y_3(\mathbf{p}; x_i) - d_i)^2. \tag{4.10}$$

The formal optimization problem for Test Problem 2*a* is to find some $\mathbf{p}_* \in \mathbb{R}^3$ such that

$$f(\mathbf{p}_*) \leq f(\mathbf{p}) \quad \text{for all} \quad \mathbf{p} \in \mathbb{R}^3. \tag{4.11}$$

Test Problem 2*b* uses the third degree polynomial simulation as the numerical model. However, the perturbed data set $\mathbf{d}^L$ from (4.5) is the set of matching points. Thus, the least–

squares objective function for Test Problem 2*b* is

$$f(\mathbf{p}) = \sum_{i=1}^{101} (y_3(\mathbf{p}; x_i) - d_i^L)^2. \tag{4.12}$$

Finally, Test Problem 2*c* is the simulation of the data set $\mathbf{d}^H$ from (4.7) using a third degree polynomial. Thus, for Test Problem 2*c*,

$$f(\mathbf{p}) = \sum_{i=1}^{101} (y_3(\mathbf{p}; x_i) - d_i^H)^2. \tag{4.13}$$

Test Problems 2*a* through 2*c* are all perturbed versions of Test Problem 1*a*. Additionally, Test Problems 2*b* and 2*c* are noisy versions of 2*a*. Although Test Problem 2*a* does not contain noisy data points, the use of a third degree polynomial to model a fifth degree polynomial introduces model structure error. Test Problems 2*a* through 2*c* represent the case where the model is inadequate to completely describe the modeled system. Consequently, these problems are high residual problems. Large residual problems challenge many optimization algorithms more than low residual problems. A third degree polynomial is a poor approximation of a fifth degree polynomial, so even at a globally optimal point, there will be a large residual value. There is no prior knowledge of existence of global or local minima for Test Problems 2*a* through 2*c*.

4.2 Groundwater Parameter Identification Test Problem

Groundwater hydrology is the field concerned with understanding and predicting the flow of water in subsurface environments [12]. The field is a combination of fluid mechanics and geosciences, since subsurface water movement reflects characteristics of each of these fields [92]. Due to the complexity of geologic formations, development of accurate models for groundwater flow is a challenging task. The practical applications of these models are far–reaching. Contaminant transport, availability of potable water, and construction planning are only a few instances where groundwater models are useful decision making tools [59]. Because groundwater parameter identification problems are so widespread and important, we choose Test Problem 3 to be a groundwater parameter identification problem using real data.

4.2.1 Governing Equations

The following equations model flow through a geologic formation. Since state variables are measured through use of a well, the governing equations reflect the presence and influence

of the well.

Our model for groundwater flow assumes a saturated, homogeneous, isotropic aquifer and describes radial flow towards or away from a well [106]. We make use of several equations. The following "generalized radial flow equation" is derived in [9]:

$$S_s \frac{\partial h(r,t)}{\partial t} = \frac{K}{r^{n_f-1}} \frac{\partial}{\partial r} \left(r^{n_f-1} \frac{\partial h(r,t)}{\partial r} \right), \tag{4.14}$$

where

$$\begin{aligned} S_s &= \text{specific storage (1/m)}, \\ h &= \text{hydraulic head (m)}, \\ t &= \text{time (sec)}, \\ K &= \text{hydraulic conductivity (m/sec)}, \\ r &= \text{radial distance from borehole (m), and} \\ n_f &= \text{flow dimension (dimensionless)}. \end{aligned}$$

This model was developed using conservation of mass and Darcy's law [59]:

$$Q(r) = -K \frac{\partial h}{\partial r} A(r), \tag{4.15}$$

where Q is the flowrate (m^3/sec) and $A(r)$ is the flow area (m^2) at a distance r from the borehole. The flow area is computed following a method from [106]:

$$A(r) = A(r_w) \left(\frac{r}{r_w} \right)^{n_f-1}, \tag{4.16}$$

where r_w (m) is the well radius and $A(r_w)$ (m^2) is the cross–sectional area of the borehole. The relationship between hydraulic head and pressure is

$$p = \rho g h, \tag{4.17}$$

where p (kPa) is pressure, ρ is the density of water (kg/m^3), and g (m/sec^2) is the acceleration due to gravity.

The model also considers the effects of test zone compressibility on groundwater flow. The equation describing compressibility is

$$Q_w = V_w C_{tz} \frac{\partial p_w(r,t)}{\partial t}, \tag{4.18}$$

where Q_w (m^3/sec) is flow at the edge of the well, V_w (m^3) is the volume of the well, p_w (kPa) is pressure within the well, and C_{tz} (1/kPa) is the test zone compressibility [8].

The initial condition is

$$p(r, 0) = p_0, \tag{4.19}$$

where $p_0 = 953.4$ kPa is the static formation pressure. The boundary condition at the well is

$$p(0, t) = p^m(t), \tag{4.20}$$

where $p^m(t)$ is the measured value of the pressure at the well at time t. The boundary condition at the external boundary is

$$p(r_b, t) = p(r_b) = p_0, \tag{4.21}$$

where r_b is the radial distance from the well to the boundary [8].

4.2.2 Numerical Simulation Techniques

Sandia National Laboratories has developed an n–dimensional Statistical Inverse Graphical Hydraulic Test Simulator (nSIGHTS), a powerful program capable of analyzing well–test data. nSIGHTS is capable of simulating "the hydraulic response of a single–phase, one–dimensional, radial/non–radial flow regime to boundary conditions applied at a borehole located at the center of the modeled flow system" [106]. nSIGHTS was specifically developed to model the case of low–permeability. This software is used to simulate the forward problem, and it also contains the optimization framework. Again, we use a Dell Precision Workstation 360 with a Microsoft Windows XP Professional Operating System for all calculations.

Graph theory is used to develop the systems of algebraic equations that approximate the solutions to the governing equations. Rather than discretizing the partial differential equations (as in a finite difference or finite element approach), graph theory involves developing a numerical model directly from the physics of the system. The system is considered as a continuum and divided into a series of concentric cylindrical volumes centered around the well [8]. Each volume is considered a node, and each node is assumed to have constant pressure throughout. The field graph is constructed by "joining the nodes with a graph edge wherever a pair of variables can be associated with a constitutive property or a boundary condition" [110]. The state variables in our system are the pressure and flowrate. The constitutive equations, boundary conditions, and balance laws are expressed in terms of the variables. The resulting system of algebraic equations is solved using appropriate numerical techniques. (See [8] for details for specific types of problems). When derivatives are approximated, the forward difference finite difference scheme is used. See [100] and [110] for a complete discussion of the Graph Theoretic Field Model. nSIGHTS's use of the model is discussed in [8].

4.2.3 Test Problem 3: Groundwater Parameter Identification Problem

The data set that we match is from a constant pressure well test found in [106]. We use the collected data from Moderately Fractured Test 16-7. The data set consists of pressure and flowrate values versus time. Figure 4.4 is a plot of the pressure data versus time. In the first time interval, the formation is at equilibrium, and the pressure has a relatively constant value at static formation pressure. The pressure is then reduced over a 90 second period. When the reduction period is complete, the pressure is held between -4.5 and -5.0 kPa. Ideally, the pressure would have a constant value during the testing period, but equipment limitations preclude truly constant values. Thus, the pressure is measured every second during the testing interval, and the measured value of the pressure is used in creation of the simulated results. When the testing interval is complete, the pressure controls are released. The final phase in Figure 4.4 illustrates the recovery of the pressure towards the static formation pressure. Although data was collected for equilibrium, testing, and recovery periods, the only data we attempt to match is that collected during the testing period.

We refer to the set of measured flowrate data as $\mathbf{d}^Q$. There are 590 time intervals in the testing period, so $\mathbf{d}^Q \in \mathbb{R}^{590}$. Figure 4.5 is an unmodified plot of the measured flowrate data. The spike in the data set is due to the effects of compressibility on the flowrate during the 90 second pressure reduction period. The scale of this plot hides important information about the flowrate. Thus, the data is transformed so that $t = 1$ corresponds to the beginning of the pressure reduction phase. The data is plotted on a logarithmic scale in Figure 4.6 for the entire testing period. This plot represents the data that we try to simulate. A visual inspection of the plot reveals noise in the data points.

Our parameter vector $\mathbf{p}$ has four components. These are the flow dimension n_f, the conductivity K, the specific storage S_s, and the compressibility c_{TZ}. The compressibility only affects the flowrate during the 90 second pressure reduction period. The spikes in Figure 4.6 are also due to the effects of compressibility during that reduction period.

The objective function for Test Problem 3 is as follows. Let $\mathbf{p}^m$ denote the vector of measured pressure values over the testing period, and let $\mathbf{y}_p(\mathbf{p}, \mathbf{p}^m)$ denote the simulated flowrate . The objective function f is

$$f(\mathbf{p}) = \sum_{i=1}^{590} (y_p(\mathbf{p}, p_i^m) - d_i^Q)^2. \tag{4.22}$$

The optimization problem is to find $\mathbf{p}_* \in \mathbb{R}^4$ such that

$$f(\mathbf{p}_*) \leq f(\mathbf{p}) \text{ for all } \mathbf{p} \in \mathbb{R}^4. \tag{4.23}$$

The noise in this test problem is not known to be bounded. Sources of noise include noise in the collected data, approximate solution of the groundwater model, and model structure

error. The existence and location of global and local minima are also unknown. However, hydraulic parameters are often highly correlated, so multiple global and local minima are possible [106].

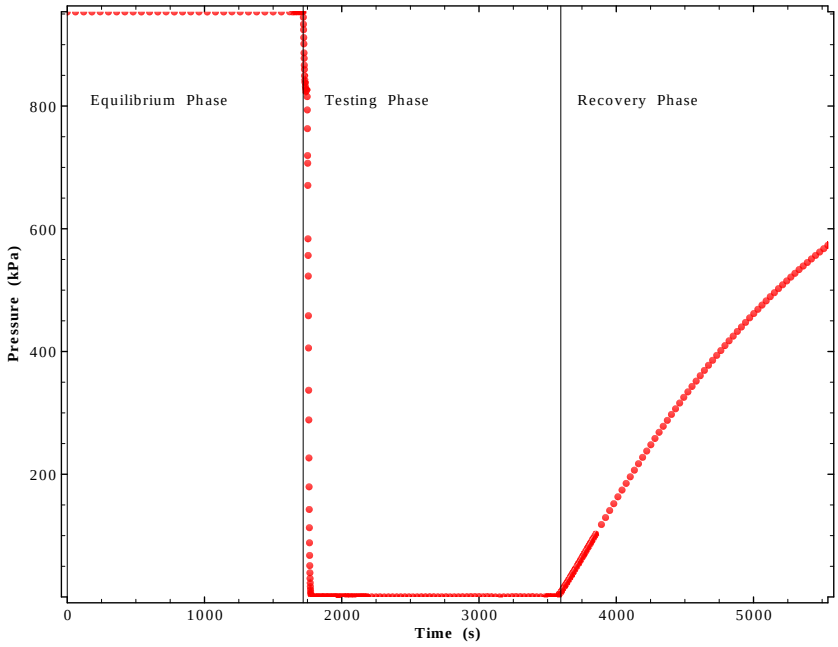

Figure 4.4: Test Problem 3: Measured Pressure Values

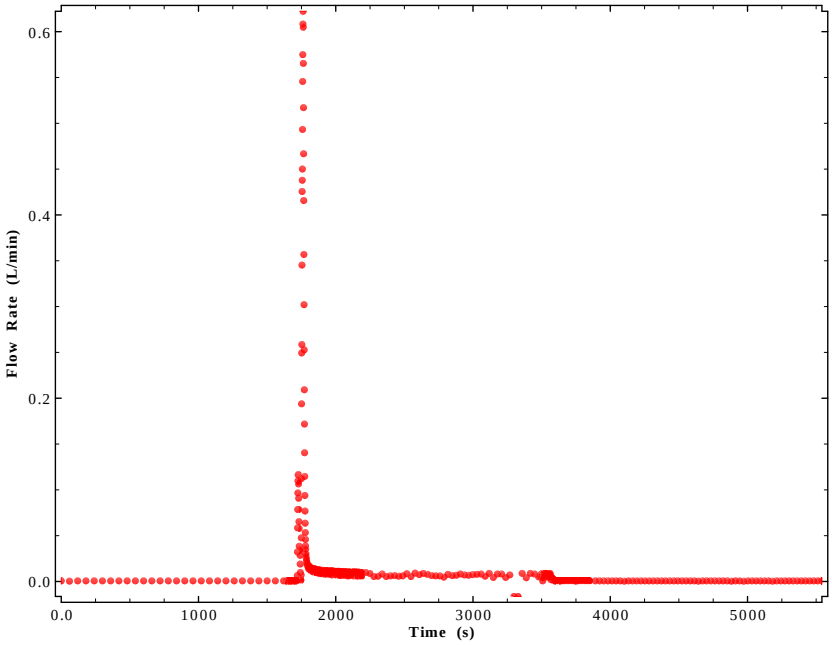

Figure 4.5: Test Problem 3: Measured Flowrate Values

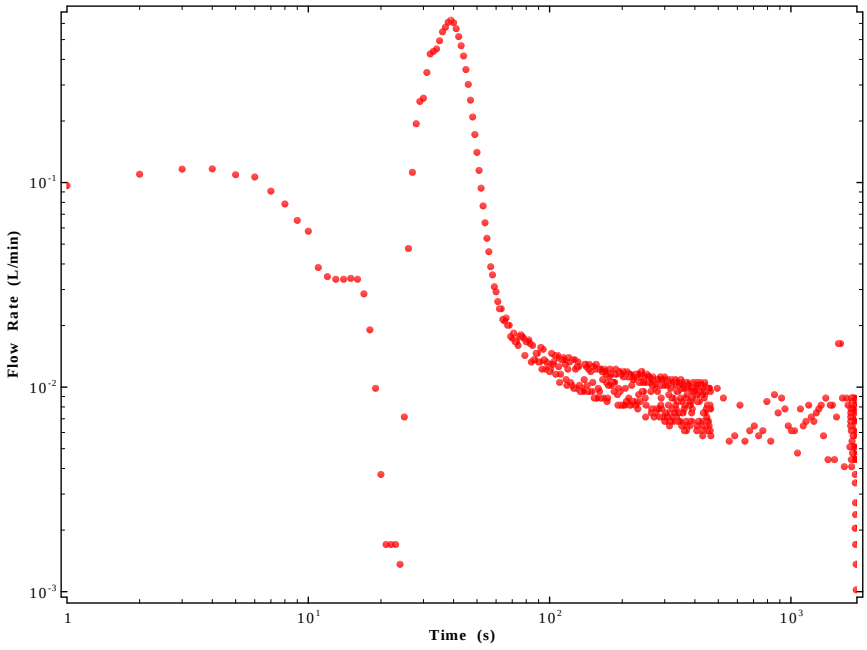

Figure 4.6: Test Problem 3: Matching Points

Chapter 5

Optimization Testing Methods

We design testing methods with the goal of answering the following three questions:

1. How does the presence of noise affect each of the optimization algorithms?
2. How do different values for optimization algorithm control parameters affect optimization, both with and without noise?
3. Do algorithmic parameter values need to be adjusted in order to better accommodate noisy optimization problems?

5.1 Comparative Performance Analysis

Our two primary performance measures are effectiveness and efficiency, as suggested by [90] and [124]. Thus, for each run of the optimization algorithms, the fit value at the termination point and the number of function evaluations required are recorded. Although the location of a global minimum for the underlying smooth problem is known for Test Problems 1a, 1b, and 1c, the distance between the termination points and the global minimum is not considered. Because these parameter identification problems are overdetermined, there could be many global minima. Thus, it is possible that a point a fair distance from the known global minimum could have a similar objective function value. Therefore, even when the location of a single global minimum is known, the use of distance as a performance measure could be misleading.

We test the three algorithms against themselves and each other. We vary a set of algorithmic parameters for each algorithm in an attempt to improve optimization results. Algorithm performance is also analyzed with respect to noise. Test Problems 1b, 1c, and 2a through 2c are noisy approximations to Test Problem 1a, so results from the noisy test problems are compared with results from Test Problem 1a to isolate noise effects. The best results for

each algorithm are then compared against one another to determine which algorithm is the most effective with the least amount of effort.

5.2 Nelder–Mead Algorithm

We vary four algorithmic parameters in the application of the NMA in order to determine which values for these parameters return better results. The reflection parameter α, expansion parameter γ, contraction parameter β, and shrinkage factor σ are the four factors under consideration. As mentioned in Section 3.1.1, the allowable ranges for the values of these parameters are $\alpha > 0$, $\gamma > 1$, $0 < \beta < 1$, and $0 < \sigma < 1$. Table 5.1 lists the commonly used values for these parameters [82] and the ranges of values that are allowed for each parameter during testing.

Table 5.1: Nelder–Mead Parameter Ranges

Parameter	Recommended Value	Tested Values	
		Lower Bound	Upper Bound
α	1.0	0.5	2.0
γ	2.0	2.0	6.0
β	0.5	0.05	0.95
σ	0.5	0.05	0.95

5.2.1 Space Sampling Technique

We employ a Latin hypercube sampling method to choose the parameter values for analysis. The random element in Latin hypercube sampling is intended to provide samples from a space without any bias. Suppose $\mathbf{a} = (a_1, a_2, \ldots, a_n)$ is a vector of optimization algorithm parameters. Each component a_i has lower and upper endpoints (l_i, u_i) on its allowable value. Let $A = \{\mathbf{a} : a_i \in (l_i, u_i), \quad i = 1, 2, \ldots, n\}$ be the set of all feasible values of $\mathbf{a}$. In order to generate s samples from A, each interval (l_i, u_i) is divided into s intervals of equal probability. We denote these intervals as $I_i^j, \quad i = 1, \ldots, n, \quad j = 1, \ldots, s$. A value a_i^1 is randomly selected from I_i^1 and so forth for $i = 1, \ldots, n, \quad j = 1, \ldots, s$. For each value of i, the s samples for each a_i^j are randomly ordered and j is renumbered from $1, 2, \ldots, s$. Then, $\mathbf{a}_j = (a_1^j, a_2^j, \ldots, a_n^j)$ for $j = 1, 2, \ldots, s$ [73]. Sandia National Laboratories' Latin hypercube sampling program LHSwin generates our samples. For more information, see [136].

We create 81 samples of the parameter space. The number 81 is chosen because it is the number of samples required for a three level full factorial experiment for the four parameters

[39]. A complete list of the samples can be found in Appendix A.

5.2.2 Testing Procedure

The NMA is applied to Test Problem 1 for each of the 81 parameter samples and all three levels of noise. We also test the vector of recommended values for the NMA algorithmic parameters. This sample has an index of 0. We use five starting points rather than one in order to avoid bias. The starting points are generated by random perturbations of the first starting point. The same five starting points are used for each run and are listed in Table 5.2. Algorithm performance is considered as a function of all five starting points so that performance is not biased by the starting point. The average objective function value at termination is recorded for the five starting points. Also, the total number of objective function evaluations is recorded. For noisy Test Problems 1b and 1c, the objective function value at the termination point is also evaluated using the non–noisy data points. In other words, we compute the least–squares fit between the smooth polynomial and the polynomial produced by optimization results. We refer to this result as the "true fit," since it measures how well the noisy optimization problem solves the underlying smooth problem. This calculation allows direct comparison of optimization results with differing levels of noise.

Table 5.2: Starting Points for Test Problem 1

Index	p_1	p_2	p_3	p_4	p_5
1	0.0694516	0.1186529	0.160918	-0.0145056	-0.0629882
2	0.0302331	-0.1339087	-0.1682731	-0.0258240	0.1897319
3	-0.1864699	-0.1639313	-0.1724107	-0.0324604	-0.1161099
4	-0.1053695	-0.1558059	0.0268050	0.0287785	-0.1293601
5	-0.1634335	-0.1834056	-0.1801854	0.0250770	0.0320980

We apply the NMA to Test Problems 2a, 2b, and 2c using the vector of recommended algorithmic parameters as well as each of the 81 parameter samples. The five starting points are listed in Table 5.3, and the average objective function value and number of simulations are again recorded for each execution of the algorithm. For noisy test problems 2b and 2c, the true fit is calculated as well.

We apply the NMA to Test Problem 3 using 500 randomly generated starting points. Extensive perturbation of starting points is a common technique for groundwater parameter identification problems [106]. For the sake of brevity, these starting points are not tabulated. The algorithm is not tested for each of the 81 parameter samples. Rather, we identify a group of likely parameters from Test Problems 1 and 2. The average objective value and

Table 5.3: Starting Points for Test Problem 2

Index	p_1	p_2	p_3
1	3.1253213	5.3393786	7.2413122
2	-0.6527531	-2.8344712	1.3604899
3	-6.0258929	-7.5722879	-1.1620783
4	8.5379366	-8.3911438	-7.3769105
5	-7.7584818	-1.2605161	-5.2249448

total number of objective function evaluations are calculated for each parameter sample. For Test Problem 3, we also record the lowest fit value at termination for each parameter sample.

5.3 Nelder–Mead/Simulated Annealing Hybrid

We apply the NMA/SA hybrid to Test Problem 3. The algorithm is not tested on Test Problems 1 and 2 because our literature search revealed that simulated annealing parameters must be carefully selected on a problem to problem basis. Thus, parameter values that perform well for Test Problems 1 or 2 are not necessarily predictors of effective parameter values for other problems. However, we are interested in the NMA/SA hybrid's performance on Test Problem 3, and whether it can outperform the NMA or SCEM. Thus, the NMA/SA hybrid is applied to Test Problem 3 using the same set of 500 starting points as the NMA.

Cooling Schedules 1 and 2 from Section 3.2.1 and their parameters are tested to determine which schedule gives the best results. Three values are considered for the reduction factor ϵ: 0.25, 0.5, and 0.95. Three values are also considered for the initial temperature T^0: 10, 100, and 500. A full factorial experiment is conducted for all combinations of the two parameters. The parameter combinations and their indices are found in Table 5.4.

For Cooling Schedule 2, the values of 10, 100, and 500 are again considered for T^0. The number of iterations with a nonzero temperature value K_{SA} is tested at 50, 100, and 200. A full factorial experiment is conducted for all combinations of the two parameters. The parameter combinations and their indices are found in Table 5.5.

For each cooling schedule, each parameter combination is applied to all 500 starting points. The average objective function value and total number of objective function evaluations are recorded, along with the lowest fit value over all 500 starting points.

Table 5.4: NMA/SA: Parameter Samples for Cooling Schedule 1

Index	T^0	ϵ
1	10	0.25
2	10	0.5
3	10	0.95
4	100	0.25
5	100	0.5
6	100	0.95
7	500	0.25
8	500	0.5
9	500	0.95

Table 5.5: NMA/SA: Parameter Samples for Cooling Schedule 2

Index	T^0	K
1	10	50
2	10	100
3	10	200
4	100	50
5	100	100
6	100	200
7	500	50
8	500	100
9	500	200

5.4 Shuffled Complex Evolution Method

We vary four control parameters in the application of the SCEM in order to determine which values for these parameters result in better results. The number of complexes p, the points per complex m, the number of "parents" per reproduction step q, and the number of evolution steps per complex β_S are the four factors under consideration. (The number of offspring per reproduction step α_S is set to a constant value of 1 as in [47].)

A previous study on the effects of altering the SCEM parameters for calibration of watershed models recommends values of $m = 2n+1$, $q = n+1$, and $\beta_S = 2n+1$, where n is the number of parameters that are being identified [49]. However, we feel that these values are so large that algorithm efficiency will be impaired. Thus, we choose smaller values for our investigation.

Table 5.6 lists the three values considered for each of these algorithmic parameters. A full factorial experiment for the four parameters is conducted on Test Problems 1 and 2 for the same sets of five starting points as listed in Tables 5.2 and 5.3. The full factorial experiment requires 81 samples of parameter combinations. The parameter samples and their indices are listed in Appendix B.

Table 5.6: SCEM Parameter Values

p	m	q	β_S
1	6	2	1
2	7	3	2
3	8	4	3

The full set of 81 samples is not applied to Test Problem 3. Instead, the best parameter combinations from Test Problems 1 and 2 are applied to the same five hundred starting points as before. We tabulate the average objective function value, total number of function evaluations, and the lowest fit value for each parameter sample.

Chapter 6

Numerical Results: The Nelder–Mead Algorithm

Complete numerical results for the NMA applied to Test Problems 1 and 2 can be found in Appendix C.

6.1 Test Problem 1

Although our results for objective function values are averages over five starting points, we will simply use the phrase *objective function values*, or *fit values*.

6.1.1 Test Problem $1a$

For Test Problem $1a$, the NMA terminates at fit values between 0.40 and 4.07×10^6. There are parameter samples that terminate with fit values at each order of magnitude between 10^{-1} and 10^6. The number of simulations required for termination varies between 747 and 11542. Table 6.1 lists the parameter samples that terminate at objective function values less than one. The number of simulations that these samples use ranges between 2159 and 3209 with an average value of 2706. The points are sorted in increasing number of simulations, as each sample requires a different number of simulations. Among the parameter samples, more computational effort does not always result in a better fit value. For example, sample 36 requires more simulations than sample 67, but results in a fit value more than twice as large. The results for parameter sample 0, the universally accepted default value for the NMA parameters, are also included in Table 6.1. Sample 0 terminates at objective functions that are three orders of magnitude larger than the other parameter samples found in Table 6.1.

Table 6.1: Test Problem 1*a*: Best NMA Parameter Combinations

Index	Number of Simulations	Objective Function
20	2159	0.578
38	2659	0.466
67	2798	0.402
36	3209	0.891
0	1695	372

Table 6.2 is a list of the parameter samples that terminate on the order of 10^6. These points require between 747 and 1743 simulations with an average of 1013. There is no overlap in the number of simulations recorded in Tables 6.1 and 6.2. Each parameter sample from Table 6.1 uses more simulations than each sample from Table 6.2.

Table 6.2: Test Problem 1*a*: Worst NMA Parameter Combinations

Index	Number of Simulations	Objective Function
70	1743	1.04×10^6
4	795	1.79×10^6
44	1026	2.18×10^6
14	747	2.35×10^6
51	832	3.67×10^6
16	937	4.07×10^6

6.1.2 Test Problem 1*b*

The NMA terminates at objective function values ranging from 2481 to 4.02×10^6 for Test Problem 1*b*. The number of simulations required varies between 740 and 10371. There are seven parameter samples that result in termination within 0.5% of the lowest fit value found. Table 6.3 is a list of these samples sorted in increasing order of number of simulations. The number of simulations varies between 1139 and 2308 with an average value of 1684. For this set of seven parameter values, the lowest value for the objective function corresponds to the lowest number of simulations. The other parameter samples in Table 6.3 use more simulations and are not as effective. The recommended parameter sample 0 is included in this table. This sample does not terminate within 0.5% of the lowest objective function value found.

Table 6.3: Test Problem 1*b*: Best NMA Parameter Combinations

Index	Number of Simulations	Objective Function	Fit With True Data Points
27	1139	2481.77	62.27
38	1283	2490.78	53.78
64	1535	2487.53	71.01
49	1751	2492.64	59.65
34	1761	2486.61	62.56
45	2010	2487.75	56.59
80	2308	2489.59	61.56
0	1750	2522.63	68.12

Table 6.4 is a list of the parameter samples that terminate on the order of 10^6. The number of simulations for these samples varies between 740 and 1352 with an average value of 931. The range of simulations required in Table 6.4 overlaps with the range from Table 6.3, but the difference in objective function values is extreme.

Table 6.4: Test Problem 1*b*: Worst NMA Parameter Combinations

Index	Number of Simulations	Objective Function	Fit With True Data Points
70	1352	1.03×10^6	1.04×10^6
4	780	1.80×10^6	1.80×10^6
44	884	2.23×10^6	2.23×10^6
14	740	2.36×10^6	2.35×10^6
51	898	3.64×10^6	3.63×10^6
16	929	4.02×10^6	4.01×10^6

Tables 6.3 and 6.4 also include the true fit values. For Test Problem 1*b*, the true fit varies between 53.78 and 4.01×10^6. The parameter samples that result in the lowest true fit values are not all tabulated in Table 6.3. Therefore, Table 6.5 is a list of the parameter samples whose true fit values are less than 60. For these points, the number of simulations varies between 1283 and 2047 with an average value of 1708. Tables 6.3 and 6.5 demonstrate that the parameter samples with the lowest fit values are not necessarily the parameters that yield the lowest true fits. However, as summarized in Table 6.4, the parameter samples that return large values for the true fit are the same parameter samples that return large values for the perturbed objective function. Thus, these parameter samples perform poorly regardless of the presence of noise.

Table 6.5: Test Problem 1*b*: Parameter Combinations Yielding Lowest True Fit Values for the NMA

Index	Number of Simulations	Objective Function	Fit With True Data Points
38	1283	2490.78	53.78
67	2047	2495.48	54.01
54	1447	2497.42	55.64
45	2010	2487.75	56.59
49	1751	2492.64	59.65
0	1750	2522.63	68.12

6.1.3 Test Problem 1*c*

For Test Problem 1*c*, the NMA terminates at objective function values between 9890 and 4.02×10^6. The number of simulations varies from 705 to 9855. There are nine parameter samples that terminate at objective functions within 0.5% of the lowest fit found. These nine samples are sorted in increasing order of required simulations in Table 6.6. For these parameter samples, the number of simulations varies between 1025 and 2168 with an average of 1520. Parameters that require a greater number of simulations do not necessarily yield lower fit values. For instance, sample 66 uses more than twice as many simulations as sample 27 and returns a higher fit value. The recommended parameter sample 0 is included in this table. This parameter sample does not result in termination within 0.5% of the lowest objective function found.

Table 6.7 is a list of the parameter samples that terminate with fit values on the order of 10^6. The number of simulations for these samples varies between 730 and 1402 with an average value of 935. This range of values overlaps with the number of simulations recorded in Table 6.6, but the difference in objective function values is extreme.

Table 6.6 includes the nine lowest recorded values for the true fit. The true fit takes on values between 6.45 and 4.01×10^6. Although the nine lowest true fit values occur at the same nine parameter samples as the lowest perturbed objective function, the lowest perturbed objective function does not correspond to the lowest true fit value. However, the lowest number of simulations from these nine samples and the lowest true fit do correspond. Table 6.7 includes the six parameter samples that terminate at the highest true fit values. For these six points, the relative order of the perturbed fit and the true fit are the same. Parameter sample 0 results in a true fit value nearly 100 times larger than the lowest true fit values found.

Table 6.6: Test Problem 1*c*: Best NMA Parameter Combinations

Index	Number of Simulations	Objective Function	Fit With True Data Points
27	1025	9934.57	6.45
50	1088	9948.89	31.98
20	1123	9968.55	50.12
38	1262	9916.18	13.77
64	1286	9969.99	21.63
67	1719	9975.97	23.72
45	1850	9973.89	25.34
66	2162	9890.95	17.21
36	2168	9930.85	27.46
0	1395	10456.29	509.76

Table 6.7: Test Problem 1*c*: Worst NMA Parameter Combinations

Index	Number of Simulations	Objective Function	Fit With True Data Points
70	1402	1.04×10^6	1.03×10^6
14	730	1.79×10^6	1.78×10^6
4	850	1.81×10^6	1.80×10^6
44	971	2.19×10^6	2.18×10^6
51	769	3.68×10^6	3.67×10^6
16	887	4.02×10^6	4.01×10^6

6.1.4 Effects Due to Noise

Table 6.8 is a summary of the NMA results when applied to Test Problems 1*a*, 1*b*, and 1*c*. The addition of noise to the data set has a noticeable impact on the NMA's performance. For Test Problems 1*b* and 1*c*, the best value for the perturbed objective function is on the order of the magnitude of the noise in the data set. The best true fit for these two problems is one to two orders of magnitude higher than the best true fit found in the unperturbed case. Thus, even for each problem's best algorithmic parameters, the perturbed optimization problems terminate at true fit values at least ten times higher than for the unperturbed problem. Although Test Problem 1*c* represents a larger data perturbation than Test Problem 1*b*, the best true fit found for 1*c* is almost ten times smaller than the best true fit from 1*b*. In fact, there are nine parameter samples that result in true fits less than 53.78 (the best true fit found for 1*b*) when applied to 1*c*. This result shows that a large perturbation of the data set

does not always cause worse results than a smaller perturbation.

The relationship between true fits and perturbed fits is important. In a noisy optimization problem (perhaps with an experimentally collected data set), the true fit is not known. At best, there may be a bound on the amount of noise in the problem. Although the perturbed objective function is known and optimized, the ultimate goal is the smallest possible value for the true fit. Ideally, a lower value for the perturbed objective function would correspond to a lower value for the true fit. However, for Test Problems 1*b* and 1*c*, there are parameters that have lower perturbed objective functions and higher true fits than other parameters. Thus, a parameter sample that yields the lowest perturbed fit value for a noisy problem is not guaranteed to yield the lowest true fit value. Another important consideration for optimization problems is the amount of effort needed to terminate. A larger amount of effort, or number of simulations, should intuitively result in a better result. However, for all three test problems, there are parameter samples that take more relative effort to return worsening results, and vice versa. Identification of parameter samples that use fewer simulations for similar results is an important general question.

Table 6.8: Test Problem 1: Comparison of Noise Effects on the NMA

	1*a*	1*b*	1*c*
Best Objective Function	0.40	2481.77	9890.95
Best True Fit	0.40	53.78	6.45
Average Number of Simulations For Best Parameter Combinations	2706	1684	1520
Least–Squares Difference Between True and Noisy Data	0	2535.10	9928.90
Good Parameter Samples	20, 38, 67, 36	27, 38, 64, 49, 34, 45, 80	27, 50, 20, 38, 64, 67, 45, 66, 36
Poor Parameter Samples	70, 4, 44, 14, 51, 16	70, 4, 44, 14, 51, 16	70, 14, 4, 44, 51, 16

Another difference in the three problems is the number of simulations required for the best parameter samples. For 1*a*, the average number of simulations refers to the parameter samples that terminate at objective function values less than one. For 1*b* and 1*c*, the average number of simulations refers to the parameter samples that terminate within 0.5% of the smallest objective function value found for each problem. The average number of simulations is higher for the unperturbed problem than for the two perturbed problems. In fact, over the 81 parameter samples, there are only nine parameters where Test Problem 1*b* or 1*c* require more simulations than Test Problem 1*a*. This is a useful trend. Since the perturbed problems terminate at higher true fit values than the unperturbed problem, the smaller number of simulations needed for the perturbed problems means that less effort is required for the less desirable result.

In all three test problems, the parameter sample 0 of recommended values does not compete well with the best parameter values. Deviating from recommended values for the NMA algorithmic parameters can save many orders of magnitude in the objective function and/or the number of simulations required. There is some overlap in the parameter samples that perform well for each problem. The four samples that do well for $1a$ also do well for $1c$. However, several parameters return low perturbed objective functions only for $1b$ or $1c$. When considering the true fit, samples 38 and 67 result in low values for both perturbed test problems. However, other parameter samples return low true fits for only one of the two problems. The same six parameter samples are the worst six performers for each of the three test problems. Clearly, there is more consistency in the poor performers than the top performers.

6.2 Test Problem 2

The NMA is applied to Test Problems $2a$, $2b$, and $2c$ using each of the 81 parameter samples.

6.2.1 Test Problem $2a$

For Test Problem $2a$, the NMA terminates at objective function values between 7590 and 1.36×10^6. Most parameter samples terminate at fit values on the order of 10^3 and 10^4. The number of simulations required for termination varies between 412 and 5622. Table 6.9 lists the parameter samples that terminate at objective function values within 0.5% of the lowest fit value found. The number of simulations that these samples use is between 934 and 3635, with an average value of 1532. The parameter samples are sorted in increasing order of simulations, and parameter samples with larger numbers of simulations do not necessarily terminate with smaller objective functions. In fact, the fit values from Table 6.9 are almost indistinguishable, but one parameter sample uses 934 simulations, while another uses 3635. For Test Problem $2a$, the recommended parameter sample 0 terminates within 0.5% of the lowest objective function found and is the most efficient parameter sample to do so.

Table 6.10 is a list of the parameter samples that terminate on the order of 10^5 and 10^6. These points require between 425 and 1183 simulations with an average of 681. These values overlap with the number of simulations recorded in Table 6.9, but the difference in objective function values is extreme.

Table 6.9: Test Problem 2*a*: Best NMA Parameter Combinations

Index	Number of Simulations	Objective Function
0	934	7590.13
20	965	7590.12
58	1096	7590.14
38	1104	7590.12
25	1121	7590.14
49	1143	7590.14
54	1190	7590.15
34	1291	7590.13
37	1299	7590.15
64	1321	7590.50
45	1335	7590.15
56	1369	7608.11
66	1521	7590.18
36	1531	7590.16
80	1616	7590.17
67	1679	7590.18
55	1804	7590.21
6	1844	7590.20
62	1845	7590.24
46	2539	7590.31
63	3635	7590.52

Table 6.10: Test Problem 2*a*: Worst NMA Parameter Combinations

Index	Number of Simulations	Objective Function
16	759	1.28×10^5
51	523	1.30×10^5
43	745	1.53×10^5
4	521	2.35×10^5
14	425	2.87×10^5
26	1183	1.21×10^6
10	608	1.36×10^6

6.2.2 Test Problem 2*b*

The NMA terminates with fit values between 9334 and 1.34×10^6 for Test Problem 2*b*. The number of simulations required varies between 463 and 6110. There are thirteen parameter samples that result in fit values at termination within 0.5% of the lowest objective function value found. Table 6.11 is a list of these samples. The number of simulations varies between 1034 and 3618 with an average value of 1474. Some parameters use more simulations than others, yet still result in larger fit values. The recommended parameter sample 0 does terminate within 0.5% of the lowest objective function found and is an efficient performer for this problem.

Table 6.11: Test Problem 2*b*: Best NMA Parameter Combinations

Index	Number of Simulations	Objective Function	Fit With True Data Points
0	1034	9334.96	7635.55
20	1044	9334.92	7635.29
38	1049	9334.94	7635.46
25	1144	9335.03	7637.16
54	1165	9334.95	7635.80
29	1181	9334.96	7636.10
34	1182	9335.02	7635.49
49	1365	9334.95	7635.73
19	1379	9334.95	7635.49
64	1543	9339.57	7652.55
36	1556	9334.99	7635.03
62	1898	9335.12	7635.56
63	3618	9335.63	7634.15

Table 6.12 is a list of the parameter samples that terminate with fit values on the order of 10^5 or higher. The number of simulations for these samples vary between 519 and 1124 with an average value of 652. These values overlap with the number of simulations recorded in Table 6.11, but the difference in objective function values is extreme.

Tables 6.11 and 6.12 also include the true fit. For Test Problem 2*b*, the true fit varies between 7634 and 1.35×10^6. The parameter samples that result in the lowest true fits are all tabulated in Table 6.11. In fact, Table 6.11 contains all parameter samples whose true fit values are within 0.5% of the lowest true fit found. The parameter sample that yields the lowest true fit value is not the parameter sample that yields the lowest perturbed fit. The parameter samples that return the largest values for the true fit are the same parameter samples that return large values for the perturbed objective function.

Table 6.12: Test Problem 2*b*: Worst NMA Parameter Combinations

Index	Number of Simulations	Objective Function	Fit With True Data Points
16	591	1.25×10^5	1.25×10^5
43	642	1.33×10^5	1.32×10^5
51	534	1.56×10^5	1.53×10^5
4	499	2.34×10^5	2.34×10^5
26	1124	1.21×10^6	1.21×10^6
10	519	1.34×10^6	1.35×10^6

6.2.3 Test Problem 2*c*

For Test Problem 2*c*, the NMA terminates with objective function values between 18462 and 1.37×10^6. The number of simulations varies from 422 to 5235. There are thirteen parameter samples that terminate at fit values within 0.5% of the lowest objective function found. These thirteen samples are sorted in increasing order of required simulations in Table 6.13. For these parameter samples, the number of simulations varies between 955 and 1970 with an average of 1376. Parameters using more simulations do not always cause a decrease in the objective function value. The recommended parameter sample 0 is included in this table and does result in termination within 0.5% of the lowest objective function found. It is also one of the more efficient parameter samples.

Table 6.14 is a list of the parameter samples that terminate on the order of 10^5 or higher. The number of simulations for these samples vary between 464 and 1181 with an average value of 669. These values overlap with the number of simulations recorded in Table 6.11, but the difference in objective function values is extreme.

Table 6.13 includes the thirteen lowest recorded values for the true fit. The true fit has values between 7658 and 1.37×10^6. Although the thirteen lowest true fit values occur at the same thirteen parameter samples as the lowest perturbed objective function, the parameter with the lowest perturbed objective function is not the parameter with the lowest true fit value. Table 6.14 includes the eight highest values for the true fit. For these eight points, the relative order of the perturbed fit and the true fit are the same.

Table 6.13: Test Problem 2*c*: Best NMA Parameter Combinations

Index	Number of Simulations	Objective Function	Fit With True Data Points
20	955	18462.88	7659.37
0	982	18462.90	7659.17
38	999	18462.95	7658.55
54	1131	18463.06	7664.34
58	1151	18463.01	7658.70
37	1292	18462.88	7660.47
49	1314	18462.95	7660.82
45	1349	18462.95	7661.71
66	1503	18463.16	7659.03
36	1549	18463.11	7662.23
62	1794	18463.27	7663.85
6	1899	18463.30	7660.65
65	1970	18463.21	7660.61

Table 6.14: Test Problem 2*c*: Worst NMA Parameter Combinations

Index	Number of Simulations	Objective Function	Fit With True Data Points
44	558	1.03×10^5	9.32×10^4
51	542	1.36×10^5	1.30×10^5
16	561	1.38×10^5	1.28×10^5
43	944	1.64×10^5	1.55×10^5
4	506	2.43×10^5	2.35×10^5
14	464	2.95×10^5	2.86×10^5
26	1181	1.21×10^6	1.21×10^6
10	599	1.37×10^6	1.37×10^6

6.2.4 Effects Due to Noise

Table 6.15 is a summary of the NMA results when applied to Test Problems 2*a*, 2*b*, and 2*c*. The impact of changing from the fifth degree polynomial model of the first set of test problems to the third polynomial model is immediately apparent. For the unperturbed Test Problem 2*a*, the smallest objective function at termination of the NMA is 7590.12. Although this value is not necessarily the smallest possible objective function for 2*a*, the value is very far from zero, thus indicating a large residual problem. The addition of noise to the data set has a noticeable impact on the NMA's performance. The objective function values at termination for the perturbed Test Problems 2*b* and 2*c* are noticeably larger than in 2*a*. In fact, for 2*b* and 2*c*, the best value for the perturbed objective function is on the order of the magnitude of the noise in the data set added to the best objective function found for the unperturbed problem 2*a*. The best true fits for 2*b* and 2*c* are very close to the best fit from 2*a*. The true fits for 2*b* and 2*c* are only 0.58% and 0.90% larger than 7590.12. However, all the true fit values are many times larger than the lowest true fit value for Test Problem 1*a*. For these test problems, the larger perturbation in the data set corresponds with a larger difference in the true fit at termination.

As noted for Test Problem 1, when the NMA is applied to Test Problems 2*b* and 2*c*, there are parameters where the perturbed objective function decreases and the true fit increases. Again, for all three test problems, there are parameter samples that take more relative effort to return worsening results, and vice versa.

Table 6.15: Test Problem 2: Comparison of Noise Effects on the NMA

	2*a*	2*b*	2*c*
Best Objective Function	7590.12	9334.92	18462.88
Best True Fit	7590.12	7634.15	7658.55
Average Number of Simulations for Best Parameter Combinations	1532	1474	1376
Sum Squared Difference Between True and Noisy Data	0	2535.10	9928.90
Good Parameter Samples	0, 6, 20, 25, 34, 36, 37, 38, 45, 46, 49, 54, 55, 56, 58, 62, 63, 64, 66, 67, 80	0, 19, 20, 25, 29, 34, 36, 38, 49, 54, 62, 63, 64	0, 6, 20, 36, 37, 38, 45, 49, 54, 58, 62, 65, 66
Poor Parameter Samples	4, 10, 14, 16, 26, 43, 51	4, 10 16, 26, 43, 51	4, 10, 14, 16, 26, 43, 44, 51

Another difference in the three problems is the number of simulations required for the best parameter samples. For each problem, the average number of simulations refers to the

parameter samples that terminate within 0.5% of the smallest objective function value found for each problem. The average number of simulations is higher for the unperturbed problem than for the two perturbed problems. Among all parameter samples and the three test problems, there are only seven instances where one of the three noisy problems require more simulations than Test Problem 1*a*. The four parameter samples that perform well for Test Problem 1*a* also perform well for at least one of Test Problems 2*a*, 2*b*, and 2*c*. Parameter samples 20, 38, and 36 perform well for all four problems.

In all three test problems, the parameter sample 0 of recommended values is one of the best parameter samples. There are many parameter samples that terminate within 0.5% of the smallest objective found for each of the three problems, and most of the samples do well for at least two of the three problems. There are six parameter samples that are among the worst performers for each of the three problems. Each of the three test problems require a smaller average number of simulations than Test Problem 1*a*.

6.3 Test Problem 3

Test Problem 3 requires significantly more computational time than Test Problems 1 or 2 due to the complexity of the model and the number of starting points. Thus, Test Problem 3 is not optimized using all 81 parameter samples. We select parameter samples based on their performance in Test Problems 1 and 2.

6.3.1 Parameter Selection

There is an unquantified amount of noise in Test Problem 3, including perturbations in the data set, model structure error, and error caused by our numerical approximation to the model. Thus, we know that the objective function differs from the true fit. However, the calculated objective function is the only indication we have of how effectively an algorithm performs. When selecting parameters to test the NMA's performance on Test Problem 3, we consider parameters that returned good results for Test Problems 1 and 2. Since Test Problems 2*b* and 2*c* share many of the noise sources in Test Problem 3, more emphasis is placed on parameter values that efficiently result in low true fit values for these problems.

Parameter samples 20, 38, and 54 are included because they are efficient and effective for 2*b* and 2*c*, as well as several other problems. Parameter sample 0 is also effective and efficient for these two test problems, but the primary reason for including this sample is to compare algorithm performance against the standard parameter values. Parameter samples 34, 36, 62, and 63 are all effective for 2*b* and/or 2*c*, but not as efficient as previous choices. Parameter samples 27, 45, 64, and 67 return low objective functions for 1*b* and 1*c*, so they are also included. Parameter sample 4 performs poorly for each of the six test problems, so it is included to determine if it also does poorly on Test Problem 3. These 13 parameter values

are the only values tested for Test Problem 3.

6.3.2 Numerical Results

For all parameter sample and starting point combinations, the smallest objective function found by the NMA for Test Problem 3 is 17.48151. As Table 6.16 shows, most parameter samples terminate at this lowest objective function value for at least one starting point. The average objective function values for the thirteen parameter samples vary from 18.92 to 30.19. The total number of simulations varies from 53,427 to 210,344. Only three parameter samples require fewer than 100,000 simulations. These three samples (4, 27, and 64) also result in termination at the three largest objective function values.

Eight of the parameter samples result in lower average objective functions than the recommended parameter 0. However, only one of these eight parameter samples uses fewer simulations than sample 0. The average number of simulations for the parameter samples that terminate at lower values than sample 0 is 130,424. Parameter sample 54 has the lowest average objective function at termination. When compared with parameter sample 0, sample 54 results in a 7.7% smaller average objective function with only a 1.7% increase in the number of simulations required. In addition, when compared with sample 0, parameter sample 20 results in a 4.8% decrease in average objective function value using 2.6% fewer simulations. Parameter sample 63 terminates at an average objective function that is 5.4% smaller than the value for sample 0. However, the number of simulations for sample 63 is more than double the number needed for sample 0.

Table 6.16: Test Problem 3: NMA Results

Index	Average Objective Function	Total Number of Simulations	Lowest Objective Function
0	20.49	104479	17.48151
4	30.19	53472	17.48528
20	19.50	101799	17.48151
27	23.44	84363	17.48151
34	19.23	111586	17.48151
36	19.02	124735	17.48151
38	20.66	108627	17.48151
45	19.69	118451	17.48152
54	18.92	106249	17.48151
62	19.97	144628	17.48152
63	19.38	210344	17.48151
64	22.28	85731	17.48152
67	19.54	125596	17.48151

6.4 Selecting NMA Parameters for Noisy Optimization

Test Problems 1, 2, and 3 demonstrate that adjustment of the NMA algorithmic parameters can result in profound differences in the final objective function values and number of simulations required. The large differences in outcome occur for the non–noisy problem $1a$ as well as the perturbed problems. Table 6.17 lists eight of the parameter samples that perform well for a several of the test problems as well as the currently accepted values for the parameter samples.

The values for β vary between 0.265 and 0.873. This is a large range of values, so the data does not suggest any particular effective value or range of value for β. The values for σ also range widely from 0.226 to 0.934. Again, the data does not suggest any values for σ. The values for α vary from 0.885 to 1.15. This range includes the recommended value of 1 for α. Indeed, every value in this range is within 15% of the recommended value. Thus, the numerical results support using a value near 1 for α. The values for γ vary from 3.52 to 5.84. The recommended value for γ is 2. This value is significantly lower than the lowest value for γ among the competitive parameter samples. The numerical results from Test Problems 1 through 3 indicate that increasing γ to 3.5 or more can improve the performance of the NMA

Table 6.17: Competitive NMA Parameter Combinations

Index	α	β	γ	σ
0	1	0.5	2	0.5
20	1.02	0.417	3.62	0.504
34	1.03	0.511	3.52	0.934
36	0.968	0.635	4.21	0.326
38	0.919	0.438	4.57	0.226
45	1.15	0.594	3.73	0.375
54	1.14	0.463	3.95	0.608
62	0.885	0.746	3.91	0.337
63	0.931	0.873	3.72	0.427
64	0.989	0.265	5.80	0.791
67	0.955	0.618	5.84	0.389

for both noisy and non–noisy optimization. A larger value for γ corresponds to searching a greater distance along a previously identified decrease direction. Ideally, an optimization algorithm would be able to exhaust the decrease direction by taking the largest possible step. The NMA only searches at a set distance along the decrease direction, however, and γ controls that distance. Thus, increasing γ results in the consideration of a new point that is farther along the decrease direction. There is a greater chance that the NMA will reject the new point when γ is larger, since the new point may be beyond the decrease direction's borders. However, if the point is not rejected, a larger value for γ results in a larger step in the decrease direction, and thus a greater improvement in fit value.

There are also several parameter samples that perform poorly for several of the test problems. Table 6.18 is a list of the worst parameter samples. The values for α are all less than 0.837 or greater than 1.30. There is no overlap between α values in the competitive parameter values and poor parameter values. This trend reinforces the idea that the "best" value for α is between 0.885 and 1.15. The values for β from Table 6.18 are all less than 0.175 or greater than 0.89. Again, there is no overlap between Tables 6.17 and 6.18 for β values. The values for γ are all less than 3.36 or greater than 4.38. There is overlap between the two tables for the values of γ. However, there are no γ values between 3.5 and 4.38 on Table 6.18. Therefore, increasing γ to a value between 3.5 and 4.38 is not contradicted. Finally, the values for σ vary between 0.203 and 0.949. There are similar σ values in both tables. These results suggest that σ does not have a strong impact on the outcome of the NMA. Shrink steps have been demonstrated to occur rarely in the course of NMA optimization [122]. The parameter σ is only used in the case of a shrink step. Given the rarity of shrink steps, it is not surprising that our numerical results demonstrate an indifference to the value of σ.

Table 6.18: Poor NMA Parameter Combinations

Index	α	β	γ	σ
0	1	0.5	2	0.5
4	1.35	0.0982	4.83	0.278
10	0.54	0.0764	4.75	0.203
14	1.64	0.125	4.38	0.371
16	1.91	0.109	2.21	0.949
26	0.524	0.894	2.03	0.54
43	1.68	0.175	4.95	0.715
44	1.89	0.137	3.27	0.406
51	1.83	0.0902	3.36	0.586
70	0.837	0.0508	2.76	0.835

Chapter 7

Numerical Results: The Nelder–Mead/Simulated Annealing Hybrid

The NMA parameters α, β, γ, and σ are not varied during testing of the NMA/SA. Instead, these values are set as constants. The values are chosen to correspond to the NMA parameter sample 67 due to the promising performance of these parameter values for most test problems. Thus, for testing of the NMA/SA hybrid, $\alpha = 0.955$, $\beta = 0.618$, $\gamma = 5.84$, and $\sigma = 0.389$.

7.1 Test Problem 3

7.1.1 Cooling Schedule 1

The NMA/SA hybrid combined with Cooling Schedule 1 is applied to Test Problem 3. The nine samples for cooling parameter values found in Table 5.4 are each used for each of the 500 starting points. Table 7.1 is a summary of the optimization results. The average objective function values at termination vary from 18.58 to 19.79 with an average value of 19.06. The number of simulations required for termination varies from 119899 to 193717 with an average value of 142105. The smallest objective function at termination for any parameter sample or starting point is 17.48151, and seven of the nine parameter samples terminate at this value for at least one of the 500 starting points.

For Cooling Schedule 1, a larger value for ϵ corresponds to a faster cooling rate. When the initial temperature value is held constant, changing the value of ϵ changes the outcome and amount of effort required for optimization. For constant initial temperature values, an increase in the value of ϵ always causes a smaller number of simulations in the course of

Table 7.1: Test Problem 3: NMA/SA and Cooling Schedule 1 Results

Index	Average Objective Function	Number of Simulations	Lowest Objective Function
1	18.92	152639	17.48151
2	19.79	123883	17.48151
3	19.55	119899	17.48151
4	18.82	177279	17.48152
5	18.58	127168	17.48152
6	19.69	121058	17.48151
7	18.60	193717	17.48151
8	18.93	138115	17.48151
9	18.64	125184	17.48151
NMA 67	19.54	125596	17.48151

optimization. However, the value of the objective function at termination does not always increase or decrease when ϵ is increased. A different result holds when the initial temperature is varied with constant values of ϵ. The number of simulations increases as the initial temperature value increases. There is not a corresponding trend in the objective function values; fit values increase or decrease when the initial temperature value increases. The smallest objective function value occurs for $\epsilon = 0.5$ and $T^0 = 100$. The smallest number of simulations needed occurs when $\epsilon = 0.95$ and $T^0 = 10$. This combination of parameters includes the largest value for ϵ and the smallest value for T^0, as would be expected from the trends observed.

The results for the NMA using parameter sample 67 are included in Table 7.1 to facilitate comparison between the NMA and NMA/SA hybrid. Six of the nine parameter samples tested for the NMA/SA hybrid terminate at lower average objective function values than the NMA. Only five of the nine samples require more simulations. Of the six parameter samples that terminate at lower average objective function values than the NMA, five require more simulations than the NMA. Parameter sample 9 uses fewer simulations than the NMA and terminates at a lower average objective function value. Parameter sample 5, which terminates at the lowest average objective function, terminates at an average objective function value that is 4.9% smaller than the NMA's average objective function with 1.3% more simulations. Thus, the addition of simulated annealing theory to the NMA can improve optimization results.

7.1.2 Cooling Schedule 2

The NMA/SA hybrid combined with Cooling Schedule 2 is also applied to Test Problem 3. The nine samples for cooling parameter values found in Table 5.5 are each used for each of the 500 starting points. Table 7.2 is a summary of the optimization results. The average objective function values at termination vary from 17.73 to 20.80 with an average value of 19.22. The number of simulations ranges from to 132841 to 237494, with an average value of 184227. The smallest objective function at termination for any parameter sample or starting point is 17.48150. Only two parameter samples terminate at this value for at least one of the 500 starting points.

Table 7.2: Test Problem 3: NMA/SA and Cooling Schedule 2 Results

Index	Average Objective Function	Number of Simulations	Lowest Objective Function
1	20.80	132841	17.48151
2	17.73	160417	17.48150
3	18.25	228685	17.48151
4	19.28	146953	17.48151
5	18.26	176831	17.48152
6	18.13	237494	17.48150
7	20.19	150868	17.48151
8	20.48	186653	17.48151
9	19.89	237303	17.48152
NMA 67	19.54	125596	17.48151

For Cooling Schedule 2, a smaller value for K_{SA} corresponds to a faster cooling rate. When the initial temperature value is held constant, changing the value of K_{SA} changes the outcome and amount of effort required for optimization. Increasing the value for K_{SA} while holding the initial temperature constant always results in an increase in the number of simulations required. However, there is not a relationship between increases in K_{SA} and the average objective function value at termination. When K_{SA} increases, the average objective function value at termination may increase or decrease. Another result holds when the initial temperature is varied with constant values of K_{SA}. An increase in initial temperature results in an increase in the number of simulations required in all cases except for one. The value of the average objective function at termination may increase or decrease when the initial temperature is increased. The lowest average objective function value at termination occurs when $K_{SA} = 100$ and $T^0 = 10$. The smallest number of required simulations occurs for $K_{SA} = 50$ and $T^0 = 10$. These are the smallest values for the parameters, and the trends

we observed predict this outcome.

Every parameter sample requires more simulations than the corresponding NMA optimization run. However, only five of the nine parameter samples terminate at lower average objective function values than the NMA. The four parameter samples that terminate at higher average objective function values than the NMA are significantly more costly than the NMA. Not only will optimization runs be more expensive and less useful, but the extra programming time will also raise the cost. For parameter samples 2 and 6 and at least one starting point, the NMA/SA terminates at an objective function value that is lower than any value found for this problem using the NMA. Also, parameter sample 2 terminates at an average objective function value of 17.73, which is lower than the corresponding NMA result by almost 10%. The improvement in efficiency comes at an extra computational cost of 28% more simulations, however.

7.2 Comparison of Cooling Schedules

Of all the 18 parameter samples that are tested for Cooling Schedules 1 and 2, the smallest average objective function value of 17.73 occurs for sample 2 in Cooling Schedule 2. Also, the Cooling Schedule 2 parameter samples 2 and 6 terminate at 17.48150 for at least one starting point. None of the parameter samples for Cooling Schedule 1 terminate at this value. Cooling Schedule 2 has a larger average number of simulations than Cooling Schedule 1. Although Cooling Schedule 2 returns the lowest average objective function value, there are also four parameter samples for Cooling Schedule 2 that terminate at higher average objective function values than the largest value for Cooling Schedule 1. Thus, Cooling Schedule 1 is more consistent than Cooling Schedule 2.

For both cooling schedules, increasing the rate of cooling results in a smaller number of simulations. A quick cooling rate means that the NMA/SA will have smaller perturbations in the fit values at an earlier time, thus resembling the NMA at an earlier stage. Although the NMA/SA does allow the computational inefficiency of uphill steps, our results for both cooling schedules demonstrate that uphill steps can improve results without gross compromises in efficiency.

The results for the NMA/SA emphasize the fact that cooling schedules and their parameter values greatly impact the outcome of optimization. A poor choice of parameter values can cause optimization to be more expensive and less efficient. As discussed earlier, parameter values must be carefully calibrated for each problem when simulated annealing is used.

Chapter 8

Numerical Results: The Shuffled Complex Evolution Method

A complete list of numerical results for the SCEM applied to Test Problems 1 and 2 can be found in Appendix D.

8.1 Test Problem 1

The SCEM is applied to Test Problems $1a$, $1b$, and $1c$ for each of the 81 algorithmic parameter samples.

8.1.1 Test Problem $1a$

For Test Problem $1a$, the order of magnitude of the fit value at termination varies between 10^{-19} and 10^{4} for different parameter samples. There are a number of parameter samples that cause the algorithm to terminate at objective functions that are between the two extreme values. The number of simulations required varies from 2507 to 25005. Table 8.1 lists the results for the parameter samples that terminate at objective functions on the order of 10^{-19} sorted in increasing order of objective function value. The 7 parameter values that terminate on the order of 10^{-19} use between 13857 and 17474 function evaluations, with an average value of 15189. Parameter combinations that use a higher number of simulations do not always result in lower fit values when compared with other parameter values.

Table 8.2 lists the parameter samples that terminate at objective functions greater than 1500. The parameter samples that return fit values over 1500 appear to have two causes for their poor performance. Several of the samples require fewer than 6000 fit evaluations. This number is much smaller than the results from Table 8.1. The other poorly performing

Table 8.1: Test Problem 1a: Best SCEM Parameter Combinations

Index	Number of Simulations	Objective Function	Premature Termination Instances
76	16555	5.34×10^{-19}	0
72	13857	5.53×10^{-19}	0
80	14584	6.53×10^{-19}	0
69	17474	7.21×10^{-19}	0
81	14968	8.01×10^{-19}	0
63	14586	8.97×10^{-19}	0
79	14298	9.01×10^{-19}	0

parameters require a large number of function evaluations. For these parameters, the algorithm is forced to terminate after 5000 function evaluations (premature termination) for at least one of the five starting points. If the algorithm were allowed to run until termination criteria (3.71) is met, the fit values might be lower. However, since optimization algorithms are not allotted infinite time, and since most parameter samples do not require 5000 fit evaluations per starting point, the parameter samples that are prematurely terminated stand out as exceptionally inefficient.

Table 8.2: Test Problem 1a: Worst SCEM Parameter Combinations

Index	Number of Simulations	Objective Function	Premature Termination Instances
12	24378	1.82×10^{3}	4
21	25005	1.92×10^{3}	5
5	4766	2.63×10^{3}	0
2	15358	3.47×10^{3}	2
11	25005	4.25×10^{3}	5
4	3215	4.78×10^{3}	0
28	5737	4.91×10^{3}	0
10	3420	3.58×10^{3}	0

8.1.2 Test Problem 1*b*

For Test Problem 1*b*, the objective function values at termination range from 2470 to 1.4×10^6. The number of simulations required varies between 1908 and 25005. There is a marked cluster of termination points that are close in value to the best objective function value found. A total of 40 samples terminate at points where the objective function is within 0.5% of the lowest value found. The average number of simulations required over these 40 points is 4882, with a maximum value of 9660 and a minimum value of 1908. As in Test Problem 1*a*, a greater number of simulations does not always imply a lower objective function value. Since there are so many points within 0.5% of the lowest value found, the efficiency measure is a large factor in overall best performance. Table 8.3 is a list of the sample values that terminate within 0.5% of the smallest objective function and use less than 3200 simulations, sorted in increasing order of simulations. Table 8.4 is a list of samples that terminate at objective function values greater than 6000. As in Test Problem 1*a*, there are algorithmic parameters that are halted after reaching 5000 iterations. However, there are 5 parameter samples that meet the termination criteria in equation (3.71) at objective function values that are very large in comparison with other parameter samples.

Table 8.3: Test Problem 1*b*: Best SCEM Parameter Combinations

Index	Number of Simulations	Objective Function	Premature Termination Instances	Fit With True Data Points
16	1908	2477.56	0	61.91
9	2240	2481.25	0	55.51
36	2866	2477.71	0	58.65
40	2974	2475.21	0	60.30
35	3036	2479.32	0	62.35
34	3176	2472.97	0	59.74

Tables 8.3 and 8.4 also include the true fit. The true fits vary from 55.51 to over 1.4×10^6. The parameter values that terminate at low values for the perturbed fit are not necessarily the parameter values that terminate at low values for the true fit. Table 8.5 is a list of the parameter samples that terminate with a true fit less than 59 and fewer than 4500 simulations. Every point in this table terminates within 0.5% of the lowest perturbed fit value. However, the samples that result in the lowest value for the perturbed fit are different from the samples that result in the lowest value for the unperturbed fit. The poorly performing parameter values are extremely consistent between the true fit and perturbed fit. Table 8.4 illustrates the fact that the same parameters perform poorly for both fits in the same relative positions.

Table 8.4: Test Problem 1*b*: Worst SCEM Parameter Combinations

Index	Number of Simulations	Objective Function	Premature Termination Instances	Fit With True Data Points
3	23181	6.52×10^3	4	3.57×10^3
12	23443	7.56×10^3	4	4.79×10^3
1	2892	1.98×10^4	0	1.69×10^4
10	4237	5.19×10^4	0	4.86×10^4
14	3453	1.00×10^5	0	9.69×10^4
19	4676	1.14×10^6	0	1.13×10^6
13	2766	1.43×10^6	0	1.42×10^6

Table 8.5: Test Problem 1*b*: Parameter Combinations Yielding Lowest True Fit Values for the SCEM

Index	Number of Simulations	Objective Function	Premature Termination Instances	Fit With True Data Points
9	2240	2481.25	0	55.51
31	4013	2477.60	0	57.18
43	3934	2474.40	0	58.42
36	2866	2477.71	0	58.65
62	4240	2474.90	0	58.75
53	4020	2476.99	0	58.80

8.1.3 Test Problem 1*c*

For Test Problem 1*c*, the average fit values at termination vary between 9752 and 90903. The number of simulations required varies between 1911 and 25005. There are 25 sample points that terminate within 0.5% of the lowest objective function found. These points use an average of 6301 simulations with 3661 and 7687 being the minimum and maximum number needed. Table 8.6 lists the numerical results for the sample values that terminate within 0.5% of the lowest point found after fewer than 5000 iterations. As parameter samples 52 and 54 illustrate, a parameter sample using more simulations than another does not guarantee a smaller fit value. Table 8.7 displays the parameter samples that terminate with objective function values greater than 14000. Again, several parameters are so ineffective that the algorithm is prematurely terminated.

Table 8.6: Test Problem 1*c*: Best SCEM Parameter Combinations

Index	Number of Simulations	Objective Function	Premature Termination Instances	Fit With True Data Points
36	3661	9799.06	0	74.79
52	4153	9778.49	0	80.11
54	4785	9789.84	0	68.32
61	4885	9764.57	0	96.76

Table 8.7: Test Problem 1*c*: Worst SCEM Parameter Combinations

Index	Number of Simulations	Objective Function	Premature Termination Instances	Fit With True Data Points
3	21438	1.48×10^4	3	4.82×10^3
12	25005	1.54×10^4	5	5.43×10^3
27	2454	3.58×10^4	0	2.57×10^4
7	2518	5.03×10^4	0	4.07×10^4
10	3639	9.09×10^4	0	8.12×10^4

The parameters that perform best for the true fit are not the best performers for the perturbed fit. Table 8.8 shows that for some of the sample parameters, the true fit can reach values as low as 15.19 in fewer than 2000 simulations. However, the five parameter samples that result in the smallest true fit are not within 0.5% of the lowest objective function value found for the perturbed problem. The same parameters perform poorly for both fits in the same relative positions.

Table 8.8: Test Problem 1*c*: Parameter Combinations Yielding Lowest True Fit Values for the SCEM

Index	Number of Simulations	Objective Function	Premature Termination Instances	Fit With True Data Points
25	1986	9929.40	0	15.19
41	4071	9876.08	0	21.09
17	2133	9902.61	0	22.02
28	4207	9935.01	0	26.34
18	2766	9937.94	0	31.88

8.1.4 Effects Due to Noise

Table 8.9 is a summary of the SCEM results when applied to Test Problems 1*a*, 1*b*, and 1*c*. The addition of noise to the data set has a noticeable impact on the outcome of the SCEM. For Test Problems 1*b* and 1*c*, the best value for the perturbed objective function is on the order of the magnitude of the noise in the data set. However, the best true fit for these two problems is twenty orders of magnitude higher than the best true fit found in the unperturbed case. Thus, even for each algorithm's best algorithmic parameters, the perturbed optimization problems terminate at true fit values many times higher than for the unperturbed problem. Although Test Problem 1*c* represents a larger data perturbation than Test Problem 1*b*, the best true fit found for 1*c* is almost four times smaller than the best true fit from 1*b*. In fact, for Test Problem 1*c*, there are twenty parameter samples that result in true fits less than 55.51, the best true fit found for 1*b*. This same trend occurred when the NMA was applied to Test Problem 1. Since both data sets are randomly perturbed by noise, it is possible that the noise perturbation for 1*b* has a less favorable structure than the noise perturbation for 1*c*, even though the magnitude of the noise in 1*c* is higher.

For Test Problems 1*b* and 1*c*, there are parameter samples where relative perturbed fit values decrease and the true fit values increase. Thus, for this test problem and the SCEM, there is not a direct and dependable correlation between the perturbed objective function and the true fit values. Also, for all three test problems, there are parameter samples that take more relative effort to return poorer results, and vice versa.

Another difference in the three problems is the number of simulations required for the best parameter samples. For 1*a*, the average number of simulations refers to the parameter samples that terminate on the order of 10^{-19}. For 1*b* and 1*c*, the average number of simulations refers to the parameter samples that terminate within 0.5% of the smallest objective function value found for each problem. The average number of simulations is higher for the

Table 8.9: Test Problem 1: Comparison of Noise Effects on the SCEM

	$1a$	$1b$	$1c$
Best Objective Function	5.34×10^{-19}	2470.04	9752.65
Best True Fit	5.34×10^{-19}	55.51	15.19
Average Number of Simulations for Best Parameter Combinations	15189	4882	6301
Sum Squared Difference Between True and Noisy Data	0	2535.10	9928.90
Good Parameter Samples	63, 69, 72, 76 ,79, 80, 81	9, 16, 28, 31, 34, 35, 36, 40, 43, 53, 62	17, 18, 25, 28, 36, 41, 52, 54, 61
Poor Parameter Samples	2, 4, 5, 10, 11, 12, 21, 28	1, 3, 10, 12, 13, 14, 19	3, 7, 10 12, 27

unperturbed problem than for the two perturbed problems. In fact, for the parameter samples that perform well for each individual test problem, $1a$ requires almost three times the average number of simulations needed for $1b$ and $1c$. Among the 81 parameter samples, only 12 parameters required more simulations for Test Problem $1b$ or $1c$ than for $1a$. The test problem that results in the best true fit values requires the highest average number of simulations for good results.

The addition of noise to Test Problem $1a$ has a marked impact on which parameters perform well. There is no overlap among the best performing parameters for Test Problems $1a$, $1b$, and $1c$. Thus, parameter samples that perform well in the non–noisy case are not good predictors for good results in the noisy cases. There is a small amount of overlap in the parameters that perform poorly for the three test problems. Parameter samples 10 and 12 perform poorly for each of the three test problems. However, parameter sample 28 is one of the poor performers for Test Problem $1a$ and one of the top performers for Test Problems $1b$ and $1c$. Thus, the addition of noise to the optimization problem causes major differences in which parameters perform well.

8.2 Test Problem 2

The SCEM is applied to Test Problems $2a$, $2b$, and $2c$ for each of the 81 algorithmic parameter samples.

8.2.1 Test Problem $2a$

The SCEM terminates with objective function values between 7590 and 1.47×10^5 when applied to Test Problem $2a$. The number of simulations varies between 1977 and 25005. There are 68 parameter samples that result in termination within 0.5% of the smallest objective function found. For these 68 samples, the average number of simulations required is 3283. Table 8.10 lists the parameter samples that terminate within 0.5% of 7590 with fewer than 2100 simulations. Parameter samples that use a greater number of simulations than other samples do not always result in a lower objective function value. Table 8.11 lists the parameter samples that terminate at objective function values greater than 15000. Several parameter values in this group require premature termination. All of the parameter samples in this table use more simulations than the parameter samples in Table 8.10. However, the difference in fit values is an increase by a factor of two or more.

Table 8.10: Test Problem $2a$: Best SCEM Parameter Combinations

Index	Number of Simulations	Objective Function	Premature Termination Instances
16	1977	7590.10	0
31	2009	7590.13	0
35	2029	7590.12	0
34	2044	7590.15	0
33	2051	7590.12	0
32	2064	7590.13	0

Table 8.11: Test Problem $2a$: Worst SCEM Parameter Combinations

Index	Number of Simulations	Objective Function	Premature Termination Instances
12	21701	1.50×10^4	3
3	23265	1.84×10^4	4
1	2363	2.40×10^4	0
2	10348	3.25×10^4	0
10	3237	1.47×10^5	0

8.2.2 Test Problem 2*b*

For Test Problem 2*b*, the smallest fit value found is 9334, and the largest is 32,608. The required number of simulations varies between 1966 and 25005. There are 68 parameter samples that terminate within 0.5% of 9334. Of these 68 samples, the average number of simulations is 3559. Table 8.12 tabulates the parameter samples that terminate within 0.5% of 9334 in fewer than 2100 simulations. Parameter samples that use a greater number of simulations than other samples do not always result in a lower objective function value.

Table 8.13 lists the parameter combinations that terminate with objective function values over 20,000. Premature termination is a factor for two of the parameter values in this table. All of the parameter samples in Table 8.13 require more simulations than all of the parameter samples in Table 8.12 for at least double the fit value.

Table 8.12: Test Problem 2*b*: Best SCEM Parameter Combinations

Index	Number of Simulations	Objective Function	Premature Termination Instances	Fit With True Data Points
32	1967	9334.95	0	7635.70
31	1979	9334.94	0	7635.74
35	1993	9334.94	0	7634.47
26	1998	9334.90	0	7636.04
34	2015	9334.96	0	7637.27
25	2090	9334.90	0	7635.86

Table 8.13: Test Problem 2*b*: Worst SCEM Parameter Combinations

Index	Number of Simulations	Objective Function	Premature Termination Instances	Fit With True Data Points
2	16233	22291.92	0	20449.56
3	21492	23071.07	3	21282.29
10	4047	24390.06	0	22723.97
11	19586	25845.92	3	24067.32
1	2498	32607.57	0	30939.44

The lowest true fit value found is 7633, and the highest is 30,939. There are 67 parameter samples that terminate within 0.5% of 7633. Over these 67 samples, the average number

of simulations is 3559. The parameter samples listed in Table 8.12 are the most efficient samples that terminate within 0.5% of the smallest value for the true fit. The parameter samples that terminate at the lowest values for the true fit are not the same parameter samples that terminate at the lowest values for the perturbed fit. The six parameters with the highest values for the perturbed fit are also the six parameters with the highest values for the true fit.

8.2.3 Test Problem 2*c*

When the SCEM is applied to Test Problem 2*c*, the smallest objective function value at termination is 18462 and the largest is 272,390 . The number of simulations varies between 1600 and 25005. There are 64 samples that terminate within 0.5% of the smallest objective function value. Of these 64 samples, the average number of simulations is 3076. Table 8.14 lists the samples that terminate within 0.5% of 18462 in fewer than 2100 simulations. Parameter samples that use a greater number of simulations than other samples do not always result in a lower objective function value.

Table 8.15 lists the parameter samples that terminate with objective function values over 25,000. Premature termination is a factor for only one of the poorly performing samples. Each parameter sample in Table 8.15 uses more simulations than any parameter sample in Table 8.14 and results in much larger fit values.

The true fit values vary from 7657 to 262,661. There are 62 points that terminate within 0.5% of the smallest true fit value. Of these 62 points, the average number of simulations is 3063. The five most efficient parameter samples that terminate within 0.5% of 7657 for the true fit are found in Table 8.14 along with one sample that does not yield a comparable true fit value. The parameter samples that terminate at the five largest values for the perturbed fit are the same parameter samples that terminate at the five largest values for the true fit.

Table 8.14: Test Problem 2*c*: Best SCEM Parameter Combinations

Index	Number of Simulations	Objective Function	Premature Termination Instances	Fit With True Data Points
18	1600	18462.85	0	7660.22
5	1816	19452.11	0	8324.13
26	1934	18462.84	0	7659.69
17	1984	18462.85	0	7660.41
34	2012	18462.94	0	7662.49
31	2028	18462.92	0	7658.79

Table 8.15: Test Problem 2*c*: Worst SCEM Parameter Combinations

Index	Number of Simulations	Objective Function	Premature Termination Instances	Fit With True Data Points
12	21221	25188.84	3	15394.97
10	3421	30566.57	0	21335.10
19	5403	52413.35	0	43777.60
2	12559	55394.46	0	46698.32
1	2571	272390.34	0	262661.10

8.2.4 Effects Due to Noise

Table 8.16 is a summary of the SCEM results when applied to Test Problems 2*a*, 2*b*, and 2*c*. The addition of noise to the data set has a noticeable impact on the outcome of the SCEM. For Test Problems 2*b* and 2*c*, the best value for the perturbed objective function is on the order of the magnitude of the noise in the data set plus the lowest true fit found in Test Problem 2*a*. In contrast to the results for the SCEM applied to Test Problem 1, the true fit values among the three cases vary by less than 0.88%. Unlike Test Problem 1, the best true fit for 2*b* is lower than the best true fit for 2*c*. For each of the three test problems, the best true fit is 27 orders of magnitude larger than the best true fit from Test Problem 1*a*.

For Test Problems 2*b* and 2*c*, there are parameter combinations where the perturbed objective function decreases and the true fit increases. Thus, for this test problem and the SCEM, there is not a direct correlation between the perturbed objective function and the true fit values. Also, for all three test problems, there are parameter samples that take more relative effort to return worsening results, and vice versa.

Another difference in the three problems is the number of simulations required for the best parameter samples. The average number of simulations refers to the parameter samples that terminate within 0.5% of the smallest average objective function found for each problem. The average number of simulations for the best parameters from each test problem vary from 3076 to 3283. Unlike Test Problem 1, the unperturbed data set does not require the highest average number of simulations. Test Problem 1*a* requires an average of 15189 simulations for its best parameter samples, and Test Problems 2*a* through 2*c* require less than one fourth as many simulations. When Test Problem 1*a* is compared with problems 2*a* through 2*c*, there are only 17 instances where one of the perturbed problems uses more simulations. All of these instances involve premature termination, which occurs when parameter values are grossly inefficient.

The addition of noise to Test Problem 2*a* has an impact on which parameters perform well. The parameter samples 31 and 34 perform well for all three problems. There are three

Table 8.16: Test Problem 2: Comparison of Noise Effects on the SCEM

	2*a*	2*b*	2*c*
Best Objective Function	7590.10	9334.90	18462.84
Best True Fit	7590.10	7633.32	7657.40
Average Number of Simulations for Best Parameter Combinations	3283	3559	3076
Sum Squared Difference Between True and Noisy Data	0	2535.10	9928.90
Good Parameter Samples	16, 31, 32, 33, 34, 35	25, 26, 31, 32, 34, 35	5, 17, 18, 26, 31, 34
Poor Parameter Samples	1, 2, 3, 10, 12	1, 2, 3, 10, 11	1, 2, 10, 12, 19

parameter samples that perform well for two of the three test problems, and several others that only do well in one case. There are three parameter samples that are in the group of poor performers for all three problems, and two that perform poorly for two of the three problems. The parameter samples that perform well for Test Problem 1*a* do not perform well for the perturbed test problems.

8.3 Test Problem 3

Again, Test Problem 3 is computationally expensive, so only a subset of the 81 parameter samples are tested for this problem.

8.3.1 Parameter Selection

When selecting parameters to test the SCEM's performance on Test Problem 3, we consider parameters that yielded good results for Test Problems 1 and 2. Since Test Problems 2*b* and 2*c* share many of the noise sources in Test Problem 3, more emphasis is placed on parameter values that efficiently result in low true fit values for these problems. Parameter sample 31 is effective and efficient for problems 2*a*, 2*b*, and 2*c*. Sample 34 is effective for these three problems but less efficient. Parameter samples 25, 18, and 32 all efficiently result in low values for the true fits for either 2*b* or 2*c*. Parameter samples 9, 16, and 36 return low values for the true fit on problem 1*b* or 1*c*, while samples 52 and 61 return low perturbed objective function values for one of these test problems. Sample 72 is tested because it returned good results for Test Problem 1. Parameter sample 10 is included because it is a poor performer on all six test problems. Finally, since sample 28 was a good performer for 1*b* and 1*c* and an

poor performer for $1a$, it is also included.

8.3.2 Numerical Results

Table 8.17 is a summary of the results when the SCEM is applied to Test Problem 3. The smallest objective function value found for the 500 starting points is 17.48150. Only three of the thirteen parameter samples terminate at this objective function value for at least one of the 500 starting points. Six of the parameter samples terminate at a lowest objective function value of 17.48151. The average objective function values for the thirteen parameter samples range from 17.4816 to 19.9751. The total number of simulations varies from 150730 to 302501.

The four parameter samples that result in the lowest average objective function values are 36, 52, 61, and 72. For these parameter samples, the average objective function is 17.483 or less. These average objective function values are very close to the smallest objective function value found, and for each of the four parameter samples, the difference between the highest and lowest objective functions at termination is less than 0.17. Thus, the results for these four parameter samples are relatively independent of the starting point. The number of simulations used for these samples ranges from 216143 to 302501. The difference in computational effort is striking for the similar outcomes.

Table 8.17: Test Problem 3: SCEM Results

Index	Average Objective Function	Number of Simulations	Lowest Objective Function	Premature Termination Instances
9	19.0374	150730	17.48151	0
10	19.9751	178265	17.48159	0
16	18.2410	178274	17.48150	0
18	18.4298	190183	17.48150	0
25	18.0465	215308	17.48150	0
28	17.8530	292424	17.48153	0
31	17.5642	237265	17.48152	0
32	17.5277	245599	17.48151	4
34	17.5243	207392	17.48152	0
36	17.4827	216143	17.48151	0
52	17.4816	239353	17.48151	0
61	17.4817	260020	17.48151	0
72	17.4816	302501	17.48151	0

The four parameters that have the lowest average objective function values also perform well for Test Problems 1a, 1b, and/or 1c. Although Test Problems 2b, 2c, and 3 all share both model structure error and perturbations in the data set, the best parameter samples from Test Problems 2b and 2c are not among the best parameter samples for Test Problem 3. Parameter sample 28, which performs well for Test Problems 1b and 1c and poorly for 1a, is not a top performer for Test Problem 3. Though parameters 9 and 16 result in relatively low true fit values for 1b and 1c, their average objective function values at termination on Test Problem 3 are among the highest values. Parameter sample 10, a poor performer for all six test problems, has the highest average objective function for Test Problem 3.

8.4 Selecting SCEM Parameters for Noisy Optimization

Table 8.18 is a list of the parameter values that perform well for several test problems. There is wide variation in the parameter values that perform well for the different test problems, and little overlap in the parameter values that do well for each individual problem. Table 8.18 does not demonstrate an appreciable bias towards any particular value for any of the four components of the parameter sample vector. For each of the four parameters, every possible tested value from Table 5.6 is found in Table 8.18 at least once. Thus, there is not a trend to recommend any particular value for any of the optimization parameters. The numerical results for the SCEM from the seven different test problems indicate that good SCEM parameters vary widely from problem to problem. This result is consistent with many metaheuristics.

Table 8.19 is a list of the SCEM parameter values that perform poorly for the seven test problems. There is much more overlap in the poorly performing parameter values than the good ones. For all of the poorly performing parameter samples, the number of complexes p is equal to 1. This is the lowest tested value for the parameter p. Obviously, a large number of complexes will result in a larger number of tested points, thereby increasing the possibility of a low objective function. Thus, the correlation between a low value for p and a high objective function is not surprising. The values for m and β_S take on every possible tested value from Table 5.6. Thus, the data does not suggest a direct correlation between any particular value for m or β_S and a poor outcome. However, $q = 2$ for every parameter sample in Table 8.19. The parameter q represents the number of parents used in the generation of each offspring. The lowest tested value for q is 2. Therefore, the data suggests that a larger number of parents per reproduction step might return better results. However, it is worth noting that Table 8.18 also contains parameter samples where $p = 1$ or $q = 2$. Lower values for these optimization parameters do not automatically doom the algorithm to poor performance.

Table 8.18: Competitive SCEM Parameter Values

Index	p	m	q	β_S
9	1	6	4	3
16	1	7	4	1
18	1	7	4	3
25	1	8	4	1
28	2	6	2	1
31	2	6	3	1
32	2	6	3	2
34	2	6	4	1
36	2	6	4	3
52	2	8	4	1
61	3	6	4	1
72	3	7	4	3

Table 8.19: Poor SCEM Parameter Values

Index	p	m	q	β_S
1	1	6	2	1
2	1	6	2	2
3	1	6	2	3
10	1	7	2	1
12	1	7	2	3
19	1	8	2	1

Chapter 9

Comparison of Algorithms

We are interested in the relative performance of the three optimization algorithms. Our list of algorithms is certainly not comprehensive, and there are many other algorithms that optimizers might select over the NMA, NMA/SA, and SCEM. However, a comparison between the three algorithms may be useful when practitioners are selecting an algorithm.

9.1 Test Problem 1

The NMA and SCEM are not equally efficient or effective for Test Problems 1a, 1b, and 1c. Table 9.1 is a comparison of optimization results for the two algorithms applied to 1a, 1b, and 1c. For each of the three problems, the SCEM is able to find a lower average objective function value than the NMA. For 1a, the lowest average objective function value for the SCEM is 18 orders of magnitude smaller than the smallest value for the NMA. The difference is not so large for the two perturbed problems. The SCEM requires more simulations than the NMA in order to obtain the smallest average objective function. For 1a, the SCEM requires six times more simulations than the NMA for the best result. For 1b and 1c, the SCEM requires, respectively, more than 5 and 2 times the number of simulations that the NMA needs. The average number of simulations over the top parameter samples shows that the SCEM uses many more simulations than the NMA for 1a. This trend also holds for Problems 1b and 1c. Although the SCEM finds lower average objective functions than the NMA for all three test problems, the NMA finds lower true fit values for 1b and 1c.

Table 9.1: Test Problem 1: Comparison of the NMA and SCEM

Algorithm	Lowest Average Objective Function	Number of Simulations For Lowest Objective	Average Number of Simulations for Group of Best Results	Lowest Average True Fit
Test Problem 1a				
NMA	0.40	2798	2706	
SCEM	5.34×10^{-19}	16555	15189	
Test Problem 1b				
NMA	2481.77	1139	1684	53.78
SCEM	2470.04	6648	4882	55.51
Test Problem 1c				
NMA	9890.95	2162	1520	6.45
SCEM	9752.65	6300	6301	15.19

9.2 Test Problem 2

For Test Problems 2a, 2b, and 2c, the lowest average objective function value is marginally lower when the SCEM is used as opposed to the NMA. Also, for 2b and 2c, the true fit values are slightly lower when the SCEM algorithm is used. The slight increase in accuracy comes at the cost of more simulations. On average, the SCEM uses at least twice as many simulations as the NMA for all three test problems. The number of simulations the SCEM uses to return the lowest average objective function is more than twice as many as the NMA for 1a and 1c, and almost twice as many for 1b.

Table 9.2: Test Problem 2: Comparison of the NMA and SCEM

Algorithm	Lowest Average Objective Function	Number of Simulations For Lowest Objective	Average Number of Simulations for Group of Best Results	Lowest Average True Fit
Test Problem 2*a*				
NMA	7590.12	965	1532	
SCEM	7590.10	1977	3283	
Test Problem 2*b*				
NMA	9334.92	1044	1474	7634.15
SCEM	9334.90	1998	3559	7633.32
Test Problem 2*c*				
NMA	18462.88	955	1376	7658.55
SCEM	18462.84	1934	3076	7657.40

9.3 Test Problem 3

Table 9.3 lists the results for all three algorithms applied to Test Problem 3. The trends for the numerical results for this problem are very clear. The NMA has the highest value for the lowest average objective function. The number of simulations required for the NMA is the smallest number. For all three algorithms (and two cooling schedules), increases in the number of simulations needed to return the lowest average objective function result in smaller objective function values. The average number of simulations follows the same trend. The SCEM requires twice as many simulations as the NMA, and the two cooling schedules for the NMA/SA require a value in between.

Table 9.3: Test Problem 3: Comparison of the NMA, NMA/SA Hybrid, and SCEM

Algorithm	Lowest Average Objective Function	Number of Simulations for Lowest Average Objective
NMA	18.92	106249
NMA/SA Cooling Schedule 1	18.58	127168
NMA/SA Cooling Schedule 2	17.73	160417
SCEM	17.48	216143

9.4 Selecting an Algorithm

When selecting one of these three algorithms for a noisy parameter identification problem, several factors must be considered. Our numerical analysis indicates that the SCEM is likely to return lower objective function values than the other methods. The drawback, however, is an increase by a factor of two or more in the amount of effort required. If fit evaluations are extremely expensive, the SCEM may not be an option. For Test Problem 1*a* (the only non–noisy test problem), the SCEM finds objective function values that are eighteen orders of magnitude lower than the NMA's best results. This is certainly an eye–catching difference. However, for the noisy Test Problems 1*b*, 1*c*, and 2*a* through 2*c*, the SCEM results are such mediocre improvements over the NMA results that the increase in cost is an important consideration. For Test Problem 3, a problem with various and unquantified sources of noise, the SCEM is much more effective than the NMA or the NMA/SA. Therefore, we cannot conclude that the presence of noise has an equalizing effect on the fit values found by the NMA, NMA/SA, and SCEM.

The SCEM has two other advantages over the NMA and NMA/SA hybrid. The numerical results for our test problems demonstrate that the SCEM is less sensitive to the algorithmic parameter values than the other two algorithms. Although the SCEM does return varying results when the algorithmic parameters are varied, the pool of parameter values that result in termination at or near the lowest fit values is much larger than the corresponding pools for the NMA and NMA/SA. If optimizers do not wish to calibrate a problem, the likelihood of choosing a good value for the SCEM parameters is probably larger than the likelihood of choosing good parameter values for the other two algorithms. The second advantage that the SCEM enjoys over the other two algorithms is that the SCEM results do not depend as heavily on the starting points are the NMA or NMA/SA hybrid. This trend is especially

visible for Test Problem 3. There are four parameter samples that terminate at average fit values that are only marginally larger than the smallest fit value. Thus, the SCEM terminates very close to the lowest fit value for all 500 starting points. If an optimizer does not wish to employ perturbation of the starting points, the results from a properly calibrated application of the SCEM to a single starting point are more likely to represent a minima than the NMA or NMA/SA results.

The NMA/SA hybrid is able to outperform the NMA. The NMA/SA terminates at lower values than the NMA for several of the NMA/SA parameter values we test on Test Problem 3. For Cooling Schedule 1 and one value of the cooling schedule parameters, the NMA/SA uses fewer simulations than the NMA and terminates at a lower fit value! For Cooling Schedule 2, the NMA/SA is always more expensive than the NMA, but only 5 parameter samples out of nine terminate at lower fit values. So, the NMA/SA can be cheaper and more effective than the NMA, and it can also be more expensive and less effective. If an optimizer does not wish to calibrate the NMA/SA, the NMA might be a less risky choice, since poor choices for the NMA/SA parameter values can be costly and ineffective.

Chapter 10

Conclusions and Future Work

Our theoretical investigation of the NMA/SA hybrid is an extension of earlier results for the NMA. The new theory requires specific assumptions regarding the cooling schedule's rate of cooling. Although our assumption regarding the cooling schedule is not easily verified in the general case, there are two classes of cooling schedules that are proven to honor the assumption. Therefore, in practice, the NMA/SA combined with cooling schedules of these two types will converge under the same conditions that are sufficient for the NMA to converge.

Our numerical study provides partial answers to the questions posed in Chapter 5. Noise in the data set and model structure error affect the test problems by increasing the minimum objective function value. In most cases, noisy versions of smooth optimization problems require fewer simulations than the smooth problem itself. For all of our test problems, the lowest true fit found for a perturbed problem is markedly larger than the lowest fit found for the unperturbed problem.

For all three algorithms and all seven test problems, the values of algorithmic parameters have a profound effect upon optimization outcome. Poor choices of algorithmic parameters can result in fit values many orders of magnitude higher than if a good set of parameter values are used. These trends hold for our perturbed and smooth test problems alike. Also, some parameter values are inefficient as well as ineffective. For each of the three algorithms and every test problem, there are combinations of parameter samples where one parameter sample requires fewer simulations than another parameter sample and yet results in a lower fit value. Thus, a parameter sample that requires a large number of simulations does not necessarily result in a low fit value. The ineffective and inefficient parameter values need to be avoided, though they may vary from problem to problem.

The performance of the NMA varies enough from problem to problem that careful choice of the algorithmic parameters will probably save time and produce better results. If calibration is not desirable, practitioners can consider deviating from the standard parameter values by

increasing γ from the generally accepted value of 2 to 3.2 to 4.3. We know from previous studies that the NMA/SA hybrid must be carefully calibrated for perturbed and smooth problems alike. Our results illustrate that the best SCEM parameters also vary widely from problem to problem. However, if efficiency is not a concern, the SCEM parameter choice is less critical, as many parameter samples are effective yet inefficient.

Our studies have raised other interesting questions related to noise–corrupted parameter identification problems. We are curious about the performance of gradient–based methods on our seven test problems. We plan to implement NL2SOL, a modified Levenberg–Marquardt method developed by Dennis, Gay, and Welsch [42, 43]. NL2SOL was designed specifically for highly nonlinear large residual least–squares optimization problems. In addition to calibrating NL2SOL to our test problems, we will also study the effects of combining NL2SOL with the NMA, NMA/SA hybrid, or SCEM. In particular, we wonder if we can combine the local convergence properties of gradient–based algorithms with the other algorithms' relative independence to starting points.

Optimization in the presence of noise poses another challenging question. As our results indicate, a decrease in the perturbed objective function does not always result in a decrease in the true fit. The question, then, is when does extra effort in the optimization algorithm become statistically meaningless? We plan to investigate a method to bound the error in the perturbed objective function. Then, we hope to develop tools for determining useful and practical termination criteria that avoid wasted effort that does not improve the true fit. Finally, the convergence properties of the SCEM have yet to be proven.

Choosing and calibrating optimization algorithms for effective and efficient solution to parameter identification problems is important. However, there are many other important issues in parameter identification problems that merit investigation. Model parameters can be correlated with one another. The issue of correlations among parameters causes several interesting problems. If parameters are correlated, then small or even large changes in two or more parameter values could result in the same objective function value. This phenomena creates the possibility of multiple, or even infinite global minima. The goal of parameter identification problems is to identify parameter values to use in solution of the forward problem. Analysts need to be able to select parameter values to use in their forward problems, even when many possibilities exist. A common approach is to perform a perturbation analysis. As in Test Problem 3, the starting point for optimization algorithms is perturbed many times, and the resulting termination points may indicate different areas of local and/or global minima. The forward problem can then be solved with many different parameter combinations. Statistical sampling of likely values for parameters will result in different solutions to the forward problem, resulting in a probability distribution for the results of the forward problem. The differences can be compared. However, this method lacks theoretical rigor, and we hope to investigate alternative methods with strong theoretical support.

We intend to implement another kind of perturbation analysis. We will perturb the collected data set with a random normally distributed noise term of relatively small magnitude. We

will sample the model parameter values resulting from each new perturbation of the data set and generate a new probability distribution for the solution of the forward problem. We will compare the probability distributions created by both methods and analyze existing theory to determine which method generates the most applicable probability distribution.

Another area of interest for our future work is determination of appropriate and defensible techniques to bound the error in our parameter estimations. In particular, we hope to develop a bounding technique that is consistent with perturbation analysis.

Bibliography

[1] D. Abramson, M. Krishnamoorthy, and H. Dang. Simulated annealing cooling schedules for the school timetabling problem. *Asia–Pacific J. Oper. Res.*, 16(1):1–22, 1999.

[2] J. W. Akitt. Function minimisation using the Nelder and Mead simplex method with limited arithmetic precision: the self regenerative simplex. *Comput. J.*, 20(1):84–85, 1977.

[3] M. M. Ali, A. Törn, and S. Viitanen. A direct search variant of the simulated annealing algorithm for optimization involving continuous variables. *Comput. Oper. Res.*, 29:87–102, 2002.

[4] E. Anderson and M. Ferris. A direct search algorithm for optimization with noisy function evaluations. *SIAM J. Optim.*, 11(3):837–857, 2001.

[5] D. V. Arnold and H.-G. Beyer. Performance analysis of evolution strategies with multi–recombination in high–dimensional $\mathbb{R}^N$–search spaces disturbed by noise. *Theor. Comput. Sci.*, 289:629–647, 2002.

[6] D. V. Arnold and H.-G. Beyer. A comparison of evolution strategies with other direct search methods in the presence of noise. *Comput. Optim. Appl.*, 24:135–159, 2003.

[7] D. V. Arnold and H.-G. Beyer. Performance analysis of evolutionary optimization with cumulative step length adaptation. *IEEE Trans. Automat. Control*, 49(4):617–622, 2004.

[8] J. Avis. GTFM–User documentation: Functional description, theoretical development, and software architecture. Sandia National Laboratories, 1996. (Copy on file in the Sandia WIPP Records Center, Carlsbad, NM as ERMS # 240244).

[9] J. R. Barker. A generalized radial flow model for hydraulic tests in fractured rock. *Water Resour. Res.*, 24(10):1796–1804, 1988.

[10] D. B. Bates and D. G. Watts. *Nonlinear Regression Analysis and Its Applications.* Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons, Inc., New York, 1988.

[11] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons, Inc., New York, 1993.

[12] J. Bear. *Hydraulics of Groundwater*. McGraw–Hill Series in Water Resources and Environmental Engineering. McGraw–Hill, United States, 1979.

[13] H.-G. Beyer. Evolutionary algorithms in noisy environments: Theoretical issues and guidelines for practice. *Comput. Meth. Appl. Mech. Eng.*, 186:239–267, 2000.

[14] H.-G. Beyer, M. Olhofer, and B. Sendhoff. On the impact of systematic noise on the evolutionary optimization performance–a sphere model analysis. *Genet. Program. Evol. Mach.*, 5:327–360, 2004.

[15] J. Borggaard. *The Sensitivity Equation Method for Optimal Design*. PhD thesis, Virginia Tech, Blacksburg, VA, December 1994.

[16] J. Borggaard and J. Burns. A sensitivity equation approach to shape optimization in fluid flows. In M. Gunzburger, editor, *Flow Control*, volume 68 of *Proceedings of the IMA*. Springer-Verlag, 1995.

[17] J. Borggaard and J. Burns. Asymptotically consistent gradients in optimal design. In N. Alexandrov and M. Hussaini, editors, *Multidisciplinary Design Optimization: State of the Art*, pages 303–314, Philadelphia, PA, 1997. SIAM.

[18] J. Borggaard and J. Burns. A PDE sensitivity equation method for optimal aerodynamic design. *J. Comput. Phys.*, 136(2):366–384, 1997.

[19] J. Borggaard, J. Burns, E. Cliff, and M. Gunzburger. Sensitivity calculations for a 2D, inviscid, supersonic forebody problem. In H. T. Banks, R. Fabiano, and K. Ito, editors, *Identification and Control of Systems Governed by Partial Differential Equations*, pages 14–24, Philadelphia, PA, 1993. SIAM.

[20] J. Borggaard and D. Pelletier. Computing design sensitivities using an adaptive finite element method. In *Proceedings of the 27th AIAA Fluid Dynamics Conference*, 1996.

[21] J. Borggaard and D. Pelletier. Optimal shape design in forced convection using adaptive finite elements. In *Proc. 36th AIAA Aerospace Sciences Meeting and Exhibit*, 1998. AIAA Paper 98-0908.

[22] J. Borggaard, D. Pelletier, and K. Vugrin. On sensitivity analysis for problems with numerical noise. In *Proc. 9th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 2002. AIAA Paper 2002-5553.

[23] D. M. Bortz and C. T. Kelley. The simplex gradient and noisy optimization problems. In J. Borggaard, J. Burns, E. Cliff, and S. Schreck, editors, *Computational Methods for Optimal Design and Control*, pages 77–90. Birkhäuser, 1998.

[24] C. G. Broyden. The convergence of a class of double rank minimization algorithms 2. The new algorithm. *J. Inst. Math. Appl.*, 6:221–231, 1970.

[25] J. Burkardt, M. Gunzburger, and J. Peterson. Discretization of cost sensitivities in shape optimization. In *Computation and Control, IV*, volume 20 of *Progr. Systems Control Theory*. Birkhäuser, 1995.

[26] J. Burkardt, M. Gunzburger, and J. Peterson. Insensitive functionals, inconsistent gradients, spurious minima, and regularized functionals in flow optimization problems. *Int. J. Comput. Fluid Dyn.*, 16(3):171–185, 2002.

[27] J. Burkardt and J. Peterson. Control of steady incompressible 2D channel flow. In M. Gunzburger, editor, *Flow Control*, volume 68 of *Proceedings of the IMA*. Springer-Verlag, 1995.

[28] J. Burman and B. Gebart. Influence from numerical noise in the objective function for flow design optimization. *Internat. J. Numer. Methods Heat Fluid Flow*, 11(1):6–19, 2001.

[29] D. Byatt. Convergent variants of the Nelder–Mead algorithm. Master's thesis, University of Canterbury, Christchurch, New Zealand, 2000.

[30] M. F. Cardoso, R. L. Salcedo, and S. F. de Azevedo. The simplex–simulated annealing approach to continuous non–linear optimization. *Computers Chem. Engng.*, 20(9):1065–1080, 1996.

[31] M. F. Cardoso, R. L. Salcedo, S. F. de Azevedo, and D. Barbosa. A simulated annealing approach to the solution of minlp problems. *Computers Chem. Engng.*, 21(12):1349–1364, 1997.

[32] R. G. Carter. On the global convergence of trust-region algorithms using inexact gradient information. *SIAM J. Numer. Anal.*, 28(1):251–265, 1991.

[33] R. G. Carter. Numerical experience with a class of algorithms for nonlinear optimization using inexact function and gradient information. *SIAM J. Sci. Comput.*, 14(2):368–388, 1993.

[34] H. Cohn and M. Fielding. Simulated annealing: Searching for an optimal temperature schedule. *SIAM J. Optim.*, 9(3):779–802, 1999.

[35] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. Global convergence of a class of trust region algorithms for optimization with simple bounds. *SIAM J. Numer. Anal.*, 25(2):433–460, 1988.

[36] V. A. Cooper, V. T. V. Nguyen, and J. A. Nicell. Evaluation of global optimization methods for conceptual rainfall–runoff model calibration. *Wat. Sci. Tech.*, 36(5):53–60, 1997.

[37] A. Corana, M. Marchesi, C. Martini, and S. Ridella. Minimizing multimodal functions of continuous variables with the "simulated annealing" algorithm. *ACM Trans. Math. Software*, 13(3):262–280, 1987.

[38] J. C. Davis. *Statistics and Data Analysis in Geology.* John Wiley & Sons, United States, 2nd edition, 1973.

[39] A. Dean and D. Voss. *Design and Analysis of Experiments.* Springer Texts in Statistics. Springer, New York, 1999.

[40] J. W. Demmel. *Applied Numerical Linear Algebra.* SIAM, Philadelphia, 1997.

[41] J. E. Dennis Jr., M. El-Alem, and K. Williamson. A trust–region approach to nonlinear systems of equalities and inequalities. *SIAM J. Optim.*, 9(2):291–315, 1999.

[42] J. E. Dennis Jr., D. M. Gay, and R. E. Welsch. Algorithm 573 NL2SOL– An adaptive nonlinear least–squares algorithm. *ACM Trans. Math. Software*, 7(3):369–383, 1981.

[43] J. E. Dennis Jr., D. M. Gay, and R. E. Welsch. An adaptive nonlinear least-squares algorithm. *ACM Trans. Math. Software*, 7(3):348–368, 1981.

[44] J. E. Dennis Jr. and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Number 16 in Classics in Applied Mathematics. SIAM, Philadelphia, 1996.

[45] J. E. Dennis Jr. and T. Steihaug. On the successive projections approach to least–squares problems. *SIAM J. Numer. Anal.*, 23(4):717–733, 1986.

[46] J. E. Dennis Jr. and V. Torczon. Direct search methods on parallel machines. *SIAM J. Optim.*, 1(7):448–474, 1991.

[47] Q. Y. Duan, V. K. Gupta, and S. Sorooshian. Shuffled complex evolution approach for effective and efficient global minimization. *J. Optim. Theory Appl.*, 76(3):501–521, 1993.

[48] Q. Y. Duan, S. Sorooshian, and V. K. Gupta. Effective and efficient global optimization for conceptual rainfall–runoff models. *Water Resour. Res.*, 28(4):1015–1031, 1992.

[49] Q. Y. Duan, S. Sorooshian, and V. K. Gupta. Optimal use of the SCE–UA global optimization method for calibrating watershed models. *J. Hydrol.*, 158:265–284, 1994.

[50] C. Elster and A. Neumaier. A grid algorithm for bound constrained optimizaiton of noisy functions. *IMA J. Numer. Anal.*, 15:585–608, 1995.

[51] C. Elster and A. Neumaier. A method of trust region type for minimizing noisy functions. *Computing*, 58:31–46, 1997.

[52] H. T. Fang and H. F. Chen. Almost surely convergent global optimization algorithm using noise–corrupted observations. *J. Optim. Theory Appl.*, 104(2):343–376, 2000.

[53] U. Felgenhauer. Algorithmic stability analysis for certain trust region methods. In A. Fiacco, editor, *Mathematical Programming with Data Perturbations*, volume 195 of *Lecture Notes in Pure and Applied Math.* M. Dekker, 1998.

[54] J. M. Fitzpatrick and J. J. Grefenstette. Genetic algorithms in a noisy environment. *Mach. Learn.*, 3(2):101–120, 1988.

[55] R. Fletcher. A new approach to variable metric algorithms. *Comput. J.*, 13:317–322, 1970.

[56] R. Fletcher and M. Powell. A rapidly convergent descent method for minimization. *Comput. J.*, 6:163–168, 1963.

[57] P. D. Frank and G. R. Shubin. A comparison of optimization-based approaches for a model computational aerodynamics design problem. *J. Comput. Phys.*, 98:74–89, 1992.

[58] F. Franzé and N. Speciale. A tabu–search–based algorithm for continuous multiminima problems. *Internat. J. Numer. Methods Engrg.*, 50:665–680, 2001.

[59] R. A. Freeze and J. A. Cherry. *Groundwater.* Prentice–Hall, Englewood Cliffs, 1979.

[60] U. M. García-Palomares and J. F. Rodríguez. New sequential and parallel derivative–free algorithms for unconstrained minimization. *SIAM J. Optim.*, 13(1):79–96, 2002.

[61] S. B. Gelfand and S. K. Mitter. Simulated annealing with noisy or imprecise energy measurements. *J. Optim. Theory Appl.*, 62(1):49–62, 1989.

[62] P. E. Gill, W. M. Murray, M. A. Saunders, and M. H. Wright. Computing forward-difference intervals for numerical optimization. *SIAM J. Sci. Comput.*, 4(2):310–321, June 1983.

[63] P. E. Gill, W. M. Murray, and M. H. Wright. *Practical Optimization.* Academic Press, London, 1981.

[64] M. S. Gockenbach and W. W. Symes. Adaptive simulation, the adjoint state method, and optimization. In *Large–Scale PDE–Constrained Optimization*, Lecture Notes in Computational Science and Engineering, pages 281–297. Springer, 2003.

[65] D. Goldfarb. A family of variable metric methods derived by variational means. *Math. Comp.*, 24:23–26, 1970.

[66] L. I. González-Monroy and Antonio Córdoba. Optimization of energy supply systems with simulated annealing: Continuous and discrete descriptions. *Phys. A*, 284:433–447, 2000.

[67] W. J. Gutjahr and G. CH. Pflug. Simulated annealing for noisy cost functions. *J. Global Opt.*, 8:1–13, 1996.

[68] E. Haber, U. M. Ascher, and D. Oldenburg. On optimization techniques for solving nonlinear inverse problems. *Inverse Problems*, 16:1263–1280, 2000.

[69] B. Hajek. Cooling schedules for optimal annealing. *Math. Oper. Res.*, 13:311–329, 1988.

[70] A. E. Hassan. Validation of numerical ground water models used to guide decision making. *Ground Water*, 42(2):277–290, Mar–Apr 2004.

[71] A.-R. Hedar and M. Fukushima. Heuristic pattern search and its hybridization with simulated annealing for nonlinear global optimization. *Optim. Methods Softw.*, 19(3–4):291–308, 2004.

[72] M. Heinkenschloss. A trust–region method for norm constrained problems. *SIAM J. Numer. Anal.*, 35(4):1594–1620, 1998.

[73] J. C. Helton and F. J. Davis. Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems. Technical Report SAND2001-0417, Sandia National Laboratories, Albuquerque, NM, 2002.

[74] R. Hooke and T. A. Jeeves. Direct search solution of numerical and statistical problems. *J. ACM*, 8:212–229, 1961.

[75] G. J.-W. Hou, A. C. Taylor, and V. M. Korivi. Discrete shape sensitivity equations for aerodynamic problems. In *Proc. 27th AIAA/SAE/ASME/ASEE Joint Propulsion Conference*, 1991. AIAA Paper 91-2259.

[76] INTERA Engineering. paCalc. http://www.interaeng.ca/tech_pacalc.htm. (Accessed November 22, 2004).

[77] H. Joe and J. C. Nash. Numerical optimization and surface estimation with imprecise function evaluations. *Stat. Comput.*, 13:277–286, 2003.

[78] C. T. Kelley. Detection and remediation of stagnation in the Nelder–Mead algorithm using a sufficient decrease condition. *SIAM J. Optim.*, 10(1):43–55, 1999.

[79] D. Kincaid and W. Cheney. *Numerical Analysis*. Brooks/Cole Publishing Company, 2nd edition, 1996.

[80] T. G. Kolda, R. M. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Rev.*, 45(3):385–482, 2003.

[81] Y-D. Kwon, S-B. Kwon, S-B. Jin, and J-Y. Kim. Convergence enhanced genetic algorithm with successive zooming method for solving continuous optimization problems. *Comput. Struct.*, 81:1715–1725, 2003.

[82] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright. Convergence properties of the Nelder–Mead simplex method in low dimensions. *SIAM J. Optim.*, 9(1):112–147, 1998.

[83] C. L. Lawson and R. J. Hanson. *Solving Least Squares Problems.* Prentice–Hall Series in Automatic Computation. Prentice–Hall, Englewood Cliffs, New Jersey, 1974.

[84] K. Levenberg. A method for the solution of certain problems in least squares. *Quart. Appl. Math.*, 2:164–168, 1944.

[85] S. Lucidi and M. Sciandrone. A derivative–free algoritm for bound constrained optimization. *Comput. Optim. Appl.*, 21:119–142, 2002.

[86] S. Lucidi and M. Sciandrone. On the global convergence of derivative-free methods for unconstrained optimization. *SIAM J. Optim.*, 13(1):97–116, 2002.

[87] B. T. Luke. Simulated annealing. http://fconyx.ncifcrf.gov/~lukeb/simann1.html and http://fconyx.ncifcrf.gov/~lukeb/simanf1.html. (Accessed September 14, 2004).

[88] D. W. Marquardt. An algorithm for least squares estimation of nonlinear parameters. *SIAM J. Ind. & App. Math.*, 11:431–441, 1963.

[89] K. I. M. McKinnon. Convergence of the Nelder–Mead simplex method to a nonstationary point. *SIAM J. Optim.*, 9(1):148–158, 1998.

[90] J. J. Moré, B. S. Garbow, and K. E. Hillstrom. Testing unconstrained optimization software. *ACM Trans. Math. Software*, 7(1):17–41, 1981.

[91] Math Works. MATLAB. The Math Works, Natick, MA, 2004.

[92] National Research Council: Committee on Ground Water Modeling Assessment. *Ground Water Models: Scientific and Regulatory Applications.* National Academy of Science, United States, 1990.

[93] J. A. Nelder and R. Mead. A simplex method for function minimization. *Comput. J.*, 7:501–511, 1965.

[94] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer Series in Operations Research. Springer–Verlag, New York, 2000.

[95] Y. Nourani and B. Andresen. A comparison of simulated annealing cooling strategies. *J. Phys. A: Math. Gen.*, 31:8373–8385, 1998.

[96] G. C. Onwubolu and B. V. Babu. *New Optimization Techniques in Engineering*, volume 141 of *Studies in Fuzziness and Soft Computing.* Springer–Verlag, Berlin, 2004.

[97] I. H. Osman and G. Laporte. Metaheuristics: A bibliography. *Ann. Oper. Res.*, 63:513–623, 1996.

[98] D. Pelletier, J. Borggaard, and J.-F. Hétu. A continuous sensitivity equation method for conduction and phase change problems. In *38th AIAA Aerospace Sciences Meeting and Exhibit*, 2000. AIAA Paper 00-0881.

[99] D. T. Pham and D. Karabga. *Intelligent Optimization Techniques: Genetic Algorithms, Tabu Search, Simulated Annealing, and Neural Networks*. Springer-Verlag, London, 2000.

[100] J. F. Pickens, G. E. Grisak, J. D. Avis, and D. W. Belanger. Analysis and interpretation of borehole hydraulic tests in deep boreholes: Principles, model development, and applications. *Water Resour. Res.*, 23(7):1341–1375, 1987.

[101] M. J. D. Powell. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *Comput. J.*, 7:155–162, 1964.

[102] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vettering. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, UK, 1992.

[103] C. J. Price, I. D. Coope, and D. Byatt. A convergent variant of the Nelder–Mead algorithm. *J. Optim. Theory Appl.*, 113(1):5–19, 2002.

[104] W. L. Price. A controlled random search procedure for global optimisation. In L. C. W. Dixon and G. P. Szegö, editors, *Towards Global Optimisation 2*, pages 71–84. North–Holland Publishing Company, 1978.

[105] C. R. Reeves, editor. *Modern Heuristic Techniques for Combinatorial Problems*. John Wiley & Sons, Inc., New York, 1993.

[106] R. Roberts. Moderately fractured rock experiment: Well test analysis using nSIGHTS. Technical Report 06819–REP-01300–10062–R00, Sandia National Laboratories, 2002. Supporting Technical Report for Ontario Power Generation, Nuclear Waste Management Division.

[107] H. H. Rosenbrock. An automatic method for finding the greatest or least value of a function. *Comput. J.*, 3:175–184, 1960.

[108] E. W. Sachs. Convergence of algorithms for perturbed optimization problems. *Ann. Oper. Res.*, 27:311–342, 1990.

[109] H. Sanvicente-Sánchez and J. Frausto-Solís. A method to establish the cooling scheme in simulated annealing like algorithms. In *International Conference on Computational Science and Its Applications (ICSSA 2004); May 14-17; Assisi, Italy*, volume 3045, pages 755–763, Berlin, 2004. Springer–Verlag.

[110] G. J. Savage and H. K. Kesavan. The Graph–Theoretic Field Model–I. Modelling and formulations. *J. Franklin Inst., volume = 307, number = 2, pages = 107–147, year = 1979.*

[111] R. Schmidt. *Advances in Nonlinear Parameter Optimization.* Lecture Notes in Control and Information Sciences. Springer–Verlag, Berlin, 1982.

[112] M. Sen and P. L. Stoffa. *Global Optimization Methods in Geophysical Inversion.* Advances in Exploration Geophysics 4. Elsevier, Amsterdam, 1995.

[113] D. F. Shanno. Conditioning of Quasi–Newton methods for function minimizations. *Math. Comp.*, 24:641–656, 1970.

[114] P. Siarry and G. Berthiau. Fitting of tabu search to optimize functions of continuous variables. *Internat. J. Numer. Methods Engrg.*, 40:2449–2457, 1997.

[115] J. Stander and B. W. Silverman. Temperature schedules for simulated annealing. *Stat. Comput.*, 4:21–32, 1994.

[116] L. Stanley. Shape sensitivities for optimal design: A case study on the use of continuous sensitivity equation methods. In D. Gao et al., editor, *Nonsmooth/Nonconvex Mechanics.* Kluwer Academic Publishers, 2001.

[117] R. Storn and K. Price. Differential evolution–A simple and efficient heuristic for global optimization over continuous spaces. *J. Global Opt.*, 11:341–359, 1997.

[118] N-Z. Sun. *Inverse Problems in Groundwater Modeling*, volume 6 of *Theory and Applications of Transport in Porous Media.* Kluwer Academic Publishers, Boston, 1999.

[119] A. Tarantola. *Inverse Problem Theory: Methods for Data Fitting and Model Parameter Estimation.* Elsevier, New York, 1988.

[120] A. C. Taylor III, G. W. Hou, and V. M. Korivi. A methodology for determining aerodynamic sensitivity derivatives with respect to variation of geometric shape. In *Proc. AIAA/ASME/ASCE/AHS/ASC 32nd Structures, Structural Dynamics, and Materials Conference*, Baltimore, MD, April 1991. AIAA paper 91-1101.

[121] A. C. Taylor III, G. W. Hou, and V. M. Korivi. Sensitivity analysis, approximate analysis and design optimization for internal and external viscous flows. In *Proc. AIAA Aircraft Design Systems and Operations Meeting*, 1991. AIAA paper 91-3083.

[122] V. Torczon. *Multi–Directional Search: A Direct Search Algorithm for Parallel Machines.* PhD thesis, Rice University, Houston, TX, 1989.

[123] V. Torczon. On the convergence of pattern search algorithms. *SIAM J. Optim.*, 7(1):1–25, 1997.

[124] M. W. Trosset. *On the Use of Direct Search Methods for Stochastic Optimization.* Technical Report 00–20, Department of Computational and Applied Mathematics, Rice University, Houston TX, 2000.

[125] É. Turgeon, D. Pelletier, and J. Borggaard. A continuous sensitivity equation approach to optimal design in mixed convection. In *Proc. 33rd AIAA Thermophysics Conference*, 1999. AIAA Paper 99-3625.

[126] É. Turgeon, D. Pelletier, and J. Borggaard. A continuous sensitivity equation method for flows with temperature dependent properties. In *Proc. 8th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 2000. AIAA Paper 2000-4821.

[127] É. Turgeon, D. Pelletier, and J. Borggaard. Application of a sensitivity equation method to the $\kappa - \epsilon$ model of turbulence. In *Proc. 15th AIAA Computational Fluid Dynamics Conference*, 2001. AIAA Paper 2001–3000.

[128] É. Turgeon, D. Pelletier, and J. Borggaard. Sensitivity and uncertainty analysis for variable property flows. In *Proc. 39th AIAA Aerospace Sciences Meeting and Exhibit*, 2001. AIAA Paper 2001–0139.

[129] É. Turgeon, D. Pelletier, S. Étienne, and J. Borggaard. Sensitivity and uncertainty analysis for turbulent flows. In *Proc. 40th AIAA Aerospace Sciences Meeting and Exhibit*, 2002. AIAA Paper 2002–0985.

[130] P. J. M. van Laarhoven and E. H. L. Aarts. *Simulated Annealing: Theory and Applications.* Mathematics and its applications. D. Reidel Publishing Company, Dordrecht, 1988.

[131] M. N. Vrahatis, G. S. Androulakis, and G. E. Manoussakis. A new unconstrained optimization method for imprecise function and gradient values. *J. Math. Anal. Appl.*, 197:586–607, 1996.

[132] J. A. Vrugt, H. V. Gupta, W. Bouten, and S. Sorooshian. A shuffled complex evolution metropolis algorithm for optimization and uncertainty assesment of hydrologic model parameters. *Water Resour. Res.*, 39(8):1201, 2003.

[133] K. E. Vugrin. On the effect of numerical noise in simulation-based optimization. Master's thesis, Virginia Tech, Blacksburg, VA, March 2003.

[134] F. H. Walters, Jr. L. R. Parker, S. L. Morgan, and S. N. Deming. *Sequential Simplex Optimization.* Chemometrics Series. CRC Press, Boca Raton, FL, 1991.

[135] D. J. Woods. *An Interactive Approach for Solving Multi–Objective Optimization Problems.* PhD thesis, Rice University, 1985.

[136] G. D. Wyss and K. H. Jorgensen. A user's guide to LHS: Sandia's latin hypercube sampling software. Technical Report SAND98-0210, Sandia National Laboratories, Albuquerque, NM, 1998.

[137] Z-J. Yang, T. Hachino, and T. Tsugi. On–line identification of continuous time–delay systems combining least–squares techniques with a genetic algorithm. *Int. J. Control*, 66(1):23–42, 1997.

[138] S. H. Zanakis, J. R. Evans, and A. A. Vazacopoulos. Heuristic methods and applications: A categorized survery. *EJOR*, 43(1):88–110, 1989.

[139] W. I. Zangwill. Minimizing a function without calculating derivatives. *Comput. J.*, 10:296–293, 1967.

[140] W. Zhai, P. Kelly, and W. B. Gong. Genetic algorithms with noisy fitness. *Math. Comp. Model.*, 23(11/12):131–142, 1996.

Appendix A

Nelder Mead Parameter Samples

Table A.1: Nelder–Mead Parameter Samples

Index	α	β	γ	σ
0	1	0.5	2	0.5
1	1.850	0.844	2.730	0.809
2	1.800	0.188	3.080	0.269
3	1.320	0.755	5.620	0.513
4	1.350	0.098	4.830	0.278
5	1.060	0.335	5.290	0.618
6	0.907	0.722	5.460	0.052
7	0.558	0.442	2.280	0.066
8	0.505	0.287	5.440	0.640
9	0.739	0.769	5.600	0.529
10	0.540	0.076	4.750	0.203
11	1.750	0.412	5.940	0.259
12	1.620	0.504	2.400	0.310
13	1.930	0.540	5.150	0.228
14	1.640	0.125	4.380	0.371
15	1.250	0.921	2.140	0.446
16	1.910	0.109	2.210	0.949
17	1.590	0.715	4.370	0.452
18	2.000	0.259	2.390	0.877
19	1.530	0.491	4.980	0.667
20	1.020	0.417	3.620	0.504
21	0.663	0.067	4.170	0.821
22	1.550	0.644	4.460	0.719
23	0.679	0.324	3.480	0.342

Table A.1: Nelder–Mead Parameter Samples

Index	α	β	γ	σ
24	1.710	0.357	4.790	0.433
25	1.380	0.392	2.180	0.913
26	0.524	0.894	2.030	0.540
27	0.782	0.230	3.200	0.779
28	1.480	0.299	2.560	0.630
29	1.050	0.534	3.120	0.494
30	1.770	0.583	2.800	0.846
31	1.190	0.241	5.400	0.899
32	1.410	0.801	5.240	0.595
33	0.703	0.151	2.850	0.861
34	1.030	0.511	3.520	0.934
35	0.774	0.948	3.840	0.247
36	0.968	0.635	4.210	0.326
37	1.280	0.363	5.850	0.890
38	0.919	0.438	4.570	0.226
39	1.950	0.861	3.010	0.772
40	0.719	0.693	4.610	0.523
41	1.270	0.695	3.820	0.757
42	0.604	0.342	5.060	0.800
43	1.680	0.175	4.950	0.715
44	1.890	0.137	3.270	0.406
45	1.150	0.594	3.730	0.375
46	1.210	0.793	4.240	0.359
47	1.790	0.520	4.300	0.138
48	1.510	0.779	2.600	0.652
49	0.752	0.600	4.510	0.563
50	1.430	0.396	5.660	0.193
51	1.830	0.090	3.360	0.586
52	1.490	0.311	4.030	0.167
53	0.618	0.807	4.890	0.141
54	1.140	0.463	3.950	0.608
55	1.570	0.681	2.300	0.741
56	1.220	0.272	3.310	0.685
57	1.650	0.858	2.670	0.730
58	1.110	0.380	4.090	0.118
59	0.588	0.216	5.970	0.294
60	0.804	0.164	4.630	0.158
61	1.720	0.477	2.960	0.574

Table A.1: Nelder–Mead Parameter Samples

Index	α	β	γ	σ
62	0.885	0.746	3.910	0.337
63	0.931	0.873	3.720	0.427
64	0.989	0.265	5.800	0.791
65	1.090	0.736	2.910	0.078
66	0.862	0.657	5.060	0.465
67	0.955	0.618	5.840	0.389
68	1.980	0.822	2.050	0.702
69	0.635	0.203	5.750	0.101
70	0.837	0.051	2.760	0.835
71	1.370	0.915	5.360	0.396
72	1.300	0.558	3.640	0.926
73	1.870	0.570	2.450	0.303
74	1.460	0.610	2.510	0.176
75	1.430	0.452	4.690	0.478
76	1.180	0.149	5.190	0.560
77	0.823	0.833	5.550	0.090
78	1.590	0.895	3.560	0.865
79	1.700	0.217	3.140	0.213
80	1.110	0.661	4.020	0.109
81	1.870	0.932	3.390	0.674

Appendix B

Shuffled Complex Evolution Parameter Samples

Table B.1: SCEM Parameter Samples

Index	p	m	q	β_S
1	1	6	2	1
2	1	6	2	2
3	1	6	2	3
4	1	6	3	1
5	1	6	3	2
6	1	6	3	3
7	1	6	4	1
8	1	6	4	2
9	1	6	4	3
10	1	7	2	1
11	1	7	2	2
12	1	7	2	3
13	1	7	3	1
14	1	7	3	2
15	1	7	3	3
16	1	7	4	1
17	1	7	4	2
18	1	7	4	3
19	1	8	2	1
20	1	8	2	2
21	1	8	2	3
22	1	8	3	1

Table B.1: SCEM Parameter Samples

Index	p	m	q	β_S
23	1	8	3	2
24	1	8	3	3
25	1	8	4	1
26	1	8	4	2
27	1	8	4	3
28	2	6	2	1
29	2	6	2	2
30	2	6	2	3
31	2	6	3	1
32	2	6	3	2
33	2	6	3	3
34	2	6	4	1
35	2	6	4	2
36	2	6	4	3
37	2	7	2	1
38	2	7	2	2
39	2	7	2	3
40	2	7	3	1
41	2	7	3	2
42	2	7	3	3
43	2	7	4	1
44	2	7	4	2
45	2	7	4	3
46	2	8	2	1
47	2	8	2	2
48	2	8	2	3
49	2	8	3	1
50	2	8	3	2
51	2	8	3	3
52	2	8	4	1
53	2	8	4	2
54	2	8	4	3
55	3	6	2	1
56	3	6	2	2
57	3	6	2	3
58	3	6	3	1
59	3	6	3	2
60	3	6	3	3

Table B.1: SCEM Parameter Samples

Index	p	m	q	β_S
61	3	6	4	1
62	3	6	4	2
63	3	6	4	3
64	3	7	2	1
65	3	7	2	2
66	3	7	2	3
67	3	7	3	1
68	3	7	3	2
69	3	7	3	3
70	3	7	4	1
71	3	7	4	2
72	3	7	4	3
73	3	8	2	1
74	3	8	2	2
75	3	8	2	3
76	3	8	3	1
77	3	8	3	2
78	3	8	3	3
79	3	8	4	1
80	3	8	4	2
81	3	8	4	3

Appendix C

Nelder Mead Numerical Results

Table C.1: NMA Numerical Results: Test Problem 1a

Index	Number of Simulations	Average Objective Function
0	1695	371.89
1	4950	375.69
2	935	401837.82
3	3895	29.74
4	795	1789495.37
5	2085	265.07
6	3615	10.30
7	1225	99803.00
8	1175	5436.58
9	2806	934.64
10	1526	385522.17
11	1894	16.55
12	1800	842.81
13	1821	356.35
14	747	2351338.13
15	10274	872.76
16	937	4065105.40
17	2829	405.08
18	1806	369235.87
19	1701	1390.19
20	2159	0.58
21	1985	2994.75

Table C.1: NMA Numerical Results: Test Problem 1*a*

Index	Number of Simulations	Average Objective Function
22	2860	325.00
23	3606	91.00
24	1984	61.87
25	1599	45.79
26	918	843734.40
27	2199	3.25
28	1998	325.95
29	2278	17.55
30	2628	538.71
31	1522	404.25
32	4603	691.34
33	1712	34122.91
34	2095	3.22
35	8001	1493.49
36	3209	0.89
37	1704	86.38
38	2659	0.47
39	5824	177.63
40	2757	24.91
41	3017	638.46
42	1722	1100.07
43	958	108392.24
44	1026	2179218.90
45	2302	2.01
46	4719	21.46
47	2401	224.97
48	3781	752.04
49	2238	30.98
50	2000	4.02
51	832	3672853.93
52	1754	769.71
53	3102	3140.53
54	1954	1.19
55	2806	250.75
56	1610	3215.98
57	5905	305.79

Table C.1: NMA Numerical Results: Test Problem 1*a*

Index	Number of Simulations	Average Objective Function
58	1696	9.80
59	1461	1282.81
60	2035	551.24
61	1896	58.08
62	3420	676.38
63	7161	15.87
64	1661	11.94
65	3674	165.25
66	2940	20.82
67	2798	0.40
68	4593	174.97
69	2041	499.70
70	1743	1040241.15
71	9796	504.39
72	2612	558.28
73	2174	143.78
74	2713	72.60
75	2047	8.44
76	1934	102430.53
77	3982	1276.53
78	9394	223.91
79	1826	564074.68
80	3302	18.59
81	11542	194.43

Table C.2: NMA Numerical Results: Test Problem 1*b*

Index	Number of Simulations	Average Objective Function	Average True Fit
0	1750	2522.63	68.12
1	4515	2868.61	301.10
2	960	826476.21	822389.40
3	3115	2619.21	210.81
4	780	1799008.25	1795338.01
5	1280	2568.09	107.68
6	2330	2666.25	322.00
7	1235	171652.89	169296.09
8	1020	8329.87	6075.88
9	2064	2863.18	281.57
10	1293	103636.44	102070.87
11	1244	2536.14	132.62
12	1322	2666.35	130.59
13	1702	3038.98	421.47
14	740	2356960.09	2351377.43
15	8762	3421.72	856.47
16	929	4019850.93	4014801.63
17	2287	2950.56	464.45
18	1348	374143.1	369598.86
19	1245	3814.51	1132.99
20	1132	2673.05	165.07
21	1632	4820.91	2170.85
22	2494	3023.66	473.31
23	1490	3045.43	696.96
24	1082	2622.71	203.53
25	1261	2596.88	145.81
26	895	834356.08	831728.62
27	1139	2481.77	62.27
28	1184	2585.07	179.08
29	1918	70621.26	69150.93
30	1505	4082.63	1434.03
31	1232	2593.81	243.33
32	3094	3790.81	1163.57
33	2885	37584.68	34438.61
34	1761	2486.61	62.66
35	5902	3859.16	1493.88

Table C.2: NMA Numerical Results: Test Problem 1*b*

Index	Number of Simulations	Average Objective Function	Average True Fit
36	2125	2535.07	76.53
37	932	3012.45	642.67
38	1283	2490.78	53.78
39	5232	2749.01	209.58
40	2199	2906	394.28
41	2631	2718.23	242.43
42	1004	3512.37	1270.01
43	983	112967.05	109399.52
44	884	2230870.37	2228584.45
45	2010	2487.75	56.59
46	3231	3810.13	1174.43
47	1526	2891.9	375.66
48	3294	3361.94	710.07
49	1751	2492.64	59.65
50	1098	2563.3	111.79
51	898	3635877.66	3633016.62
52	1562	2575.58	103.78
53	1861	33520.24	31577.39
54	1447	2497.42	55.64
55	2250	3099.15	591.29
56	1317	5565.55	3401.89
57	4733	3102.6	508.47
58	1218	2552.87	93.65
59	1536	5018.75	2311.55
60	1561	3121.79	828.23
61	1445	3777.19	1125.02
62	2887	3113.94	566.75
63	4665	2567.66	72.06
64	1535	2487.53	71.01
65	2789	2806.27	413.22
66	1716	2791.83	480.28
67	2047	2495.48	54.01
68	4073	2743.91	203.92
69	1292	2774.93	240.45
70	1352	1033779.54	1035414.40
71	8006	3369.76	803.81

Table C.2: NMA Numerical Results: Test Problem 1*b*

Index	Number of Simulations	Average Objective Function	Average True Fit
72	1749	2546.54	89.92
73	1528	2907.02	329.54
74	1717	3632.73	1062.95
75	1393	2717.9	249.52
76	1738	875636.14	870368.16
77	3371	3742.25	1367.88
78	6166	3178.54	604.69
79	1076	557725.33	553696.45
80	2308	2489.59	61.56
81	10371	2767.89	228.09

Table C.3: NMA Numerical Results: Test Problem 1*c*

Index	Number of Simulations	Average Objective Function	Average True Fit
0	1395	10456.29	509.76
1	4305	10282.35	351.81
2	870	426266.26	416153.89
3	2455	11424.58	1486.53
4	850	1813871.84	1804514.06
5	1080	10294.78	327.86
6	2215	10273.44	418.47
7	1120	112560.07	102672.33
8	1000	10843.33	908.01
9	2164	10324.61	441.34
10	1304	226268.81	217107.99
11	931	10979.77	1260.60
12	1181	10838.08	902.60
13	1583	10308.25	334.45
14	730	1788415.06	1779067.69
15	7518	10644.76	719.21
16	887	4019835.06	4010778.31
17	2190	10734.98	820.30
18	1783	378423.91	369249.50
19	1234	11636.16	1678.62

Table C.3: NMA Numerical Results: Test Problem 1*c*

Index	Number of Simulations	Average Objective Function	Average True Fit
20	1123	9968.55	50.12
21	1806	11260.60	1304.74
22	1819	11181.27	1196.16
23	2156	10003.18	94.56
24	1030	10094.39	168.83
25	1068	10135.31	214.67
26	872	847733.84	837821.60
27	1025	9934.57	6.45
28	705	12172.92	2236.87
29	1592	10327.85	450.45
30	1839	10594.74	622.23
31	1077	10128.98	163.35
32	3352	11012.38	1092.04
33	1943	43717.23	33988.37
34	1598	10027.48	56.12
35	6215	11729.97	1835.54
36	2168	9930.85	27.46
37	1052	10363.58	461.78
38	1262	9916.18	13.77
39	4993	10138.92	207.75
40	1506	10353.10	420.25
41	2392	10559.21	592.07
42	1006	11023.39	1151.63
43	845	122489.79	112912.18
44	971	2186110.97	2176235.97
45	1850	9973.89	25.34
46	3331	11367.92	1446.98
47	1214	10540.03	568.10
48	2643	11299.87	1320.98
49	1468	10134.02	290.33
50	1088	9948.89	31.98
51	769	3682629.78	3672857.26
52	961	11658.63	1672.56
53	1663	12443.01	2587.58
54	1237	10332.83	433.65
55	2144	10427.64	493.02

Table C.3: NMA Numerical Results: Test Problem 1*c*

Index	Number of Simulations	Average Objective Function	Average True Fit
56	1131	14289.58	4419.95
57	4438	10276.89	348.94
58	1184	10018.42	62.72
59	1338	11215.05	1315.86
60	1326	11094.16	1113.36
61	1080	10719.87	758.59
62	2309	13161.98	3253.75
63	4339	10015.03	64.10
64	1286	9969.99	21.63
65	3006	10046.70	177.71
66	2162	9890.95	17.21
67	1719	9975.97	23.72
68	3838	10113.78	180.36
69	1249	10372.40	448.70
70	1402	1043379.56	1031648.28
71	7033	11032.14	1076.12
72	1535	10255.34	303.43
73	1438	10359.21	393.38
74	1623	10523.92	559.96
75	1135	10508.59	601.00
76	1069	862424.17	853586.17
77	3113	11436.03	1535.03
78	6324	10426.95	516.93
79	1068	573564.95	564088.62
80	2190	10038.08	56.96
81	9855	10081.40	160.29

Table C.4: NMA Numerical Results: Test Problem 2*a*

Index	Number of Simulations	Average Objective Function
0	934	7590.13
1	2652	18916.69
2	412	25123.77
3	2036	21859.39

Table C.4: NMA Numerical Results: Test Problem 2*a*

Index	Number of Simulations	Average Objective Function
4	521	234636.56
5	959	14888.11
6	1844	7590.20
7	1765	36542.81
8	814	31649.28
9	1994	19296.01
10	608	1362588.41
11	896	27700.34
12	1234	11917.90
13	764	26299.64
14	425	286813.71
15	5343	20525.47
16	759	127921.07
17	1657	24431.30
18	483	29928.58
19	817	28589.14
20	965	7590.12
21	494	39781.18
22	1383	14758.33
23	1095	25552.97
24	742	27434.81
25	1121	7590.14
26	1183	1206825.45
27	1154	39957.02
28	887	18114.76
29	953	22095.25
30	1073	24640.47
31	886	30672.66
32	2035	24906.31
33	1213	28251.33
34	1291	7590.13
35	4219	40550.52
36	1531	7590.16
37	1299	7590.15
38	1104	7590.12
39	3088	18785.62

Table C.4: NMA Numerical Results: Test Problem 2*a*

Index	Number of Simulations	Average Objective Function
40	1484	24831.42
41	1501	20225.40
42	902	33270.61
43	745	153300.47
44	590	93211.59
45	1335	7590.15
46	2539	7590.31
47	890	28539.34
48	1965	21247.74
49	1143	7590.14
50	875	24901.78
51	523	129715.09
52	1152	17175.14
53	902	36614.10
54	1190	7590.15
55	1804	7590.21
56	1369	7608.11
57	3043	22893.94
58	1096	7590.14
59	647	41727.55
60	500	19718.10
61	937	22509.54
62	1845	7590.24
63	3635	7590.52
64	1321	7590.50
65	1944	12174.63
66	1521	7590.18
67	1679	7590.18
68	2176	26382.76
69	857	63898.51
70	532	66039.94
71	4639	25415.80
72	1001	18364.07
73	1096	17418.57
74	1245	22324.84
75	1007	19932.01

Table C.4: NMA Numerical Results: Test Problem 2*a*

Index	Number of Simulations	Average Objective Function
76	554	29136.23
77	2620	19261.62
78	3830	24501.87
79	520	55433.44
80	1616	7590.17
81	5622	23138.10

Table C.5: NMA Numerical Results: Test Problem 2*b*

Index	Number of Simulations	Average Objective Function	Average True Fit
0	1034	9334.96	7635.55
1	2488	26294.48	24512.00
2	463	26625.60	24936.49
3	1962	19252.12	17512.08
4	499	233805.26	234143.79
5	986	16689.72	14933.01
6	1699	13148.71	11429.30
7	2725	44065.38	42661.12
8	746	33295.72	31597.41
9	2232	29372.71	27624.65
10	519	1341655.87	1346231.50
11	997	22932.13	21179.57
12	1114	18929.89	17146.77
13	865	28477.92	26666.25
14	514	36552.32	34582.02
15	5103	22240.22	20458.12
16	591	125255.53	125071.14
17	1539	26746.72	24949.88
18	463	31754.85	29943.68
19	1379	9334.95	7635.49
20	1044	9334.92	7635.29
21	464	42485.04	40765.61
22	1147	25591.82	23795.94
23	1092	27326.15	25612.81

Table C.5: NMA Numerical Results: Test Problem 2*b*

Index	Number of Simulations	Average Objective Function	Average True Fit
24	737	24854.56	23066.30
25	1144	9335.03	7637.16
26	1124	1209201.97	1208983.63
27	1071	40887.80	40098.10
28	1178	15890.34	14057.17
29	1181	9334.96	7636.10
30	1028	23653.65	21874.30
31	847	29432.05	27670.40
32	2070	26733.00	24943.20
33	1562	18774.47	16587.16
34	1182	9335.02	7635.49
35	4351	31367.01	29522.32
36	1556	9334.99	7635.03
37	909	19324.32	17541.35
38	1049	9334.94	7635.46
39	2789	25324.50	23498.52
40	1409	25318.46	23586.36
41	1404	21973.60	20192.48
42	1018	27957.36	26148.69
43	642	132809.93	132409.21
44	693	95434.34	94946.20
45	1131	20345.82	18558.76
46	2341	14193.29	12463.41
47	1221	21720.58	19962.47
48	1844	25976.07	24190.51
49	1365	9334.95	7635.73
50	1183	10955.20	9227.05
51	534	156403.43	153232.30
52	770	22792.51	21028.66
53	888	33491.42	31549.10
54	1165	9334.95	7635.80
55	1509	16988.67	15246.10
56	768	22230.66	20481.52
57	2764	27154.21	25359.12
58	953	19982.22	18237.28
59	754	43330.84	41893.14

Table C.5: NMA Numerical Results: Test Problem 2*b*

Index	Number of Simulations	Average Objective Function	Average True Fit
60	964	22856.90	21152.41
61	820	24473.37	22688.61
62	1898	9335.12	7635.56
63	3618	9335.63	7634.15
64	1543	9339.57	7652.55
65	1658	24723.04	22949.87
66	1353	19076.35	17326.90
67	1311	17379.63	15627.83
68	2235	27284.57	25470.51
69	726	92934.47	92070.72
70	464	67397.44	66047.06
71	4448	27406.77	25604.06
72	954	19288.90	17504.10
73	1028	23865.73	22087.92
74	1072	26786.45	24994.64
75	992	23958.86	22196.67
76	522	33846.83	31988.76
77	2215	20236.14	18463.24
78	3906	25758.03	23956.55
79	533	56287.10	55193.60
80	1416	13784.39	12060.88
81	6110	20045.45	18248.25

Table C.6: NMA Numerical Results: Test Problem 2*c*

Index	Number of Simulations	Average Objective Function	Average True Fit
0	982	18462.90	7659.17
1	2620	32313.86	22979.78
2	422	34897.07	25123.43
3	1775	31594.88	21973.90
4	506	243058.73	234638.25
5	932	25140.79	14945.62
6	1899	18463.30	7660.65
7	1558	42055.54	32645.47

Table C.6: NMA Numerical Results: Test Problem 2*c*

Index	Number of Simulations	Average Objective Function	Average True Fit
8	801	38902.89	29995.95
9	1770	29070.30	19336.58
10	599	1372769.96	1365105.82
11	638	37171.40	28358.32
12	1100	23571.68	13494.86
13	877	35264.99	26297.87
14	464	295282.74	286439.33
15	4612	34636.59	25434.05
16	561	137913.48	127924.19
17	1630	29765.71	20038.75
18	451	38508.17	29930.45
19	758	37230.96	28393.41
20	955	18462.88	7659.37
21	535	50269.24	39787.73
22	1171	33695.67	24430.41
23	873	35450.35	25618.46
24	846	38096.16	28946.92
25	952	19712.06	9170.98
26	1181	1214857.47	1206856.83
27	847	51434.82	40069.90
28	833	32494.79	22610.97
29	957	29688.96	19893.76
30	1248	29448.83	19709.60
31	707	37031.11	27619.08
32	1909	37723.00	28929.39
33	1763	41287.40	32164.11
34	890	30580.90	20906.25
35	3848	48839.33	40867.23
36	1549	18463.11	7662.23
37	1292	18462.88	7660.47
38	999	18462.95	7658.55
39	2770	31785.78	22649.38
40	1393	34738.19	24888.78
41	1353	32564.11	23254.08
42	785	40170.99	30974.59
43	944	164394.05	155341.10

Table C.6: NMA Numerical Results: Test Problem 2*c*

Index	Number of Simulations	Average Objective Function	Average True Fit
44	558	102792.50	93235.37
45	1349	18462.95	7661.71
46	2105	31785.48	22181.02
47	861	37364.32	28552.41
48	1967	31043.19	21392.96
49	1314	18462.95	7660.82
50	1145	30340.24	20587.10
51	542	136439.77	129716.11
52	1068	25912.13	15751.68
53	786	45647.84	37546.94
54	1131	18463.06	7664.34
55	1410	26431.61	16567.64
56	1082	19749.09	9218.30
57	2640	36140.22	27232.22
58	1151	18463.01	7658.70
59	829	46771.34	38126.93
60	548	29212.35	19666.86
61	1003	31937.16	22537.32
62	1794	18463.27	7663.85
63	3162	27059.37	16949.86
64	1275	20300.81	9628.22
65	1970	18463.21	7660.61
66	1503	18463.16	7659.03
67	1389	22509.67	12177.94
68	2199	34362.76	25365.68
69	726	73059.86	64767.50
70	514	74332.64	66036.84
71	4457	34749.30	25541.22
72	939	27760.29	17948.16
73	1041	27066.68	17442.73
74	1151	31976.78	22366.17
75	977	29744.59	19976.11
76	577	73568.46	65500.53
77	2271	26876.44	16993.73
78	3477	36666.86	27810.61
79	500	64088.10	55269.39

Table C.6: NMA Numerical Results: Test Problem 2*c*

Index	Number of Simulations	Average Objective Function	Average True Fit
80	1446	27094.42	16984.54
81	5235	32213.01	23150.63

Appendix D

Shuffled Complex Evolution Numerical Results

Table D.1: SCEM Numerical Results: Test Problem 1*a*

Index	Number of Simulations	Average Objective Function	Premature Terminations
1	2507	1086.94	0
2	15358	3473.58	2
3	21856	941.92	4
4	3215	4779.82	0
5	4766	2628.04	0
6	7214	675.48	0
7	5441	350.56	0
8	4200	44.24	0
9	6519	1463.85	0
10	3420	35783.79	0
11	25005	4253.70	5
12	24378	1818.49	4
13	3687	70.60	0
14	4989	119.49	0
15	17561	62.34	1
16	3230	21.25	0
17	3768	27.45	0
18	2591	219.57	0
19	5423	807.68	0
20	21933	1467.50	4

Table D.1: SCEM Numerical Results: Test Problem 1*a*

Index	Number of Simulations	Average Objective Function	Premature Terminations
21	25005	1917.52	5
22	4660	401.40	0
23	9916	6.62	0
24	12359	12.88	1
25	4008	9.40	0
26	5508	52.34	0
27	4553	32.21	0
28	5737	4980.79	0
29	7595	115.64	0
30	6670	551.57	0
31	5678	25.98	0
32	6510	6.00	0
33	7732	8.41	0
34	9818	17.55	0
35	7861	3.73	0
36	7077	1.08	0
37	9334	36.47	0
38	7822	9.71	0
39	12262	42.28	1
40	9979	1.55	1
41	11716	34.36	0
42	6713	6.45	0
43	9811	3.17	0
44	14604	0.05	0
45	12880	0.06	0
46	9935	3.71	0
47	16784	5.57	1
48	13477	44.80	0
49	17670	0.42	2
50	18236	0.16	3
51	21806	0.03	4
52	14780	2.06E-18	0
53	14035	2.20E-18	0
54	16291	1.11E-18	0
55	11801	18.35	0
56	16274	2.45	1

Table D.1: SCEM Numerical Results: Test Problem 1*a*

Index	Number of Simulations	Average Objective Function	Premature Terminations
57	23322	3.45	3
58	14165	3.20E-18	0
59	18496	2.98E-17	0
60	18462	2.98	1
61	12210	1.05E-18	0
62	11601	1.77E-18	0
63	14586	8.97E-19	0
64	20294	0.33	3
65	17999	1.22	1
66	13879	1.50	1
67	13865	1.53E-18	0
68	14577	1.57E-18	0
69	17474	7.21E-19	0
70	13009	1.22E-18	0
71	12373	1.24E-18	0
72	13857	5.53E-19	0
73	25005	1.96E-04	5
74	20271	0.27	3
75	25005	0.27	5
76	16555	5.34E-19	0
77	14857	0.30	0
78	16330	2.29E-18	0
79	14298	9.01E-19	0
80	14584	6.53E-19	0
81	14968	8.01E-19	0

Table D.2: SCEM Numerical Results: Test Problem 1*b*

Index	Number of Simulations	Average Objective Function	Average True Fit	Premature Terminations
1	2892	19796.08	16868.23	0
2	17717	3561.58	1128.96	2
3	23181	6521.23	3573.31	4
4	2908	4091.38	1410.30	0
5	2625	3291.42	691.43	0

Table D.2: SCEM Numerical Results: Test Problem 1*b*

Index	Number of Simulations	Average Objective Function	Average True Fit	Premature Terminations
6	2754	2503.95	86.72	0
7	2664	2574.34	161.15	0
8	3066	2621.74	217.98	0
9	2240	2481.25	55.51	0
10	4237	51924.67	48637.44	0
11	17909	3640.33	1040.94	3
12	23443	7556.46	4791.48	4
13	2766	1427364.29	1420904.96	0
14	3453	100244.69	96878.93	0
15	9345	4414.74	1776.75	0
16	1908	2477.57	61.91	0
17	3147	2726.08	389.34	0
18	2275	2756.70	251.18	0
19	4676	1135437.83	1132404.96	0
20	25005	2683.43	245.46	5
21	25005	3873.90	1294.09	5
22	2938	2506.36	66.88	0
23	2641	2484.84	71.28	0
24	7587	2503.02	74.55	0
25	2841	2543.03	90.92	0
26	3683	2996.04	475.41	0
27	3917	2563.17	105.49	0
28	2809	2483.00	72.38	0
29	5221	2541.42	102.53	0
30	5289	2501.98	76.00	0
31	4013	2477.60	57.18	0
32	4753	2500.90	74.01	0
33	2799	2635.79	167.17	0
34	3176	2472.97	59.74	0
35	3036	2479.32	62.35	0
36	2866	2477.71	58.65	0
37	3173	2497.03	83.76	0
38	4960	2887.70	393.97	0
39	6058	2541.80	99.70	0
40	2974	2475.21	60.30	0
41	4563	2475.96	59.69	0

Table D.2: SCEM Numerical Results: Test Problem 1*b*

Index	Number of Simulations	Average Objective Function	Average True Fit	Premature Terminations
42	3450	2499.23	66.55	0
43	3934	2474.40	58.42	0
44	3841	2486.13	64.38	0
45	4508	2473.03	61.64	0
46	4554	2477.09	55.94	0
47	7883	2482.45	61.04	0
48	4475	2749.00	228.76	0
49	3954	2474.03	59.36	0
50	4563	2474.68	60.32	0
51	4131	2481.06	62.80	0
52	4454	2475.43	63.55	0
53	4020	2476.99	58.80	0
54	5849	2474.34	60.35	0
55	4684	2493.93	66.05	0
56	8986	2519.51	83.72	0
57	4482	2477.25	62.79	0
58	5270	2475.32	59.24	0
59	4355	2473.78	62.54	0
60	5852	2473.53	63.83	0
61	4757	2476.38	63.94	0
62	4240	2474.90	58.75	0
63	5401	2472.36	60.04	0
64	5904	2475.09	58.50	0
65	5142	2475.90	60.86	0
66	6522	2484.80	67.53	0
67	6498	2471.30	61.24	0
68	5600	2472.02	60.74	0
69	4444	2473.99	60.02	0
70	5515	2473.52	58.89	0
71	5761	2472.55	59.73	0
72	5214	2472.63	59.61	0
73	6415	2479.00	64.47	0
74	9660	2474.58	59.66	0
75	6773	2488.53	75.89	0
76	6010	2472.98	59.44	0
77	5615	2475.39	58.96	0

Table D.2: SCEM Numerical Results: Test Problem 1*b*

Index	Number of Simulations	Average Objective Function	Average True Fit	Premature Terminations
78	6648	2470.04	59.61	0
79	6352	2471.86	60.04	0
80	5513	2471.53	59.79	0
81	5906	2471.17	60.33	0

Table D.3: SCEM Numerical Results: Test Problem 1*c*

Index	Number of Simulations	Average Objective Function	Average True Fit	Premature Terminations
1	2423	13468.42	3483.63	0
2	5179	10798.48	741.64	0
3	21438	14779.99	4815.06	3
4	2149	11106.32	1125.79	0
5	2652	9954.37	70.17	0
6	4701	12767.57	2798.89	0
7	2518	50289.33	40653.31	0
8	1911	11729.42	1750.96	0
9	2208	13932.74	4037.83	0
10	3639	90902.88	81152.56	0
11	22734	10321.47	364.84	4
12	25005	15445.72	5428.97	5
13	2718	10235.37	284.77	0
14	4978	12792.85	2881.83	0
15	5748	10071.5	56.06	0
16	2794	9969.74	146.2	0
17	2133	9902.61	22.02	0
18	2766	9937.94	31.88	0
19	4608	13340.06	3380.91	0
20	25005	11366.13	1366.34	5
21	25005	12134.4	2129.36	5
22	2437	10013.92	33.88	0
23	5997	11549.19	1631.56	0
24	12791	10023.09	61.73	0
25	1986	9929.4	15.19	0
26	2335	10008.65	46	0

Table D.3: SCEM Numerical Results: Test Problem 1*c*

Index	Number of Simulations	Average Objective Function	Average True Fit	Premature Terminations
27	2454	35845.09	25652.48	0
28	4207	9935.01	26.34	0
29	3382	10048.87	206.01	0
30	4345	10018.61	32.62	0
31	3039	9870.66	71.77	0
32	4131	9892.02	52.78	0
33	3681	9856.65	58.39	0
34	3354	9889.43	56.23	0
35	3320	9834.17	49.43	0
36	3661	9799.06	74.79	0
37	4249	10009.74	36.76	0
38	4388	10074.54	90.29	0
39	5690	11204.66	1210.64	0
40	4686	9801.97	78.13	0
41	4071	9876.08	21.09	0
42	5115	10146.89	368.22	0
43	3900	9861.72	33.47	0
44	4317	9845.98	50.53	0
45	4247	9859	35.32	0
46	6384	9914.81	68.54	0
47	7044	9847.55	55.36	0
48	6875	9837.66	41.22	0
49	4668	9834.83	68.14	0
50	6787	9783.33	97.41	0
51	6190	9780.26	65.48	0
52	4153	9778.49	80.11	0
53	5936	9757.76	123.12	0
54	4785	9789.84	68.32	0
55	5358	9913.12	27.75	0
56	5423	10045.65	217.04	0
57	8465	9910.66	74.23	0
58	5739	9767.33	100.79	0
59	7159	9797.25	87.03	0
60	6765	9767.67	122.91	0
61	4885	9764.57	96.76	0
62	6399	9775.08	111.67	0

Table D.3: SCEM Numerical Results: Test Problem 1*c*

Index	Number of Simulations	Average Objective Function	Average True Fit	Premature Terminations
63	5593	9779.3	86.53	0
64	6223	9879.95	34.45	0
65	7138	9874.04	31.9	0
66	8229	9863.71	42.33	0
67	6962	9758.17	130.22	0
68	5946	9758.7	122.32	0
69	7527	9782.55	110.97	0
70	6869	9755.27	135.62	0
71	6564	9753.01	151.58	0
72	6470	9792.22	90.49	0
73	6485	9781.9	104.87	0
74	11058	9854.12	67.26	0
75	11211	9866.11	60.82	0
76	7362	9754.99	136.98	0
77	7480	9752.81	148.46	0
78	6765	9752.65	150.52	0
79	7056	9752.9	144.03	0
80	6300	9752.65	153.34	0
81	7687	9752.71	148.82	0

Table D.4: SCEM Numerical Results: Test Problem 2*a*

Index	Number of Simulations	Average Objective Function	Premature Terminations
1	2363	24041.44	0
2	10348	32517.35	0
3	23265	18417.91	4
4	1978	10298.90	0
5	2484	7844.22	0
6	3114	7622.48	0
7	2164	10001.45	0
8	2380	7593.65	0
9	3153	7619.73	0
10	3237	146562.68	0
11	25005	10243.81	5

Table D.4: SCEM Numerical Results: Test Problem 2*a*

Index	Number of Simulations	Average Objective Function	Premature Terminations
12	21701	15046.95	3
13	3104	7748.73	0
14	3393	7590.11	0
15	4797	7590.10	0
16	1977	7590.10	0
17	2323	7590.10	0
18	2255	7590.10	0
19	5792	10840.72	0
20	25005	11328.19	5
21	25005	10924.91	5
22	5426	7590.10	0
23	9534	7590.10	1
24	7302	7590.10	0
25	2166	7590.10	0
26	3640	7590.10	0
27	2432	7590.10	0
28	2596	7590.25	0
29	3652	7591.06	0
30	6436	7590.22	0
31	2009	7590.13	0
32	2064	7590.13	0
33	2051	7590.12	0
34	2044	7590.15	0
35	2029	7590.12	0
36	2199	7590.12	0
37	2678	7590.13	0
38	3314	7590.13	0
39	5887	7590.25	0
40	2354	7590.14	0
41	2237	7590.12	0
42	2447	7590.12	0
43	2230	7590.14	0
44	2455	7590.12	0
45	2546	7590.11	0
46	2763	7590.12	0
47	3580	7590.10	0

Table D.4: SCEM Numerical Results: Test Problem 2*a*

Index	Number of Simulations	Average Objective Function	Premature Terminations
48	3503	7590.32	0
49	2542	7590.12	0
50	2546	7590.12	0
51	2757	7590.11	0
52	2512	7590.14	0
53	2628	7590.11	0
54	2722	7590.12	0
55	3070	7590.15	0
56	3502	7590.12	0
57	3686	7590.56	0
58	2897	7590.15	0
59	2796	7590.13	0
60	3078	7590.13	0
61	3126	7590.15	0
62	2967	7590.16	0
63	2974	7590.13	0
64	3224	7590.12	0
65	3478	7590.11	0
66	3676	7590.11	0
67	3121	7590.12	0
68	3120	7590.15	0
69	3154	7590.15	0
70	3501	7590.15	0
71	3631	7590.16	0
72	3416	7590.13	0
73	3769	7590.12	0
74	4008	7590.11	0
75	3964	7590.11	0
76	3527	7590.13	0
77	3653	7590.12	0
78	3753	7590.16	0
79	3897	7590.13	0
80	3787	7590.13	0
81	3802	7590.12	0

Table D.5: SCEM Numerical Results: Test Problem 2*b*

Index	Number of Simulations	Average Objective Function	Average True Fit	Premature Terminations
1	2498	32607.57	30939.44	0
2	16233	22291.92	20449.56	0
3	21492	23071.07	21282.29	3
4	2729	14736.08	12995.92	0
5	3290	20515.09	18802.33	0
6	3146	10534.66	8841.97	0
7	3527	9397.60	7688.78	0
8	2738	9766.38	8047.60	0
9	1966	11633.88	9994.05	0
10	4047	24390.06	22723.97	0
11	19586	25845.92	24067.32	3
12	25005	11039.31	9312.77	5
13	3408	9335.60	7633.32	0
14	4663	11845.22	10161.43	0
15	4646	9359.19	7664.41	0
16	2558	9334.91	7636.15	0
17	2622	9334.94	7635.91	0
18	2186	9334.91	7635.59	0
19	5860	9341.15	7652.36	0
20	24748	9342.77	7648.51	4
21	25005	11184.65	9565.03	5
22	2521	9334.91	7635.89	0
23	4856	9334.90	7635.91	0
24	13617	9334.90	7635.88	1
25	2090	9334.90	7635.86	0
26	1998	9334.90	7636.04	0
27	2268	9334.90	7635.93	0
28	3070	9334.95	7637.06	0
29	3323	9338.81	7653.48	0
30	4435	9334.94	7635.61	0
31	1979	9334.94	7635.74	0
32	1967	9334.95	7635.69	0
33	2126	9334.93	7635.03	0
34	2015	9334.96	7637.27	0
35	1993	9334.94	7634.47	0
36	2159	9334.95	7636.65	0

Table D.5: SCEM Numerical Results: Test Problem 2*b*

Index	Number of Simulations	Average Objective Function	Average True Fit	Premature Terminations
37	2335	9334.96	7637.11	0
38	4584	9334.92	7635.58	0
39	3909	9334.99	7636.85	0
40	2264	9334.92	7635.57	0
41	2215	9334.92	7635.63	0
42	2479	9334.94	7636.64	0
43	2405	9334.93	7635.27	0
44	2441	9334.94	7636.23	0
45	2358	9334.93	7634.44	0
46	2898	9334.94	7635.53	0
47	3040	9334.91	7635.75	0
48	4460	9334.94	7634.57	0
49	2447	9334.92	7635.76	0
50	2696	9334.92	7636.75	0
51	2591	9334.92	7636.15	0
52	2758	9334.94	7635.31	0
53	2670	9334.95	7637.31	0
54	2609	9334.95	7635.49	0
55	3031	9334.91	7635.46	0
56	3274	9335.02	7635.26	0
57	3698	9334.94	7635.87	0
58	2830	9334.95	7634.99	0
59	2827	9334.92	7637.07	0
60	2863	9334.93	7634.99	0
61	2929	9334.94	7636.55	0
62	2849	9334.99	7635.65	0
63	2935	9334.95	7636.50	0
64	3396	9334.92	7636.26	0
65	3639	9334.99	7634.45	0
66	3719	9334.94	7634.18	0
67	3046	9334.96	7636.85	0
68	3295	9334.93	7636.15	0
69	3302	9334.93	7636.25	0
70	3199	9334.94	7635.82	0
71	3426	9334.95	7636.43	0
72	3331	9334.96	7635.18	0

Table D.5: SCEM Numerical Results: Test Problem 2*b*

Index	Number of Simulations	Average Objective Function	Average True Fit	Premature Terminations
73	3524	9334.92	7635.86	0
74	3972	9334.91	7635.80	0
75	3724	9334.93	7635.32	0
76	3602	9334.95	7637.16	0
77	3600	9334.93	7636.04	0
78	3546	9334.93	7635.81	0
79	3703	9334.95	7636.57	0
80	3675	9334.95	7635.89	0
81	3926	9334.93	7635.72	0

Table D.6: SCEM Numerical Results: Test Problem 2*c*

Index	Number of Simulations	Average Objective Function	Average True Fit	Premature Terminations
1	2571	272390.34	262661.10	0
2	12559	55394.46	46698.32	0
3	18940	24609.02	14791.52	2
4	2531	20158.74	9503.93	0
5	1816	19452.11	8324.13	0
6	4325	20022.64	9511.47	0
7	2393	20617.26	9983.64	0
8	2248	18466.15	7667.91	0
9	3870	18495.14	7729.99	0
10	3421	30566.57	21335.10	0
11	23214	23582.27	13541.25	3
12	21221	25188.84	15394.97	3
13	7150	18468.19	7664.71	0
14	2586	18613.67	7733.19	0
15	4374	18462.96	7657.40	0
16	2507	23763.15	12417.31	0
17	1984	18462.85	7660.41	0
18	1600	18462.85	7660.22	0
19	5403	52413.35	43777.60	0
20	20925	19910.03	9421.43	4
21	25005	23511.34	13393.89	5

Table D.6: SCEM Numerical Results: Test Problem 2*c*

Index	Number of Simulations	Average Objective Function	Average True Fit	Premature Terminations
22	2492	18533.43	7769.48	0
23	4777	18462.84	7659.89	0
24	6722	18462.84	7659.82	0
25	2985	23848.78	12502.63	0
26	1934	18462.84	7659.69	0
27	2744	18462.84	7659.59	0
28	2774	18463.01	7658.48	0
29	2903	18463.03	7663.16	0
30	3969	19468.60	8900.35	0
31	2028	18462.92	7658.79	0
32	2163	18462.95	7663.23	0
33	2324	18462.89	7660.14	0
34	2012	18462.94	7662.49	0
35	2142	18462.91	7659.18	0
36	2155	18462.92	7658.88	0
37	2519	18462.89	7659.62	0
38	3640	18462.89	7661.22	0
39	3914	18463.09	7661.01	0
40	2351	18462.91	7660.39	0
41	2295	18462.88	7659.76	0
42	2398	18462.90	7658.75	0
43	2213	18462.95	7662.45	0
44	2331	18462.95	7660.40	0
45	2255	18462.90	7657.92	0
46	3150	18462.99	7660.09	0
47	3043	18463.03	7661.03	0
48	3141	18462.89	7660.63	0
49	2612	18462.91	7658.54	0
50	2618	18462.87	7659.61	0
51	2652	18462.87	7658.27	0
52	2509	18462.90	7659.36	0
53	2508	18462.88	7659.75	0
54	2563	18462.93	7660.22	0
55	2715	18462.94	7660.22	0
56	3262	18462.99	7662.34	0
57	3266	18462.92	7660.36	0

Table D.6: SCEM Numerical Results: Test Problem 2*c*

Index	Number of Simulations	Average Objective Function	Average True Fit	Premature Terminations
58	2931	18462.96	7658.59	0
59	2794	18462.93	7660.31	0
60	3077	18462.88	7660.06	0
61	2711	18462.98	7658.98	0
62	2771	18462.93	7661.52	0
63	2969	18462.95	7659.16	0
64	3010	18462.88	7660.51	0
65	3424	18462.93	7660.20	0
66	3781	18462.90	7661.14	0
67	2946	18462.89	7660.25	0
68	3005	18462.90	7660.14	0
69	3084	18462.91	7660.70	0
70	3224	18462.96	7661.19	0
71	3224	18462.91	7660.24	0
72	3380	18462.92	7659.94	0
73	3510	18462.88	7659.04	0
74	3816	18462.90	7659.48	0
75	4457	18462.89	7660.68	0
76	3534	18462.90	7661.13	0
77	3461	18462.96	7659.03	0
78	3809	18462.88	7659.05	0
79	3778	18462.89	7658.85	0
80	3625	18462.93	7662.12	0
81	3595	18462.90	7659.55	0

Vita

Kay Ellen White Vugrin was born in Lubbock, Texas to Ken and Barbara White on November 16, 1976. She graduated with Honors from Lubbock High School in 1995. She earned a Bachelor's Degree in Industrial Engineering and graduated Summa Cum Laude from Texas Tech University in 2000. Kay was awarded a National Science Foundation Graduate Research Fellowship to support her graduate studies and completed a Master's degree in Mathematics at Virginia Polytechnic Institute and State University in 2003. She is currently employed by Sandia National Laboratories. This dissertation is the final requirement for her Doctoral Degree in Mathematics from Virginia Polytechnic Institute and State University.