SIMULATION-BASED DESIGN USING VARIABLE FIDELITY OPTIMIZATION

A Dissertation

Submitted to the Graduate School
of the University of Notre Dame
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

by

Shawn Elam Gano, B.S.E., M.S.

______________________________
John E. Renaud, Director

Graduate Program in Aerospace and Mechanical Engineering
Notre Dame, Indiana
April 2005
Abstract

by

Shawn Elam Gano

Simulation-based design optimization is playing an increasingly prominent role in the design of everything from spacecraft to consumer products. Applying non-linear optimization techniques to simulation-based design becomes prohibitively expensive as computer models become more complex and increase in fidelity. A common engineering practice is to drive the preliminary design process using lower fidelity models as surrogates of expensive high fidelity simulations. Higher fidelity models are then used in the final design stages to refine the design. However, using automated optimization methods at this stage may still require enormous computational resources. Recently, variable fidelity schemes have been developed to address this problem by incorporating both models into one optimization framework. In these methods the low fidelity models are scaled to approximate the high fidelity simulations. This scaling allows the optimization to be performed using mainly low fidelity function calls, reducing the overall computational cost, while requiring only a few high fidelity evaluations to update the scaling function. Currently, two main scaling varieties are used: first order multiplicative and first order additive. In the multiplicative approach the low fidelity model is multiplied by a scaling function to approximate the high fidelity model; simi-
larly, in the additive approach a scaling function is added to the low fidelity model.

The focus of this dissertation is on improving the efficiency and applicability of variable fidelity optimization algorithms. Highlights of original contributions made in this research include: (1) An adaptive hybrid scaling method that relieves designers from having to choose \textit{a priori} which scaling method, multiplicative or additive, is most suitable to their problem with limited information. (2) Second order scaling methods which use approximate Hessian information, resulting in super-linear convergence rates. (3) A kriging-based global scaling method, which uses past design information to improve the global accuracy of the scaling model and was shown to reduce the computational cost of optimization by over 60% compared to single fidelity methods. (4) A metamodel update management strategy to reduce the cost of using kriging metamodels sequentially in large design problems. (5) Extension of the variable fidelity framework to solve reliability based design problems, which significantly lowers the computational cost, compared to traditional methods.
To my family.

“Luck is the residue of design.”
- Branch Rickey, former owner of the Brooklyn Dodgers

“Engineering is the practice of safe and economic application of the scientific laws governing the forces and materials of nature by means of organization, design and construction, for the general benefit of mankind.”
- S. E. Lindsay

“The engineer’s first problem in any design situation is to discover what the problem really is.”
- Unknown

“It is impossible to design a system so perfect that no one needs to be good.”
- T. S. Eliot

“A common mistake that people make when trying to design something completely foolproof is to underestimate the ingenuity of complete fools.”
- Douglas Adams
CHAPTER 3: TAYLOR SERIES BASED SCALING APPROXIMATIONS

3.1 First Order Scaling Models

3.1.1 First Order Multiplicative Scaling

3.1.2 First Order Additive Scaling

3.1.3 First Order Adaptive Hybrid Scaling

3.2 Second Order Scaling Models

3.2.1 Multiplicative Second Order Scaling

3.2.2 Additive Second Order Scaling

3.2.3 Quasi-Second Order Scaling

3.2.3.1 Initial Scaling of the Hessian

3.2.3.2 The BFGS and Damped BFGS Updates

3.2.3.3 The Symmetric-Rank-1 Update

3.2.4 Second Order Adaptive Hybrid Scaling

3.2.4.1 Example: Second Order Adaptive Hybrid Scaling

3.2.4.2 Example: Summary of Taylor Series Based Scaling Methods

3.3 Numerical Implementation Studies

3.3.1 2D Analytic Problem

3.3.2 Energy Efficient Transport High-Lift Airfoil Design

3.4 Summary

CHAPTER 4: KRIGING-BASED SCALING APPROXIMATIONS

4.1 Kriging Overview

4.2 Warm Starting the Kriging Method

4.3 Supplying Gradient Information

4.4 Numerical Implementation Studies

4.4.1 2D Analytic Problem

4.4.2 Energy Efficient Transport High-Lift Airfoil Design

4.5 Summary and Conclusions

CHAPTER 5: METAMODEL UPDATE MANAGEMENT STRATEGIES FOR KRIGING MODELS

5.1 Optimization Methods for Finding the Kriging Model Parameters

5.1.1 Constructing the Kriging Model Using a Quasi-Newton Method

5.1.2 Constructing the Kriging Model Using Pattern Search

5.1.3 Constructing the Kriging Model Using Adaptive Simulated Annealing

5.1.4 Comparison Between Optimizers When Building a Kriging Model: 1D Example

5.2 Metamodel Update Management Strategies For Sequentially Building Kriging Models
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.4</td>
<td>Combustion chamber geometry.</td>
<td>98</td>
</tr>
<tr>
<td>5.5</td>
<td>Cantilever beam with actuators.</td>
<td>100</td>
</tr>
<tr>
<td>5.6</td>
<td>Dependency diagram of the Control-Augmented Structure design problem</td>
<td>100</td>
</tr>
<tr>
<td>6.1</td>
<td>Reliability based optimization produce designs away from failure driven constraints</td>
<td>115</td>
</tr>
<tr>
<td>6.2</td>
<td>Transformation of the failure domain to a normalize space makes it possible to locate the MPP</td>
<td>119</td>
</tr>
<tr>
<td>6.3</td>
<td>VF-RBDO framework</td>
<td>125</td>
</tr>
<tr>
<td>6.4</td>
<td>High fidelity design space of the Barnes Problem.</td>
<td>129</td>
</tr>
<tr>
<td>6.5</td>
<td>Low fidelity design space of the Barnes Problem.</td>
<td>130</td>
</tr>
<tr>
<td>6.6</td>
<td>Initial and reliable optimum design configurations of the AEET.</td>
<td>136</td>
</tr>
</tbody>
</table>
### TABLES

3.1 VARIABLE FIDELITY OPTIMIZATION RESULTS FOR THE ANALYTIC TEST PROBLEM .......................... 58
3.2 VARIABLE FIDELITY OPTIMIZATION TAYLOR SERIES BASED RESULTS FOR THE AEET PROBLEM ............................. 62
4.1 VARIABLE FIDELITY OPTIMIZATION RESULTS FOR THE ANALYTIC TEST PROBLEM .......................... 77
4.2 VARIABLE FIDELITY OPTIMIZATION RESULTS FOR THE AEET PROBLEM .......................... 79
5.1 OPTIMIZATION RESULTS FOR THE ONE DIMENSIONAL EXAMPLE .......................... 92
5.2 STARTING AND OPTIMUM DESIGNS FOR THE INTERNAL COMBUSTION ENGINE DESIGN PROBLEM .......................... 99
5.3 STARTING AND OPTIMUM DESIGNS FOR THE CONTROLS-AUGMENTED STRUCTURE PROBLEM .......................... 101
5.4 INTERNAL COMBUSTION ENGINE OPTIMIZATION RESULTS .......................... 104
5.5 CONTROL-AUGMENTED STRUCTURE OPTIMIZATION RESULTS .......................... 107
6.1 COEFFICIENTS FOR THE BARNES PROBLEM .......................... 128
6.2 VF-RBDO DOUBLE-LOOP PMA RESULTS FOR THE MODIFIED BARNES PROBLEM .......................... 131
6.3 VF-RBDO DOUBLE-LOOP PMA RESULTS FOR THE AEET PROBLEM .......................... 135
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\beta$</td>
<td>Multiplicative scaling function</td>
</tr>
<tr>
<td>$\beta_i$</td>
<td>Reliability index of the $i^{th}$ failure mode</td>
</tr>
<tr>
<td>$\Delta$</td>
<td>Trust region size</td>
</tr>
<tr>
<td>$\epsilon_f$</td>
<td>Objective function convergence tolerance</td>
</tr>
<tr>
<td>$\epsilon_x$</td>
<td>Design variable convergence tolerance</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>Additive scaling function</td>
</tr>
<tr>
<td>$g^{rc}$</td>
<td>Reliability inequality constraints</td>
</tr>
<tr>
<td>$U$</td>
<td>Standard normal random variables</td>
</tr>
<tr>
<td>$V$</td>
<td>Vector of random variables</td>
</tr>
<tr>
<td>$\eta$</td>
<td>Limit state parameters</td>
</tr>
<tr>
<td>$\theta$</td>
<td>Distribution parameters of the random variables $V$</td>
</tr>
<tr>
<td>$g^D$</td>
<td>Deterministic constraints</td>
</tr>
<tr>
<td>$g^R$</td>
<td>Failure drive or probabilistic constraints</td>
</tr>
<tr>
<td>$l$</td>
<td>Design space lower bounds</td>
</tr>
<tr>
<td>Symbol</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td><strong>p</strong></td>
<td>Parameters that are fixed in the design problem</td>
</tr>
<tr>
<td><strong>R</strong></td>
<td>Correlation matrix</td>
</tr>
<tr>
<td><strong>r</strong></td>
<td>Correlation vector</td>
</tr>
<tr>
<td><strong>u</strong></td>
<td>Design space upper bounds</td>
</tr>
<tr>
<td><strong>v</strong></td>
<td>Realization of the random variables <strong>V</strong></td>
</tr>
<tr>
<td><strong>x</strong></td>
<td>Design vector</td>
</tr>
<tr>
<td><strong>x^l</strong></td>
<td>Lower bound of the design vector</td>
</tr>
<tr>
<td><strong>x^u</strong></td>
<td>Upper bound of the design vector</td>
</tr>
<tr>
<td><strong>L</strong></td>
<td>Log-likelihood function</td>
</tr>
<tr>
<td><strong>W</strong></td>
<td>Hybrid weighting value</td>
</tr>
<tr>
<td><strong>ρ</strong></td>
<td>Trust region ratio</td>
</tr>
<tr>
<td><strong>σ^2</strong></td>
<td>Variance</td>
</tr>
<tr>
<td><strong>g</strong></td>
<td>Inequality constraint</td>
</tr>
<tr>
<td><strong>h</strong></td>
<td>Equality constraint</td>
</tr>
<tr>
<td><strong>Θ</strong></td>
<td>Damping factor in damped BFGS method</td>
</tr>
<tr>
<td><strong>θ</strong></td>
<td>Correlation parameters</td>
</tr>
<tr>
<td><strong>ϱ</strong></td>
<td>Likelihood ratio</td>
</tr>
<tr>
<td><strong>A_i</strong></td>
<td>The i-th kriging model management parameter</td>
</tr>
</tbody>
</table>
\( c_d \)  Sectional drag coefficient

\( c_l \)  Sectional lift coefficient

\( c_m \)  Sectional moment coefficient about the aerodynamic center

\( f \)  Objective or merit function

\( f_N(v) \)  Joint probability density function (PDF) of \( V \)

\( H \)  Hessian matrix

\( I \)  Identity matrix

\( K \)  Optimization scheme used to determine the kriging model parameters

\( L \)  Likelihood function

\( MSE \)  Mean squared error

\( N_D \)  Number of deterministic constraints

\( N_R \)  Number of failure driven or probabilistic constraints

\( n_s \)  Number of sample sites used in kriging model

\( n_v \)  Number of variables used in kriging model

\( P \)  Probability of failure

\( s_n \)  Change in the design vector from the past iteration

\( y_n \)  Change in the gradient of the objective function from the past iteration

\( B \)  Global kriging trend function
R Correlation function

y Function modeled using kriging

Z Stochastic process model in kriging

\textit{high} High fidelity model

\textit{i,j,k} Free indices for elements in a vector or matrix

\textit{low} Low fidelity model

m Number of design variables

n Current iteration number

\textit{scaled} Scaled low fidelity value

t Target value

\textit{u} Denotes an unscaled constraint

\(^\wedge\) Predicted value or function

\(\sim\) Approximate function
ACKNOWLEDGMENTS

I thank my advisor Dr. John Renaud for his insightful guidance, encouragement, friendship, and for the summer research opportunity that lead to my graduate studies at Notre Dame. I would also like to thank the rest of my committee members: Dr. Hafiz Atassi, Dr. Stephen Batill, Dr. Michael Stanišić, and Dr. Aaron Striegel; each of whom have taken the time to personally share with me their knowledge and passion for learning.

I would also like to thank the following people for their help with various aspects of my research: Dr. Jay Martin (ARL Penn State), Dr. Tim Simpson (Penn State), Dr. Eric Nielsen (NASA Langley - FUN3D/2D), and Dr. Mike Eldred (Sandia National Labs). I also gratefully acknowledge funding provided by: Dr. Kam Ng and the Office of Naval Research, Dr. Brian Sanders (AFRL), Dr. Terry Weishhaar (DARPA), and the National Science Foundation.

To all of my friends and colleagues of the Design Automation Laboratory at Notre Dame, thank you for all the insightful discussions and great times: Xiaoyu (Stacey) Gu, Victor Pérez, Dhanesh Padmanabhan, Weiyu Liu, Parveen Chandila, Andrés Tovar (co-inventor of lab-ball), Harish Agarwal, Alejandro Espinoza, Neal Patel, and Charlie Penninger.

Rebekah, thank-you for your patience amid this endeavor and for sharing with me your grammatical expertise.

Finally, I would like to thank my parents for providing continual support and for teaching me, each in their own unique way, to believe in myself.
The results of optimization are all around us. They can be seen most abundantly in nature, which Darwin meticulously studied in his book entitled *On the origin of species by means of natural selection, or preservation of favoured races in the struggle for life*[^12]. Optimization can been seen in our normal day-to-day lives in the form of consumer products, which are designed to maximize profits in competitive marketplaces. Stock market investors use optimization as they strive to increase yields while avoiding unnecessary risk. Airline schedules and crew coordination are all optimization problems which are solved to save millions, if not billions, of dollars annually[^61] [^110]. The notion of optimization is not new to human decision making; in fact, it has been used for many millennia. Around 45 BC, Cicero used the phrase *minima de malis* — the least of evils — in *De Officiis* as a guide to minimize conflict. However, it wasn’t until the 1940s that optimization became a feasible tool to solve large, complex problems; this corresponds directly to the advent of the digital computer. This active research topic has been evolving to solve ever larger and more complex problems since.

The study of optimization is a union of both rigor and heuristics which rely on theory as well as experimentation. Many modern fields, from engineering and science to economics and business, use optimization on a routine basis. It can
also be studied from a pure mathematical point of view. Because of its wide use, optimization methods need to be studied carefully and improved upon so that they become more efficient and more robust, enabling them to solve a larger range of problems.

The focus of this work is optimization from the engineering paradigm; though it could also be applied to other fields. Engineering problems typically have a nonlinear design space and have many constraints as well as competing objectives. The problems usually involve the design of some product, whether it be a single part or a whole system of interdependent parts. These types of problems usually make use of physical computer simulations such as finite element analysis (FEA) to determine structural characteristics or computational fluid dynamics (CFD) to determine fluid flow interactions. Using these types of analyses in the optimization process is referred to as simulation-based design optimization and can be computationally expensive to perform.

1.1 Why Optimize?

An answer to this question may be: to use scarce resources effectively. Or to quench the human desire to be the best. It could also be to increase profits or any number of other reasons. In many industrial settings, however, the designs are produced by so called \textit{experts} who have designed similar products in the past. Optimization allows for the searching of a design space that may lead to novel designs that the expert might not think of because he or she is so influenced by a previous design. Design automation using optimization does not replace human
designers; it is a tool which they can use to produce better designs quicker. The
designer still has to choose the appropriate optimization method, pose the correct
optimization problem, use the appropriate models to evaluate the objective and
constraints, and validate the designs obtained from the optimization process.

Optimization also provides a consistent methodology in which best designs can
be selected in an efficient and less time consuming way. Consider the following
scenario presented by Vanderplaats. Given a design problem that can be
described using 3 design variables, we wish to investigate the designs for 10 values
of each variable. Let’s also assume that it takes a tenth of a second to analyze
any design on a digital computer. This gives 1000 designs to be evaluated and
compared requiring 100 seconds to obtain the optimal solution. This amount of
time wouldn’t normally cause a problem. However, now consider a more realistic
design problem that takes 10 design variables to describe a single design. Again,
investigating 10 values for each variable results in $10^{10}$ possible design. Assuming
now that the computational cost per design evaluation is 10 seconds, the total
time required to find the optimal solution is $10^{11}$ seconds or over 3168 years! This
is obviously not a practical approach to solve the problem and a more efficient
approach to design automation is needed — and is provided by optimization
methods.

1.2 Current Limitations of Using Design Optimization

Optimization methods can be a great tool for designers and can increase produc-
tivity, but they do have their limitations. A few of the most significant problems

3
that challenge current state-of-the-art algorithms are discussed here. Problems
that involve design variables that are not continuous, known as combinatorial
problems or mixed integer problems, are difficult because of the sheer number
of combinations possible; this is the type of problem demonstrated in the
previous section. Branch and bound techniques can provably solve these types
of problems but still require overwhelming numbers design trials. Other
heuristic based approaches such as genetic algorithms (GA), particle
swarming, and simulated annealing have been the most successful
approaches in dealing with these types of problems. Global optimization on
multi-modal functions is a difficult problem because most optimization schemes
rely on local convergence criteria, and it is generally impossible to ever prove a
solution is the global one. Another issue is the size of the design problem.
Current large scale commercial optimizers can solve problems with 250,000 design
variables and topology optimization problems with over two million variables.
The size of the design space is not always the limiting factor; if the design evalu-
ations themselves are computationally expensive, then the problem of optimizing
the design can be prohibitively time consuming. Addressing this problem using
variable fidelity methods, which are introduced in the next section, is the main
goal of this dissertation.

Other issues that must be addressed before a level of trust can be placed in an
optimizer’s solution exist mainly in engineering applications where simulations
are abundant. The designer has to have a good feel for the uncertainty between
simulation and true physics, be aware of noisy functional output that may result
in inaccurate trend or gradient information, and must deal with regions in the
design space where the simulations may fail.

1.3 Variable Fidelity Optimization

Variable fidelity and other model management methods have been developed to solve optimization problems that involve simulations with large computational expense. However, these methods can be used to optimize any problem where various fidelity models exist or can be constructed. In this context, the level of fidelity refers to the amount of physics or types of assumptions used in the model. A higher fidelity model is one that contains physics or details that do not exist or are not accounted for in a lower fidelity model. Based on this definition, the real system as it occurs in nature has infinite fidelity with an infinite number of input variables.

A major constraint to the design process in engineering is cost. This cost is directly proportional to computer resources and the time required to run the optimization of simulations to predict system performance. Even though computers have increased in computational power at an exponential rate, Venkataraman and Haftka\cite{VenkataramanHaftka} pointed out that model complexity and fidelity have increased, maintaining approximately the same execution time.

Many times in the design process, various levels of model fidelity are developed, from low order models for initial conceptual design to the full-physics based models used for the final optimization analysis. The lower fidelity models are typically much cheaper to evaluate, but designs produced using these models
neglect important physical effects included in the more expensive higher fidelity models. The framework presented here attempts to build a scaling function that matches the result of the lower fidelity model to the higher fidelity model. The bulk of the optimizer’s required function calls will be to the lower fidelity model which are updated by the scaling function. The methodology provably converges to the solution of the more expensive models with substantially less calls than would be required if optimization were done solely using the higher fidelity model. The proof of convergence relies on, at least, a first order matching between the models, and the use of a trust region management scheme [40].

Early works that used variable fidelity models for optimization were largely based on heuristics, and the processes were not guaranteed to converge to the high fidelity solution. Consequently, some cases converged to the low fidelity solution [27, 64]. After realizing that significant differences could arise in the fidelity of models, new methods were developed which were proven to converge to the high fidelity solution. Some of these methods used gradient information [10] and others did not [45]. Haftka [63] devised a multiplicative scaling factor to update the value of lower fidelity models to match the higher fidelity models. Chang et al. [33] incorporated sensitivity information into the scaling factor. Alexandrov [13, 15, 16] combined this scaling approach and fundamental trust-region methods [40] into an approximation management framework (AMF). The combination of satisfying the first order consistency condition along with other theoretical assumptions produced a provably convergent methodology. The AMF was demonstrated using various existing optimization routines such as the Augmented Lagrangian Method or Sequential Quadratic Programming. Giunta and Eldred implemented a similar
trust region based method into the DAKOTA project at the Sandia National Laboratory [58]. A pattern search based framework was also developed by Booker et al. [25].

Response surface based methods that use variable fidelity data have been studied extensively by Rodríguez et al. [129, 130, 131]. In this work, coupled multidisciplinary design optimization problems were solved. Response surfaces were constructed using the fully coupled solutions and decoupled approximate solutions generated from the use of the global sensitivity equations (GSE) developed by Sobieszczanski-Sobieski [142]. The response surfaces were sequentially optimized using the Augmented Lagrangian method, and a trust region model management framework was used to prove convergence [128].

Successive approximate optimization (SAO) algorithms generate response surfaces with a similar motivation as variable fidelity methods, namely to conduct optimization on an inexpensive surrogate model. SAO methods have been successful in reducing computational cost, especially when the sampling needed to create the response surface is on the order of the number of design variables [118, 119]. As this research shows, variable fidelity methods can further reduce computational cost by using a function similar to a response surface, which is based more on physics — a low fidelity model updated with a scaling function. The scaling function is essentially a response surface of the error between any two fidelity models [86]. *The majority of the work in this investigation is concerned with how this model is constructed and managed.*
Variable fidelity methods require only one high fidelity function and gradient evaluation per iteration, in comparison to typical SAO methods that use a sampling of high fidelity calls to produce a low fidelity surrogate. Variable fidelity methods have another advantage over other adaptive response surface methods that use only high fidelity samples: the low fidelity models provide a global approximation which is locally corrected using scaling functions to produce a better overall approximation of the high fidelity function.

Previous work dealing with managing the use of various fidelity models have used Taylor series based scaling functions to match the models. In this work, a novel method for matching the various levels of fidelity to further decrease the computational expense of the design process using kriging models is developed and evaluated. A few motivations for using a kriging based model, as opposed to a Taylor series or polynomial based approach, stem from the fact that these other methods use only local information to scale on a global level. The kriging approach allows past information to be incorporated into the scaling function. Furthermore, polynomial scaling functions tend to diverge as the distance from the current design increases, whereas kriging models are typically bounded. Additionally, a kriging model can easily incorporate any previous results obtained from analyses completed using the various fidelity models prior to the optimization process.

The variable fidelity optimization problem can be written in its most general form using the standard nonlinear programming problem:
\[
\begin{align*}
\text{minimize} & \quad f(x) \\
\text{subject to:} & \quad g(x) \leq 0 \\
& \quad g_u(x) \leq 0 \\
& \quad h(x) = 0 \\
& \quad h_u(x) = 0 \\
& \quad l \leq x \leq u,
\end{align*}
\]

where \( f \) is the objective function, \( x \) are the design variables, \( g \) is the vector of inequality constraints, \( h \) is the vector of equality constraints, the subscript \( u \) means the constraint is unscaled, and \( l \) and \( u \) are the upper and lower bounds respectively of the design variables. In the general case, unscaled constraints are included in the optimization and are unaffected by the various fidelity models. An example of such constraints is a constraint solely based on the design variables (i.e. geometry of the design). Scaling these constraints may unnecessarily slow convergence. Furthermore, the objective and the constraints can be evaluated using various fidelity models and the solution is desired of the highest fidelity model.

1.4 Summary of Original Research Contributions

This dissertation investigates and extends current state-of-the-art variable fidelity model management algorithms. The new contributions reduce the total computational time needed to optimize complex high fidelity computer simulations by using low fidelity models and novel scaling techniques. The following list gives a brief summary of the most important original research contributions of this work.

- Development of an adaptive, hybrid, multiplicative-additive, scaling method.
Some design problems are solved faster using an additive scaling function and some with the multiplicative scaling function. It is typically not known \textit{a priori} which model will be the most efficient. This method adaptively combines the two methods, resulting in a more robust implementation and computational savings, since no time is lost determining which method would be more efficient.

- \textit{Development of second order and quasi-second order scaling methods.} This method is an extension of the currently used first order methods that also includes second order terms. The second order information hasn’t been used in the past due to the computational expense required — $O(m^2)$ high fidelity function calls for finite differencing, where $m$ is the number of design variables. The second order terms are instead approximated using cumulative first order information, reducing the number of expensive high fidelity function calls needed to $O(m)$ for finite differencing, which is the same number needed in the first order methods. This means the method essentially obtains a super-linear convergence rate using no extra high fidelity calls.

- \textit{Development of a scaling technique using kriging.} This method uses a global approach for scaling the low fidelity model to match the high fidelity response. This idea contrasts the Taylor series based methods which use only local information to scale the low fidelity model globally. The new technique stems from the fact that the low fidelity model is a global approximation to the high fidelity model, so a global scaling method should be more beneficial than a local one. This method reduces the high fidelity
function calls required for convergence by allowing the optimizer to take larger steps because the scaled low fidelity model will give a better approximation to the high fidelity model over a larger range. This method also allows for incorporating previous data obtained from running either of the low or high fidelity models prior to the optimization, which increases the approximations quality, promoting faster convergence.

- Development of a warm started kriging based scaling method. Because the kriging model needs to accumulate a few sample points before it is able to provide a good global approximation, a warm start is proposed. The optimization would begin by using a Taylor series based method while building up the kriging model. Then it would switch over to the kriging model to accelerate convergence without the penalty of a slow convergence rate at the beginning of the optimization.

- Development of a metamodel update management strategy. In problems with large numbers of design variables, the cost of refitting the kriging model during each variable fidelity optimization iteration can become computationally burdensome. Many metamodels, including kriging, require a set of model parameters to be re-optimized, or refit, in order to produce a good approximation of the original model from a limited set of data points. However, for any given set of model parameters the resulting metamodel is still interpolative, albeit a potentially poor fit of the original model. Between iterations of the variable fidelity optimization process only a few sample points are added to the kriging metamodel; hence, the model parameters may not need to be updated each iteration. A metamodel update management strategy
(MUMS) is developed to gauge when the parameters need to be refit, reducing the most computationally expensive aspect of using kriging metamodels sequentially. Two different MUMS are developed, one using a likelihood ratio (L-MUMS) and the other a trust region ratio (TR-MUMS).

- Development of a reliability based design optimization approach using variable fidelity. Reliability based design seeks to find designs that optimize an objective while also insuring that the final design is less prone to failure due to uncertainties in design parameters such as manufacturing tolerances, material properties, or unexpected loadings. However, this optimization process tends to be quite computationally expensive. An integrated framework that reduces the cost of reliability based design using variable fidelity methods is developed.

1.5 Overview of the Dissertation

In the remainder of this dissertation the aforementioned methods are derived, implemented, and compared to existing methods. Test problems are used to demonstrate the methods ability to reduce the computational cost of performing high fidelity optimization. In Chapter 2 the variable fidelity algorithm is explained in detail along with a discussion of the additive, multiplicative, and adaptive hybrid scaling function. The Taylor series based methods of approximating the scaling functions are developed in Chapter 3. The first order methods are presented first and are the methods currently used in variable fidelity methods. The second order methods, including the quasi-second order methods are then derived. Next, all of the Taylor series based methods are compared using two
demonstration problems. The kriging based scaling approximation is introduced in Chapter 4 along with the warm start variant. The kriging and warm start kriging methods are compared to the Taylor series based methods using the same design test problems. Methods for reducing the cost of sequentially building kriging models in variable fidelity optimization are developed, implemented, and tested in Chapter 5. Chapter 6 covers the application of variable fidelity methods to reliability based design optimization, allowing such problems to be solved in less time. Final conclusions and recommendations for future research are presented in Chapter 7.
CHAPTER 2

THE VARIABLE FIDELITY OPTIMIZATION FRAMEWORK

Variable fidelity optimization, or model management methods, were developed for one main purpose: to reduce the computational expense of design optimization for complex high fidelity simulations that require large amounts of computer resources for a single evaluation. These methods rely on a suite of fidelity models while using the fewest number of high fidelity function calls possible. The low fidelity models can take on many forms; for instance, they can be surrogate models or low order physics models. In this chapter a detailed description of the variable fidelity framework is given, starting with a general overview of sequential approximate optimization methods.

2.1 Sequential Approximate Optimization Using Response Surfaces

Sequential approximation optimization (SAO) is a class of methods for the design of systems which are cost prohibitive to compute large numbers of function evaluations. They are especially effective for use in complex systems consisting of many disciplines or multidisciplinary design optimization (MDO). In typical SAO methods a local response surface approximation (RSA) is built from a sampling of the design space about the current design. Once the RSA has been
constructed it is very inexpensive to evaluate and its optimal point can be found using traditional constrained nonlinear optimizers. The optimal point of the RSA is compared to the true model. If the design has improved, the process repeats, sequentially building approximations, until the method converges. There are extensive examples of SAO methods in the literature including work by: Rodríguez et al. [128, 130], Pérez et al. [118, 119], Romero et al. [132], and Rasmussen [125].

There are many types of response surface approximations used. Insightful reviews of various types of methods have been done by Chung and Alonso [36] and Simpson et al. [141]. The most widely used methods are low order polynomial regression models [26, 109]. Kriging models have been gaining popularity [59, 101, 139]. Other models that have been used for RSA include radial basis functions [30] and neural networks [94, 145].

An important aspect of constructing the response surface approximations is how the design space is sampled. The sampling patterns, also known as experimental design arrays, come in many varieties, each with their own benefits and associated costs. Traditional design arrays such as the full factorial experiments (FF) or central composite designs (CCD) require $O(2^m)$ number of samples [109], where $m$ is the number of design variables. Other sampling strategies such as Latin hypercubes [105, 146] and orthogonal arrays [68, 69, 113] require $O(m^2)$ samples without losing the space filling properties. This is a significant reduction in the number of samples needed for high dimension problems; however, these methods still suffer from the curse of dimensionality. An adaptive experimental design for construction of second order response surface approximations was developed by
Pérez et al.

that required samples only on the order of the dimension of the design space, \( O(m) \).

The variable fidelity algorithm, presented in detail next, is a SAO type framework that uses a response surface built from a low fidelity model and a scaling function. The scaling function is sequentially built from only one high and one low fidelity function call and a gradient evaluation of each model. This is in contrast to doing the traditional sampling of other SAO methods which can be quite computationally expensive in terms of the number of high fidelity function calls required. Another advantage to using this type of response surface is that it uses physical information from a lower fidelity model instead of assuming some underlying mathematical properties of the design space such as linear, quadratic, or Gaussian. The low fidelity model is scaled to match the high fidelity model to include physical phenomena not present in the low fidelity model; this improves convergence rates and helps the overall process converge to the high fidelity optima.

2.2 The Variable Fidelity Optimization Algorithm

The typical framework for variable fidelity optimization is depicted in Figure 2.1 and is based, in part, on work done by Alexandrov and is described by Gano et al. This framework is designed to reduce the number of high fidelity function calls during the optimization process by using a scaling function and lower fidelity models. The following describes the basic steps of the framework:
**Step 1 Initialization:** At the starting design point, $x_0$, the objective function is evaluated using both $f_{\text{high}}(x_0)$ and $f_{\text{low}}(x_0)$, the high and low fidelity models respectively. All of the high and low fidelity inequality and equality constraints $g_{\text{high}}(x_0)$, $g_{\text{low}}(x_0)$, $h_{\text{high}}(x_0)$, and $h_{\text{low}}(x_0)$ are also evaluated. The initialization step is concluded by evaluating the $l_1$ penalty function using Equation (2.4).

**Step 2 Gradient Evaluation:** The gradient of the objective for both the high and low fidelity models $\nabla f_{\text{high}}(x_n)$, $\nabla f_{\text{low}}(x_n)$ are evaluated at the current design point, $x_n$. Also the Jacobian for the constraints are evaluated using both the high and low fidelity models: $\nabla g_{\text{high}}(x_n)$, $\nabla g_{\text{low}}(x_n)$, $\nabla h_{\text{high}}(x_n)$, and $\nabla h_{\text{low}}(x_n)$.

**Step 3 Construct Scaling Model:** A scaling model is constructed to insure matching between the different fidelity models. This model can be based on many different methods; additive and multiplicative are the most common and
are discussed in more detail later. Each method can be modeled as first order, second order, or kriging based and are described in detail in Chapters 3 and 4. A scaling model is constructed for each constraint as well as for the objective function. The scaling models \( f_{\text{scaled}}(x) \), \( g_{\text{scaled}}(x) \), and \( h_{\text{scaled}}(x) \) all depend on zero and at least first order information, as well as the low fidelity model. This dependence is summarized in Equations (2.1)-(2.3).

\[
\begin{align*}
    f_{\text{scaled}}(x) &= f(f_{\text{high}}(x_n), f_{\text{low}}(x_n), \nabla f_{\text{high}}(x_n), \nabla f_{\text{low}}(x_n), f_{\text{low}}(x)) \quad (2.1) \\
    g_{\text{scaled}}(x) &= g(g_{\text{high}}(x_n), g_{\text{low}}(x_n), \nabla g_{\text{high}}(x_n), \nabla g_{\text{low}}(x_n), g_{\text{low}}(x)) \quad (2.2) \\
    h_{\text{scaled}}(x) &= h(h_{\text{high}}(x_n), h_{\text{low}}(x_n), \nabla h_{\text{high}}(x_n), \nabla h_{\text{low}}(x_n), h_{\text{low}}(x)) \quad (2.3)
\end{align*}
\]

**Step 4 Optimize Scaled Low Fidelity Model:** The low fidelity model scaled with the scaling model constructed in Step 3 is optimized. The optimization problem solved in this step is:

\[
\begin{align*}
    \text{minimize} & \quad f_{\text{scaled}}(x) \\
    \text{subject to:} & \quad g_{\text{scaled}}(x) \leq 0 \\
                        & \quad g_n(x) \leq 0 \\
                        & \quad h_{\text{scaled}}(x) = 0 \\
                        & \quad h_n(x) = 0 \\
                        & \quad ||x - x_n||_\infty < \Delta_n \\
                        & \quad 1 \leq x \leq u.
\end{align*}
\]

The choice of optimizer used is based on preference. In the work done by Alexandrov[15], three optimizers were compared: augmented Lagrangian method, multilevel algorithms for large-scale constrained optimization (MAESTRO)[16].
(used for coupled MDO problems), and sequential quadratic programming (SQP).

For typical single discipline problems, Alexandrov found SQP to be the most promising, and it is used in this research. The unscaled constraints are included in this step to ensure that they are always satisfied.

**Step 5 Evaluate New Design and $l_1$ Penalty Function:** A new candidate point, $x^*_n$, is found as a result of the solving the optimization problem in Step 4. At this new candidate point the high fidelity objective, $f_{\text{high}}(x^*_n)$, and constraints, $g_{\text{high}}(x^*_n)$ and $h_{\text{high}}(x^*_n)$, are evaluated. The objective and constraint values are used to calculate a current value of the $l_1$ penalty function, $P$, for the high and scaled low fidelity models. The penalty function is defined as

$$ P(x) = f(x) + \frac{1}{\mu_n} \sum \max(0, g_i(x)) + \frac{1}{\mu_n} \sum |h_j(x)|, \quad (2.4) $$

where $\mu$ is the penalty weight which is typically decreased by a factor of ten each time a new point is accepted. This penalty weighting drives all the active constraints to zero as the algorithm converges.

**Step 6 Trust Region Management:** In order to help guarantee convergence of the variable fidelity optimization framework, a trust region model management strategy is employed\[39, 107, 131]. This method provides a means for adaptively managing the allowable move limits for the approximate design space. Originally, these methods were used to ensure the convergence of Newton based methods, but have since found broader application.

A trust region ratio allows the trust region model management framework to
monitor how well the approximation matches the high fidelity design space. The
trust region ratio, $\rho_n$, is calculated at the new candidate point $x^*_n$:

$$
\rho_n = \frac{P(x^*_n)_{\text{high}} - P(x^*_n)_{\text{high}}}{P(x^*_n)_{\text{scaled}} - P(x^*_n)_{\text{scaled}}},
$$

(2.5)

where $P()_{\text{high}}$ and $P()_{\text{scaled}}$ are the $l_1$ penalty functions for the high and scaled low
fidelity models and the point $x_n$ was the initial point of the optimization. Notice
that by definition $P(x^*_n)_{\text{scaled}} = P(x^*_n)_{\text{high}}$ because the scaled low fidelity model
matches the high fidelity model at that point. The trust region ratio is the ratio
of the actual change in the function to the predicted change of the function by the
scaled lower fidelity model. Because the constraints are also approximated, the
trust region ratio must account for this and converge to a feasible design, which
is the reasoning behind using the $l_1$ penalty function.

The trust region size is governed by the following standard rules\cite{58,128}:

$$
\Delta_{n+1} = \begin{cases} 
    c_1 \Delta_n & : \rho_n \leq R_1 \lor \rho_n > R_3 \\
    \Delta_n & : \ R_1 < \rho_n < R_2 \\
    \Gamma \Delta_n & : \ R_2 \leq \rho_n \leq R_3 
\end{cases}
$$

(2.6)

where $\Gamma = c_2$ if $\|x_k^* - x_{ck}\|_\infty = \Delta_k$ otherwise $\Gamma = 1$. A typical set of values for
the range limiting constants are $R_1 = 0.25$, $R_2 = 0.75$, and $R_3 = 1.25$, while the
trust region multiplication factors are typically $c_1 = 0.25$ and $c_2 = 3$. Physically,
$\rho$ represents how good of an approximation our scaled low fidelity model is com-
pared to the high fidelity model. If $\rho$ is near 1, the approximation is quite good.
If $\rho$ is near zero, then the approximation is not as good, but it still captures the
minimization trend. If \( \rho \) is negative, then the point is a worse design. In this case the point is rejected, the trust region size is reduced by the factor \( c_1 \), and the algorithm returns to Step 4. As long as \( \rho > 0 \), the point is accepted and the algorithm proceeds to Step 7.

A method to avoid the somewhat arbitrary values for \( R_1, R_2, R_3, c_1, \) and \( c_2 \) was developed by Wujek and Renaud\cite{154,155} called the trust region ratio approximation method (TRAM). TRAM provides a continuous updating of the trust region move limits using available gradient information. This approach has not been implemented in this research but may be of interest for future studies.

**Step 7 Convergence Test:** For the implementation used in this research, convergence was determined by the following stopping criterion:

\[
\begin{align*}
    f_{\text{high}}(x_n) - f_{\text{high}}(x_{n-1}) &< \epsilon_f, \\
    \|x_n - x_{n-1}\| &< \epsilon_x,
\end{align*}
\]

where \( \epsilon_f \) and \( \epsilon_x \) are tolerances supplied by the user, and \( n \) is the current iteration counter. If any of the two inequalities at the current point is true, the algorithm is considered converged. If the convergence test is true, then the final design is found, otherwise, the algorithm returns to Step 2.

Convergence to a Karush-Kuhn-Tucker (KKT) point can tested at the final design point by evaluating the projected gradient as is done by Rodríguez et al.\cite{129}.
If the projected gradient at the final design is zero then the design satisfies the KKT conditions.

2.3 Requirements for Proof of Convergence

In order to practically use the variable fidelity framework, the issue of convergence must be addressed. While proof of convergence may not be necessary for a practicing engineer whose main goal is an improved design, many researchers have had a good deal of success in using provably convergent methods that use trust region management methods[10, 46, 130]. To prove convergence for such a method, three conditions are required: use of a trust region management strategy, at least first order matching between the scaled low fidelity function and the high fidelity model[25], and an inner optimization scheme that is provably convergent. The trust region management insures improvement of the design and, in this framework, means that in the worst case scenario, the framework degenerates to running the optimization on the high fidelity model alone with a step size restriction. Furthermore, first order consistency guarantees that an improved design of the high fidelity model can be achieved by a sufficiently small step, \( \epsilon \), using the scaled low fidelity model[14]. This is not insured with only zero order matching. The zero order matching conditions are:

\[
\lim_{\epsilon \to 0} f_{\text{scaled}}(x + \epsilon) = f_{\text{high}}(x), \quad (2.9)
\]

\[
\lim_{\epsilon \to 0} g_{\text{scaled}}(x + \epsilon) = g_{\text{high}}(x), \quad (2.10)
\]

\[
\lim_{\epsilon \to 0} h_{\text{scaled}}(x + \epsilon) = h_{\text{high}}(x). \quad (2.11)
\]
The first order conditions match the local trend and are mathematically stated as

\[
\lim_{\epsilon \to 0} \nabla f_{\text{scaled}}(x + \epsilon) = \nabla f_{\text{high}}(x),
\]  \hspace{1cm} (2.12)

\[
\lim_{\epsilon \to 0} \nabla g_{\text{scaled}}(x + \epsilon) = \nabla g_{\text{high}}(x),
\]  \hspace{1cm} (2.13)

\[
\lim_{\epsilon \to 0} \nabla h_{\text{scaled}}(x + \epsilon) = \nabla h_{\text{high}}(x).
\]  \hspace{1cm} (2.14)

2.4 Inner Optimizer Selection

The choice of the inner optimizer used in Step 4 of the variable fidelity framework relies on the type of problem to be solved. For general problems with smooth and continuous design spaces, sequential quadratic programming (SQP) has been shown to be the most effective\cite{15}. For multidisciplinary problems that require a consistent design among many disciplines, methods such as those presented by Balling and Sobieszczanski-Sobieski\cite{20] should be used; this will reduce the cost by decoupling the disciplines and enforcing consistency only at convergence. Design problems such as those that include discrete variables, discontinuous design spaces, simulations that may fail over part of the design space, or simulations where gradient information is unattainable can use zero order scaling and pattern search methods as seen in the work by Dolan \textit{et al.}\cite{48] and Lewis \textit{et al.}\cite{88]. These methods cannot be guaranteed to converge to the high fidelity optima without modification to the variable fidelity framework, so they are more expensive.

The SQP optimizer used in this research was Mathworks MATLAB’s \textit{fmincon} which is provided in the Optimization Toolbox. In this method, a quadratic programming (QP) subproblem is solved at each iteration. An estimate of the
Hessian of the Lagrangian is updated at each iteration using the BFGS formula. A line search is performed using a merit function similar to that proposed by Han\textsuperscript{67} and Powell\textsuperscript{123, 124}. The QP subproblem is solved using an active set strategy similar to that described in Gill \textit{et al.}\textsuperscript{57}. Another commercial state of the art SQP optimizer which should be considered for practical implementation of the variable fidelity framework is SNOPT which was developed by Stanford Business Software Inc\textsuperscript{52, 56}.

2.5 Scaling Methods Versus Space Mapping

In Step 3 of the variable fidelity algorithm a scaling function is constructed so that when it is combined with the lower fidelity model it approximates the high fidelity model. Another approach to approximating the high fidelity model is to transform the design space of the low fidelity model by space mapping (SM) techniques. This section discusses the main differences between these approaches, along with a justification for choosing the scaling based approach for variable fidelity optimization. For a more detailed comparison refer to the work done by Alexandrov and Lewis for first order scaling methods\textsuperscript{14}.

The scaling methods produce a transformation function, as seen in Equations (2.1)-(2.3), that can transform the low fidelity model to match the high fidelity model. In contrast SM techniques transform the design space itself and, therefore, do not produce a function or model. The space mapping transformation, \( \phi(x; p) \), uses a set of parameters, \( p \), to modify the design space of the low fidelity model to approximate the high fidelity model. These parameters are chosen by solving
a least squares minimization problem for a given set of designs, $x_i$, in which the high fidelity model has been evaluated. The parameters $p$ are chosen such that

$$p = \arg \min_p \sum_{i=1}^{N} ||f_{\text{high}}(x_i) - f_{\text{low}}(\phi(x_i; p))||^2_2.$$  \hfill (2.15)

For a more complete description of SM see a review of its application to engineering optimization and modeling by Bakr et al.\cite{Bakr2008}. SM methods cannot typically match the local gradient information. Therefore, using them in a variable fidelity model management scheme cannot be guaranteed to convergence to the high fidelity solution. Space mapping techniques work well when the low fidelity model is a good representation of the high fidelity model, since the local trends will be similar for both models. Scaling models, on the other hand, can be constructed to insure local trends of the high fidelity model are captured when scaling the low fidelity model. The scaling methods were selected because they can provide first order consistency, which allows for a provably convergent algorithm.

2.6 Scaling Methods

Variable fidelity or approximate model management frameworks can be derived from an infinite number of scaling functions, however, existing methods have generally used two varieties: multiplicative and additive. Currently, the most common is the multiplicative framework, devised by Alexandrov and Lewis\cite{Alexandrov2000} based on Haftka’s\cite{Haftka2000} scaling function. The additive method was presented by Lewis and Nash\cite{Lewis1999}. Both methods are based on constructing an unknown function to update the lower fidelity model, which in turn, will approximate the higher fidelity model.
2.6.1 Multiplicative Scaling

A given set of high and low fidelity models, \( f_{\text{high}}(x) \) and \( f_{\text{low}}(x) \), can be matched by multiplying the low fidelity model by an unknown function \( \beta(x) \). This is posed mathematically as

\[
f_{\text{high}}(x) = \beta(x) f_{\text{low}}(x). \tag{2.16}
\]

This scaling model was first proposed and used for approximating structural response by Haftka et al.\cite{63}. Solving for the unknown multiplicative scaling function results in

\[
\beta(x) = \frac{f_{\text{high}}(x)}{f_{\text{low}}(x)}. \tag{2.17}
\]

From inspection of Equation (2.17), it is shown that the function \( \beta(x) \) is the scaling ratio of the high fidelity model to the low fidelity model, and when it is multiplied by the low fidelity model, the high fidelity model is achieved\cite{63}. However, the exact scaling function \( \beta(x) \) is not known and must be approximated. Different methods to approximate the scaling function are studied and demonstrated in Chapters 3 and 4.

In this investigation, the multiplicative scaling methodology, as introduced, was found to have a divide-by-zero problem under certain conditions. This issue was not identified by Alexandrov\cite{12, 15, 16} or Chang\cite{33}. The issue arises when either the constraint or objective values of the low fidelity model approach zero.
This causes a problem because the low fidelity value is in the denominator of Equation 3.3 which defines the multiplicative scaling function. The divide by zero problem can occur as constraints become active or the objective nears zero. To solve this problem, a simple extra offset factor is proposed. This is obtained by adding a constant term, \( C \), to the low fidelity objective and/or constraint values. When this factor is added to low fidelity constraints and/or objective, it prevents the divide-by-zero problem. A strategy for choosing this factor and investigating its effect on the sensitivity of the method is still to be completed. The scaling function \( \beta \) does automatically adjust for this added scalar and, therefore, doesn’t introduce any further complications as long as the offset value is reasonable in size.

2.6.2 Additive Scaling

A given set of high and low fidelity models, \( f_{\text{high}}(x) \) and \( f_{\text{low}}(x) \), can also be matched by adding the low fidelity model to an unknown function \( \gamma(x) \). This is expressed mathematically as

\[
 f_{\text{high}}(x) = f_{\text{low}}(x) + \gamma(x). \tag{2.18}
\]

The additive scaling function can be solved for by subtracting the low fidelity function from both sides:

\[
 \gamma(x) = f_{\text{high}}(x) - f_{\text{low}}(x). \tag{2.19}
\]

From Equation 2.19 it is clear that the function \( \gamma(x) \) is the additive scaling of the high fidelity model to the low fidelity model, or the error between them. When
this function is added to the low fidelity model, the response of the high fidelity model is produced. A similar function for the constraints can be developed in the same manner as Equations (2.18) and (2.19). Again, the exact scaling function \( \gamma(x) \) is not known and must be approximated. Methods to approximate the additive scaling function are also studied and demonstrated in Chapters 3 and 4.

2.7 Adaptive Hybrid Scaling - Combining Additive and Multiplicative Methods

In this investigation a new adaptive hybrid scaling method is developed. Previous studies have used either a multiplicative or additive scaling function to update the low fidelity response to approximate the high fidelity model. In general, some sets of fidelity models are matched better using one method or the other, and there is no way to know this \textit{a priori}. This section presents a novel methodology that could be more robust by including both types of scaling.

Combining both the multiplicative and additive scaling functions such that they still properly scale the low fidelity model to match the high fidelity model requires the use of a weighted average of the two methods. Using a weighting term, \( \mathcal{W} \), this sum is

\[
f_{\text{high}}(x) \approx \mathcal{W} f_{\text{low}}(x) \tilde{\beta}(x) + (1 - \mathcal{W}) (f_{\text{low}}(x) + \tilde{\gamma}(x)). \tag{2.20}
\]

The multiplicative and additive scaling approximations, \( \tilde{\beta}(x) \) and \( \tilde{\gamma}(x) \), are constructed to match the low and high fidelity models using zero and first order information at the current design point so the only unknown is the weighting
term $\mathcal{W}$. To determine the value of $\mathcal{W}$, a further condition must be enforced. Concurrent work being done by Eldred et al.\cite{Eldred} proposes to use a previously evaluated point, $x_{pp}$, to adjust the value of $\mathcal{W}$ such that the model passes through that point as well. Enforcing this conditions gives the relation

$$f_{\text{high}}(x_{pp}) = \mathcal{W}f_{\text{low}}(x_{pp})\beta(x_{pp}) + (1 - \mathcal{W})(f_{\text{low}}(x_{pp}) + \gamma(x_{pp})). \quad (2.21)$$

After solving for $\mathcal{W}$, the weighting function then takes the value

$$\mathcal{W} = \frac{f_{\text{high}}(x_{pp}) - (f_{\text{low}}(x_{pp}) + \gamma(x_{pp}))}{f_{\text{low}}(x_{pp})\beta(x_{pp}) - (f_{\text{low}}(x_{pp}) + \gamma(x_{pp}))}. \quad (2.22)$$

Equation (2.22) uses the current additive and multiplicative scaling functions along with any previous point, $x_{pp}$, where the high fidelity model was evaluated. There is some freedom in choosing the past point. One option is to simply use the last accepted design point. However, for this work the nearest point is used. The advantage of using the nearest point is that it could have been a design that was evaluated but rejected; this would help keep the next iteration from moving in this undesired direction. A new weighting value can be computed both for the objective and each constraint at each iteration. Updating these weights at each iteration allows the framework to adapt to the best model for the current area of the design space.

The approximated scaling functions, $\tilde{\beta}(x)$ and $\tilde{\gamma}(x)$, used in the hybrid method can be of any of the forms developed in this research: for example, first order, second order, or kriging models. The averaging formulation of the adaptive hybrid method maintains the local consistency conditions of the different approximations, and therefore, retains their provably convergent properties.
The adaptive hybrid method has more overhead in memory usage and CPU time in computing both scaling functions; though, this extra computational cost is typically negligible compared to a high fidelity function call. The extra expense of this method is offset substantially if just one high fidelity call is saved.

2.8 Summary

A variable fidelity optimization framework was outlined and described in Section 2.2 Variable fidelity methods are a subset of a class of algorithms referred to as sequential approximate optimization methods. A main distinguishing characteristic of the variable fidelity method is that it only requires one high fidelity function call and, if possible, a gradient evaluation per iteration. A response surface is constructed to match a low fidelity model to match the local trend of the high fidelity model. This matching is done using multiplicative scaling functions, additive scaling functions, or an adaptive hybrid method that combines the two scalings into a single function. The variable fidelity framework can be proved to converge to the high fidelity solution because it utilizes a trust region move limit strategy along with its local trend matching. This framework can be applied to many types of problems including: single discipline optimization, multidisciplinary design optimization, design spaces that are smooth and continuous, design spaces that are not smooth or continuous, and even to simulations that fail at certain points. This robustness is reliant on the choice of inner optimizer of the framework which finds the optima of the scaled low fidelity model; though, for non-smooth, non-continuous design spaces, convergence cannot be guaranteed.
CHAPTER 3

TAYLOR SERIES BASED SCALING APPROXIMATIONS

Scaling models are of utmost importance to the variable fidelity framework. The scaling functions that match the low fidelity model to the high fidelity model are unknown and need to be approximated in clever ways to reduce computational expense. This can be accomplished by ensuring design convergence using the lowest number of high fidelity function calls as possible. This chapter describes ways to construct the scaling functions practically and inexpensively, just one high fidelity function call and gradient call per iteration, using Taylor series expansion. A first order scaling for each of the multiplicative, additive, and hybrid methods is given first. This is followed by a second order approximation of the same methods. Traditionally the cost of the second order methods has been prohibitive; to completely avoid the extra function calls required in obtaining second order information, variable metric methods are used. Each of the first and second order methods are derived theoretically and then demonstrated analytically on a simple unconstrained one-dimensional problem. This provides intuitive physical information as to how the various methods are applied and differ from one another. At the end of the chapter, two larger design problems are solved allowing for a practical comparison of the various Taylor series based methods.
3.1 First Order Scaling Models

For both of the multiplicative and additive scaling methods it is possible to construct various scaling models. In this section the first order method is presented; a higher order method is presented in the next section.

3.1.1 First Order Multiplicative Scaling

The first order multiplicative approximation model is found using Chang’s scaling function $\beta(x)$. At a given design point, for example the current design, this function is defined as

$$\beta(x_n) = \frac{f_{\text{high}}(x_n)}{f_{\text{low}}(x_n)}.$$

(3.1)

This scaling factor at any other point can be approximated using a Taylor series to first order

$$\tilde{\beta}(x) = \beta(x_n) + \nabla \beta(x_n)^T (x - x_n).$$

(3.2)

To evaluate this, the gradient information is needed and can be obtained by differentiating Equation (3.1), resulting in

$$\nabla \beta(x_n) = \begin{bmatrix} \frac{f_{\text{low}}(x_n) \frac{\partial f_{\text{high}}}{\partial x_1}(x_n)}{f_{\text{low}}(x_n)^2} & -\frac{f_{\text{high}}(x_n) \frac{\partial f_{\text{low}}}{\partial x_1}(x_n)}{f_{\text{low}}(x_n)^2} \\ \vdots & \vdots \\ \frac{f_{\text{low}}(x_n) \frac{\partial f_{\text{high}}}{\partial x_m}(x_n)}{f_{\text{low}}(x_n)^2} & -\frac{f_{\text{high}}(x_n) \frac{\partial f_{\text{low}}}{\partial x_m}(x_n)}{f_{\text{low}}(x_n)^2} \end{bmatrix}.$$  

(3.3)
Therefore, a first order update on the low fidelity model is

\[ f_{\text{high}} \approx \tilde{\beta}(x) f_{\text{low}}. \quad \text{(3.4)} \]

This model insures that at the initial design point, the updated low fidelity model matches the function and the gradient of the high fidelity model. The identical process is done in order to scale each constraint.

Example: First Order Multiplicative Scaling

In order to demonstrate the variable fidelity framework and more specifically the different types of scaling methods, the first iteration of a one dimensional example is worked out explicitly. This example is repeated for each of the different scaling methods presented in this chapter and the results are summarized. This simple problem is useful for increasing understanding of the methodology and gaining intuition as to how the various scalings work and can be applied for different types of problems. The problem is unconstrained with the following high and low fidelity objective functions:

\[ f_{\text{high}}(x) = (x - 4)^2, \quad \text{(3.5)} \]
\[ f_{\text{low}}(x) = (x + 1). \quad \text{(3.6)} \]

The initial design point is taken to be \( x_0 = 1 \) with an initial trust region size of \( \Delta_0 = 1 \). At the initial design the fidelity models have the values of \( f_{\text{high}}(x_0) = 9 \) and \( f_{\text{low}}(x_0) = 2 \). For each of the scaling methods we will also need gradient
information; for this example problem analytic gradients will be used:

\[
\frac{\partial f_{\text{high}}}{\partial x} = \nabla f_{\text{high}} = 2(x - 4), \quad (3.7)
\]

\[
\frac{\partial f_{\text{low}}}{\partial x} = \nabla f_{\text{low}} = 1. \quad (3.8)
\]

The final information needed to complete the first two steps of the variable fidelity framework is the gradient value of each objective function at the initial design point: \( \nabla f_{\text{high}}(1) = -9 \) and \( \nabla f_{\text{low}}(1) = 1 \). All of the examples presented in this chapter use the same first two steps but will differ from Step 3 onward.

To build the first order multiplicative scaling function, Equations (3.7) and (3.8) are substituted into Equations (3.2) and (3.3) which simplifies to

\[
\tilde{\beta}(x) = \frac{9}{2} - \frac{21}{4}(x - 1). \quad (3.9)
\]

Minimizing the scaled low fidelity model, \( \tilde{\beta}(x)f_{\text{low}}(x) \), subject to the trust region step size constraint results in a new design point \( x_1 = 2 \). This minimization completes Step 4. The design space including the high fidelity, low fidelity, and scaled low fidelity model is shown in Figure 3.1. The figure shows that the scaled model has matched the value and gradient of the high fidelity function at the initial design point.

In Step 5 a penalty function is calculated, but because the problem is unconstrained, the penalty function is simply the value of the high fidelity function at this new design point \( f_{\text{high}}(x_1 = 2) = 4 \). In Step 6 the trust region management is
performed. First a trust region ratio is calculated to determine if this new point should be accepted and to check the quality of the approximation this scaling model provides. The ratio is calculated as follows:

\[
\rho = \frac{f_{\text{high}}(x_0) - f_{\text{high}}(x_1)}{f_{\text{low}}(x_0)\beta(x_0) - f_{\text{low}}(x_1)\beta(x_1)} = \frac{9 - 4}{9 - \frac{9}{4}} = 0.4444.
\]  (3.10)

Since 0.25 < \rho < 0.75, the new point is accepted and the trust region size is unchanged. This means that the scaling model provided a decrease in the high fidelity objective function and gave an acceptable approximation.

3.1.2 First Order Additive Scaling

The first order additive scaling method is similar to the first order multiplicative scaling method because it tries to approximate the high fidelity model by
applying a correction to the lower fidelity model. The additive method was used by Lewis and Nash\cite{87} to solve multigrid problems but can be used more generally.

At a given design point, the additive scaling function has the value

\[
\gamma(x_n) = f_{\text{high}}(x_n) - f_{\text{low}}(x_n). \tag{3.11}
\]

This additive scaling factor at any other point can be approximated using a Taylor series to first order:

\[
\tilde{\gamma}(x) = \gamma(x_n) + \nabla \gamma(x_n)^T (x - x_n). \tag{3.12}
\]

Evaluating this requires gradient information which can be obtained by differentiating Equation \ref{3.11} This gives

\[
\nabla \gamma(x_n) = \nabla f_{\text{high}}(x_n) - \nabla f_{\text{low}}(x_n). \tag{3.13}
\]

Therefore, a first order update on the low fidelity model is

\[
f_{\text{high}}(x) \approx f_{\text{low}}(x) + \tilde{\gamma}(x). \tag{3.14}
\]

This model insures that at the current design point, the updated low fidelity model matches both the function and the gradient of the high fidelity model exactly, which is required for proof of convergence. Nearby points should also approximate the high fidelity response well.
Example: First Order Additive Scaling

To build the first order additive scaling function Equations (3.7) and (3.8) are substituted into Equations (3.12) and (3.13). This simplifies to

\[ \tilde{\gamma}(x) = -7x + 14. \] (3.15)

Minimizing the scaled low fidelity model, \( f_{low}(x) + \tilde{\gamma}(x) \), subject to the trust region step size constraint results in a new design point \( x_1 = 2 \). This minimization completes Step 4 of the variable fidelity framework. The design space including the high fidelity, low fidelity, and scaled low fidelity model is shown in Figure 3.2. The figure shows that the scaled model has indeed matched the value and gradient of the high fidelity function at the initial design point.

Next the trust region management is performed. First a trust region ratio is calculated to see if this new point should be accepted and to see how good of an approximation this scaling model provides. The ratio is calculated as follows:

\[
\rho = \frac{f_{high}(x_0) - f_{high}(x_1)}{(f_{low}(x_0) + \gamma(x_0)) - (f_{low}(x_1) + \gamma(x_1))} = \frac{9 - 4}{9 - 3} = 0.8333. \] (3.16)

Since \( 0.75 < \rho < 1.25 \) the new point is accepted and the trust region size is expanded. This means that the scaling model was a good approximation for the high fidelity model and can be trusted over a larger segment of the design space.
3.1.3 First Order Adaptive Hybrid Scaling

The adaptive hybrid scaling method was developed in Section 2.7. Here this method is approximated using the first order methods discussed previously in this chapter. The high fidelity model is approximated with a scaled low fidelity model consisting of a weighted sum of the first order additive and multiplicative scalings. This is stated mathematically as

\[ f_{\text{high}}(x) \approx f_{\text{scaled}}(x) = W f_{\text{low}}(x) + (1 - W) (f_{\text{low}}(x) + \tilde{\gamma}(x)). \]  

(3.17)

The weighting term, \( W \), is calculated using Equation (2.22) when past information is available; otherwise \( W = \frac{1}{2} \). An example of using this method is given next.

Example: First Order Adaptive Hybrid Scaling

Figure 3.2. One-dimensional example using a first order additive scaling function.
In this example two cases are studied. The first case is using the hybrid method with no adaptive weighting, as is typically used for the first iteration. The second case uses the adaptive hybrid method assuming that a previous point, \( x_{pp} = 0 \), existed and the two different fidelity functions have been evaluated at that point.

In the first case the first order hybrid scaling function is constructed by substituting the two first order scaling models, Equations (3.9) and (3.15), with the high and low fidelity models, Equations (3.7) and (3.8), into Equation (3.17) with a weighting of \( \mathcal{W} = 0.5 \). The first order hybrid scaled low fidelity function simplifies to:

\[
f_{scaled}(x) = -\frac{3}{8}(7x^2 + 2x - 33).
\] (3.18)

Minimizing \( f_{scaled}(x) \) subject to the trust region step size constraint results in a new design point \( x_1 = 2 \). The design space including the high fidelity, low fidelity, and scaled low fidelity model is shown in Figure 3.3. The figure shows that the model matches the value and gradient of the high fidelity function at the initial design point.

Next the trust region ratio is calculated to see if this new point should be accepted and to see how good of an approximation this scaling model provides. The ratio is calculated as follows:

\[
\rho = \frac{f_{high}(x_0) - f_{high}(x_1)}{(f_{low}(x_0) + \gamma(x_0)) - (f_{low}(x_1) + \gamma(x_1))} = \frac{9 - 4}{9 - \frac{3}{8}} = 0.5797. \tag{3.19}
\]

Since \( 0.25 < \rho < 0.75 \) the new point is accepted and the trust region size remains the same. This scaling provides a better approximation than the first order
multiplicative method but not as good as the first order additive method.

In the second case it is assumed that there is past information from the point $x_{pp} = 0$. Using this information and Equation (2.22), the adaptive weighting can be calculated:

$$W = \frac{f_{\text{high}}(x_{pp}) - (f_{\text{low}}(x_{pp}) + \gamma(x_{pp}))}{f_{\text{low}}(x_{pp})\tilde{\beta}(x_{pp}) - (f_{\text{low}}(x_{pp}) + \gamma(x_{pp}))} = -\frac{4}{21}. \quad (3.20)$$

Using this new weighting the scaled low fidelity function becomes

$$f_{\text{scaled}}(x) = (x - 4)^2, \quad (3.21)$$

which is an exact match to the high fidelity function. Because of the trust region restriction the new design point will be $x_1 = 2$. Since it is an exact match the trust region ratio is by definition 1. This scaling has outperformed each of the
other first order methods on this example problem. This exact match is not surprising due to the order reduction of the low fidelity model and the derivation of the adaptive weighting method.

3.2 Second Order Scaling Models

In this section the scaling techniques are expanded to second order. The increased order allows for faster convergence, though it requires second derivative information. This information can be approximated using consecutive first order information and is described in the later parts of this section.

3.2.1 Multiplicative Second Order Scaling

Using the same idea as in the first order scaling method, an approximate multiplicative scaling function can be derived to match second order information. This approach was first used by Gano et al. and Eldred et al. using both second order information and approximate second order information. The approach is analogous to the first order method except the Taylor series approximation is expanded to include the second order terms as the name implies. The result for the multiplicative method is

\[ \tilde{\beta}(x) \approx \beta(x_c) + \Delta x^T \nabla \beta(x_c) + \frac{1}{2} \Delta x^T \nabla^2 \beta(x_c) \Delta x. \] (3.22)

Using the same gradient result as in Section 3.2, the only remaining term needed is the Hessian of \( \beta \); this can be found by differentiating again, which simplifies to:
\[ \nabla^2 \beta(x_c) = \begin{bmatrix}
    h_{1,1} & h_{1,2} & \cdots & h_{1,n} \\
    h_{2,1} & \ddots & \vdots & \vdots \\
    \vdots & \ddots & \ddots & \vdots \\
    h_{n,1} & \cdots & \cdots & h_{n,n}
\end{bmatrix}, \tag{3.23} \]

where

\[ h_{i,j} = \frac{1}{f_{\text{low}}^2} \left( 2f_{\text{high}} \frac{\partial f_{\text{low}}}{\partial x_i} \frac{\partial f_{\text{low}}}{\partial x_j} + f_{\text{low}}^2 \frac{\partial^2 f_{\text{high}}}{\partial x_i \partial x_j} - f_{\text{low}} \left( \frac{\partial f_{\text{low}}}{\partial x_j} \frac{\partial f_{\text{high}}}{\partial x_i} + \frac{\partial f_{\text{high}}}{\partial x_j} \frac{\partial f_{\text{low}}}{\partial x_i} + f_{\text{high}} \frac{\partial^2 f_{\text{low}}}{\partial x_i \partial x_j} \right) \right), \tag{3.24} \]

where all the functions and partial derivatives are evaluated at \( x = x_c \) and \( i \) and \( j \) are the indices for the Hessian matrix which run from 1 to the number of design variables. Computing this symmetric full rank matrix would be quite expensive; therefore, Hessian update methods such as BFGS and SR1 may be used to compute these terms. The scaling function has the same form as in the first order method and can similarly be computed for the constraints as well as the objective function.

To maintain the computational efficiency of the first order method and gain some advantage by using the second order scaling function, the Hessian information must be estimated without any additional cost. Since the cost is heavily associated with high fidelity function calls, the Hessian can be approximated without having to use second order finite differences which require on the order of \( n^2 \) high fidelity function calls. Two of the most popular methods to approximate the Hessian are the BFGS\(^{[22]} \) and SR1\(^{[112]} \) methods; they are discussed later. Also,
a scaling technique\cite{112} can be used before the first update to help convergence. This initial scaling technique is given in more detail in Section 3.2.3.1.

Example: Multiplicative Second Order Scaling

The second order multiplicative scaling function is built by substituting Equations (3.7) and (3.8) into Equations (3.22), (3.3), and (3.23). This simplifies to

\[ \tilde{\beta}(x) = \frac{1}{8}(25x^2 - 92x + 103). \]  

Minimizing the scaled low fidelity model, \( f_{low}(x)\tilde{\beta}(x) \), subject to the trust region step size constraint results in a new design point \( x_1 = 1.700 \) \( (f_{high}(x_1) = 5.2881) \).

This minimization completes Step 4 of the variable fidelity framework. The design space including the high fidelity, low fidelity, and scaled low fidelity model is shown in Figure 3.4.

Next the trust region ratio is calculated to see whether or not this new point should be accepted and to see how good of an approximation this scaling model provides. The ratio is calculated as follows:

\[ \rho = \frac{f_{high}(x_0) - f_{high}(x_1)}{(f_{low}(x_0) + \gamma(x_0)) - (f_{low}(x_1) + \gamma(x_1))} = \frac{9.0 - 5.2881}{9.0 - 6.3619} = 1.4070. \]  

Since \( \rho > 1.25 \) the new point is accepted and the trust region size is kept the same. When the trust region ratio is greater than 1 it means that the high fidelity model had a larger decrease than the scaling model predicted. The first order
method had a trust region ratio that was further from 1 so the second order method is a better approximation. However, this method resulted in a smaller step because of the local minima created by the more complex design space.

3.2.2 Additive Second Order Scaling

The additive second order method is found by following the same steps as in the second order multiplicative method. First a Taylor series of $\gamma$ is expanded out to second order around the point $x_c$:

$$\tilde{\gamma}(x) \approx \gamma(x_c) + \Delta x^T \nabla \gamma(x_c) + \frac{1}{2} \Delta x^T \nabla^2 \gamma(x_c) \Delta x. \quad (3.27)$$

The first order information was found in Section 3.12 so the only remaining information needed is the Hessian of $\gamma$; this can be found by taking the gradient
of the gradient of $\gamma$:

$$\nabla^2 \gamma(x_c) = \nabla^2 f_{\text{high}}|_{x=x_c} - \nabla^2 f_{\text{low}}|_{x=x_c}. \quad (3.28)$$

As with the multiplicative method, the second order information is too expensive to evaluate via finite differences. Methods to approximate this information are discussed next. Similar information is also required to build scaling functions for the constraints.

Example: Additive Second Order Scaling

The second order additive scaling function is built by substituting Equations (3.7) and (3.8) into Equations (3.27), (3.13), and (3.28) which simplifies to

$$\hat{\gamma}(x) = x^2 - 9x + 15. \quad (3.29)$$

When this is combined with the low fidelity model it results in the exact high fidelity function! Because of the trust region size constraint the next point would be $x_1 = 2$ and the trust region ratio would be $\rho = 1$. This result is due to the fact that the low fidelity function can be matched exactly to the high fidelity function by adding a second order polynomial.
3.2.3 Quasi-Second Order Scaling

The second order information that is needed in both second order techniques can be very costly to compute. There exist techniques to approximate the second order information from first order information, which is calculated at each iteration of the variable fidelity optimization process. In this investigation the second order information can, therefore, be obtained at no additional cost, in terms of function calls, compared to the first order scaling methods from Sections 3.1.3.2 and 3.1.3.12. The two most prevalent methods used are the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update and the symmetric-rank-1 (SR1) update. These methods developed out of need to reduce the cost of Newton like optimization algorithms. The resulting class of algorithms became known as quasi-Newton methods and were shown by Dennis and Moré\[47\] to be highly effective.

3.2.3.1 Initial Scaling of the Hessian

Typically, the initial guess for the Hessian information, $H$, for the objective and each constraint function is chosen to be the identity matrix. The scaling of this initial guess may be undesirable, so a scaling method is highly suggested as is discussed by Nocedal and Wright \[112\]. The scaling is derived from the approximate values of the eigenvalues of the Hessian. This scaling factor is given as

$$H_o = \frac{y_n^T y_n}{y_n^T s_n} I,$$

(3.30)

where
\[ y_n = \nabla f_n - \nabla f_{n-1} \quad (3.31) \]
\[ s_n = x_n - x_{n+1}. \quad (3.32) \]

In the above equations, \( f \) is the function to which the approximate second order information is desired.

3.2.3.2 The BFGS and Damped BFGS Updates

Broyden\[29\], Fletcher\[50\], Goldfarb\[60\], and Shanno\[137\] developed a rank-2 update for approximating the Hessian based on consecutive first order information; this method is referred to as the BFGS update and is defined as

\[ H_{n+1} = H_n - \frac{H_n s_n s_n^T H_n}{s_n^T H_n s_n} + \frac{y_n y_n^T}{s_n^T y_n}, \quad (3.33) \]

where \( H_n \) is the approximation to the Hessian after \( n \) updates. The BFGS method can become unstable when the curvature condition, \( s_n^T y_k > 0 \), is not met or for very nonlinear problems with an indefinite Hessian. A more robust implementation is the damped BFGS\[112\]. The damped BFGS was developed to dampen the effect of performing an update when the curvature condition, \( s_n^T y_n > 0 \), was either violated or nearly violated. This update is defined by the following expression.

\[ H_{n+1} = H_n - \frac{H_n s_n s_n^T H_n}{s_n^T H_n s_n} + \frac{r_n r_n^T}{s_n^T r_n}, \quad (3.34) \]
where the damping factor $r_n$ is defined as

$$r_n = \Theta_n y_n + (1 - \Theta_n) H_n s_n,$$  \hspace{1cm} (3.35) 

and the scalar $\Theta_n$ is determined based on the following rules:

$$\Theta_n = \begin{cases} 
1 & : s_n^T y_n \geq 0.2 s_n^T H_n s_n \\
\frac{0.8s_n^T H_n s_n}{s_n^T H_n s_n - s_n^T y_n} & : s_n^T y_n < 0.2 s_n^T H_n s_n 
\end{cases} \hspace{1cm} (3.36)$$

Both the damped and undamped BFGS updates have another important feature. If they are applied to a positive definite matrix, then the update will remain positive definite. This is especially useful when using approximate line searches.

The work done by Eldred et al. \cite{49} noted that the damped BFGS update may significantly degrade convergence performance for model management trust region based methods. While damped BFGS update works appropriately for Newton based optimization methods, Eldred et al. stated that steps generated in model management types of frameworks generally are not Newton-like and can, therefore, violate the curvature condition. They suggested using the standard BFGS update with the caveat, that if the denominator becomes too small,

$$|s_n^T y_n| < 10^{-6} s_n^T H_n s_n,$$ \hspace{1cm} (3.37)

then the update is skipped for that iteration and $H_{n+1} = H_n$. This issue may be problem dependent, though as it was used successfully by Gano et al.\cite{53,54}. The BFGS or damped BFGS may still behave poorly on certain problems, given that neither directly addresses the problem when there are negative eigenvalues in ei-
ther the high or low fidelity Hessian. In such cases the SR1 method should be used.

3.2.3.3 The Symmetric-Rank-1 Update

The symmetric-rank-1 (SR1) update, unlike the BFGS update, does not guarantee that the updated matrix maintains positive definiteness. This property could be beneficial to the variable fidelity framework because it can capture the true nature of the second order information without relying on the assumption that the design space is positive definite. A downside is that this update is only rank one. The SR1 update is

\[
H_{n+1} = H_n + \frac{(y_n - H_ns_n)(y_n - H_ns_n)^T}{(y_n - H_ns_n)^T s_n}.
\]

(3.38)

A safeguard must be employed when using the SR1 update in variable fidelity optimization to prevent division by near zero or zero. If

\[
|(y_n - H_ns_n)^T s_n| < 10^{-6} \|y_n - H_ns_n\|_2 \|s_n\|_2
\]

(3.39)
is true then the update is skipped for that iteration; \(H_{n+1} = H_n\).

3.2.4 Second Order Adaptive Hybrid Scaling

The adaptive hybrid scaling method was developed in Section 2.7, and its first order approximation was described earlier in this chapter. Here the approximation is extended to second order. The high fidelity model is approximated with a scaled
low fidelity model consisting of a weighted sum of the second order additive and multiplicative scalings. This is summarized in the following equation:

\[
    f_{\text{high}}(x) \approx f_{\text{scaled}}(x) = W f_{\text{low}} \tilde{\beta}(x) + (1 - W) (f_{\text{low}}(x) + \tilde{\gamma}(x)),
\]

(3.40)

where \( \tilde{\beta}(x) \) and \( \tilde{\gamma}(x) \) are given by Equations (3.22) and (3.27), respectively, and can use the true Hessian or approximate Hessian information. The weighting term, \( W \), is calculated using Equation (2.22) when past information is available; otherwise, by convention \( W = \frac{1}{2} \). An example of using this second order method is described next.

3.2.4.1 Example: Second Order Adaptive Hybrid Scaling

As in the first order example, two cases are studied. The first case is using the hybrid method with no adaptive weighting, as is typically done in the first iteration. The second case uses the adaptive hybrid method, assuming that there existed a previous point, \( x_{pp} = 0 \), and the two fidelity functions have been evaluated at that point.

In the first case the second order hybrid scaling function is constructed by substituting the two second order scaling models, Equations (3.25) and (3.29), with the high and low fidelity models, Equations (3.7) and (3.8), into Equation (3.40) with a weighting of \( W = 0.5 \). The second order hybrid scaled low fidelity function simplifies to:

\[
    f_{\text{scaled}}(x) = \frac{1}{16} (25x^3 - 59x^2 - 53x + 231).
\]

(3.41)
Minimizing $f_{scaled}(x)$ subject to the trust region step size constraint gives a new design point $x_1 = 1.9380$ ($f_{high}(x_1) = 4.2520$). The design space including the high fidelity, low fidelity, and scaled low fidelity model is shown in Figure 3.5.

![Figure 3.5. One-dimensional example using a second order hybrid scaling function.](image)

Next the trust region ratio is calculated to see if this new point should be accepted and to determine how accurate of an approximation this scaling model provides. The ratio is calculated as follows:

$$
\rho = \frac{f_{high}(x_0) - f_{high}(x_1)}{(f_{low}(x_0) + \gamma(x_0)) - (f_{low}(x_1) + \gamma(x_1))} = \frac{9.0 - 4.2520}{9.0 - 5.5414} = 1.3728.
$$ (3.42)
Since $\rho > 1.25$ the new point is accepted and the trust region size remains the same. This scaling provides a better approximation than the first order multiplicative method but not as good as the first order additive method which matched the high fidelity model exactly.

In the second case it is assumed that there is past information from the point $x_{pp} = 0$. Using this information and Equation (2.22) the adaptive weighting can be calculated:

$$W = \frac{f_{high}(x_{pp}) - (f_{low}(x_{pp}) + \tilde{\gamma}(x_{pp}))}{f_{low}(x_{pp})\tilde{\beta}(x_{pp}) - (f_{low}(x_{pp}) + \tilde{\gamma}(x_{pp}))} = 0.$$  \hspace{1cm} (3.43)

The weighting of 0 corresponds to just using the additive method which exactly matched the high fidelity model. This again helps demonstrate the utility that the adaptive hybrid method provides in selecting the correct scaling method without the need for a designer’s intervention.

3.2.4.2 Example: Summary of Taylor Series Based Scaling Methods

A summary of all of the Taylor series based scaling scaling functions is presented in Figure 3.6. This figure shows all of the scaled low fidelity functions along with the high and low fidelity functions on a single plot. The plot shows that in most cases the second order methods provide a better approximation within the trust region than do the corresponding first order methods. Both the first and second order adaptive hybrid methods as well as the second order additive model match the high fidelity model exactly for this demonstration problem.
3.3 Numerical Implementation Studies

To demonstrate the savings of the approximate first order scaling, second order scaling, and hybrid methods developed in this chapter, two problems are solved. The first problem is a simple two dimensional analytic problem that is computationally inexpensive and shows how the various methods differ. The second problem is a realistic engineering design problem that uses complex computational models. In this problem, an advanced energy efficient transport (AEET) high-lift airfoil is designed by placing the slats and flap components to maximize its lifting capability in slow flight.

The goal of these demonstrations is to compare and contrast the various variable
fidelity scaling methods. In order to compare the various methods, we will take a look primarily at the number of high fidelity function calls. Other aspects of the methods that will be compared are the number of low fidelity function calls, the final trust region size, and the number of iterations performed. The number of low fidelity function calls may be important, depending on their computational expense relative to the high fidelity model. The trust region size at the end of the optimization gives insight into how well the scaling function performed; the larger the region the better. For comparison purposes, the number of function calls needed for a standard sequential quadratic programming (SQP) optimization performed on the high fidelity model alone is also presented for each problem.

### 3.3.1 2D Analytic Problem

This test problem serves two main purposes: it first allows for verification that the implementations converge and second it demonstrates the cost savings of the variable fidelity methods over the existing traditional methods. The problem is analytic and two dimensional, making it easy to create contour plots of the design space for a given iteration. These contour plots can be used to visually compare the high fidelity space as well as the scaled low fidelity space. The problem is given in Equations 3.44 through 3.47. The initial design point was [1.5, 1.5]\(^T\), and both design variables were bounded between 0.1 and 10. The low fidelity model adds linear and nonlinear noise factors to the high fidelity model to change the shape of the design space and location of the optima. For this problem all gradient information was obtained using finite differencing.
\[
f_{\text{high}} = 4x_1^2 + x_2^3 + x_1x_2 \quad (3.44) \\
g_{\text{high}} = \frac{1}{x_1} + \frac{1}{x_2} - 2 \quad (3.45) \\
f_{\text{low}} = 4(x_1 + 0.1)^2 + (x_2 - 0.1)^3 + x_1x_2 + 0.1 \quad (3.46) \\
g_{\text{low}} = \frac{1}{x_1} + \frac{1}{x_2 + 0.1} - 2 - 0.001 \quad (3.47)
\]

Each of the methods used converged to the same solution, the optima of the high fidelity problem, and all used the same convergence criteria of \( \epsilon_f = \epsilon_x = 0.0001 \).

The contour plots at an intermediate iteration of the optimization are given in Figure 3.7 for the multiplicative first order and second order BFGS methods. The second order model better approximates the objective and constraints over a larger range. Table 3.1 shows the results of the variable fidelity optimization using combinations of first order, second order BFGS, second order SR1, and second order full hessian updates. The number of high and low fidelity function calls, total number of iterations, and final trust region size \( \Delta \) are given for comparison. The SQP optimization results are also given at the end of the table for comparison to a state-of-the-art single fidelity optimization method. The final column of Table 3.1 tabulates the savings in high fidelity function calls for each method compared to the SQP results.

The plots in Figure 3.7 depict some of the differences between using a first order and second order approach. The most noticeable difference between these plots is that the second order model does a much better job of approximating the objective function and the constraint boundary; this would imply a faster convergence.
Figure 3.7. Comparing the design spaces of the 1st order and 2nd order BFGS multiplicative scaling methods after 4 iterations.
rate. Another major difference is that the trust region size is much larger for the second order method. This larger trust region is a qualitative indication that the second order model is doing a better job scaling the low fidelity model to match the high fidelity model.

Many observations can be made from comparing the results in Table 3.1. A significant trend observed is that the second order methods improved the performance of most of the scaling methods. For this problem, the additive method worked much better than the multiplicative method; however, the hybrid scaling method did very well adapting to match the additive method. In general, the BFGS and SR1 second order methods performed better than the first order methods. And there was no significant difference between the BFGS and SR1 methods for this problem. The second order method using the full Hessian calculation had mixed results in terms of having fewer high fidelity function calls than the first order model. However, with an increase in the number of design variables the second order information would become much more expensive to compute. Furthermore, all of the variable fidelity methods used fewer high fidelity function calls than the SQP method, showing the method can even be competitive for small problems depending on the relative computational cost between the high and low fidelity models.

3.3.2 Energy Efficient Transport High-Lift Airfoil Design

For about three decades, since the mid 1970s, the National Aeronautics and Space Administration (NASA) has conducted research to improve the efficiency
<table>
<thead>
<tr>
<th>Method</th>
<th>High</th>
<th>Low</th>
<th>Iter.</th>
<th>Final Δ</th>
<th>Savings/SQP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Multiplicative Scaling</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1st Order</td>
<td>35</td>
<td>187</td>
<td>14</td>
<td>0.0002</td>
<td>20.5%</td>
</tr>
<tr>
<td>2nd Order, BFGS</td>
<td>18</td>
<td>145</td>
<td>7</td>
<td>0.0625</td>
<td>59.1%</td>
</tr>
<tr>
<td>2nd Order, SR1</td>
<td>18</td>
<td>147</td>
<td>7</td>
<td>0.1875</td>
<td>59.1%</td>
</tr>
<tr>
<td>2nd Order, FULL</td>
<td>25</td>
<td>115</td>
<td>4</td>
<td>1.0000</td>
<td>43.2%</td>
</tr>
<tr>
<td><strong>Additive Scaling</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1st Order</td>
<td>13</td>
<td>89</td>
<td>4</td>
<td>0.2500</td>
<td>70.5%</td>
</tr>
<tr>
<td>2nd Order, BFGS</td>
<td>13</td>
<td>94</td>
<td>4</td>
<td>1.0000</td>
<td>70.5%</td>
</tr>
<tr>
<td>2nd Order, SR1</td>
<td>16</td>
<td>110</td>
<td>5</td>
<td>1.0000</td>
<td>63.6%</td>
</tr>
<tr>
<td>2nd Order, FULL</td>
<td>25</td>
<td>88</td>
<td>4</td>
<td>1.0000</td>
<td>43.2%</td>
</tr>
<tr>
<td><strong>Adaptive Hybrid Scaling</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1st Order</td>
<td>20</td>
<td>159</td>
<td>9</td>
<td>0.0010</td>
<td>54.5%</td>
</tr>
<tr>
<td>2nd Order, BFGS</td>
<td>13</td>
<td>96</td>
<td>4</td>
<td>1.0000</td>
<td>70.5%</td>
</tr>
<tr>
<td>2nd Order, SR1</td>
<td>13</td>
<td>93</td>
<td>4</td>
<td>1.0000</td>
<td>70.5%</td>
</tr>
<tr>
<td>2nd Order, FULL</td>
<td>25</td>
<td>97</td>
<td>4</td>
<td>1.0000</td>
<td>43.2%</td>
</tr>
<tr>
<td><strong>SQP</strong></td>
<td>44</td>
<td>-</td>
<td>8</td>
<td>-</td>
<td>0%</td>
</tr>
</tbody>
</table>
of jet transport aircraft. Part of this research effort included the energy efficient transport program, which developed supercritical airfoils with larger section thickness-to-chord ratios, higher aspect ratios, higher cruise lift coefficients, and less swept wings. Because these wings had higher lift coefficients at cruise they could be smaller and more fuel efficient. With the reduced wing area, these new wings needed a high-lift flap system to ensure that takeoff and landing requirements could be met.

The problem solved here is: given a high-lift airfoil, find the optimal placement of its slat, vane, and flap to provide a maximum coefficient of lift for takeoff or landing configurations. The problem was developed from the experimental and numerical work done at NASA Langley\[55\, 91\, 108\]. The problem consists of nine design variables which control the horizontal, vertical, and rotational orientation of the slat vane and flap relative to their cruise configuration. The rotation is measured positive counter-clockwise about each control surface’s leading edge. Figure \ref{fig:3.8} shows the layout of the supercritical airfoil and the control surface movements. The only constraints placed on the system were that the gaps between control surfaces must be positive in value for gridding purposes.

The flow conditions for the problem consisted of a Reynolds number of 9 million, Mach number of 0.25, and an angle of attack of 3 degrees relative to the baseline configuration. The flow was solved using the inviscid Euler’s equations for the low fidelity model. The grid consisted of about 45,000 elements, which extended to 30 times the chord length in each direction. A portion of this grid is shown in Figure \ref{fig:3.9}a. For the high fidelity model, a full Navier-Stokes solution was used;
the grid consisted of about 100,000 elements as seen in Figure 3.9. The CFD runs took approximately 9 minutes and 2.5 hours for the low and high fidelity models, respectively. In testing the variable fidelity framework on this problem, a kriging surrogate model was created for the purpose of reducing the cost to a tractable level for debugging and testing, which follows from the work done by Alexandrov[15]. However, a relative cost, measured in hours, for each method was calculated using the true computational expense of the high and low fidelity models.

Both fidelity models were solved using a computational fluid dynamics package developed at NASA Langley called FUN2D [17, 18]. This package uses fully unstructured mesh, which were generated using the advancing-front local-reconnection method described by Marcum [97, 98].

The results of the optimization trials are presented Table 3.2. Based on the results from the analytic case, only the adaptive hybrid method was used on this problem. Using the same convergence criteria as before, all of the trials converged to the same solution. In addition to the number of high and low fidelity function
calls, number of iterations required for convergence, and the final trust region size, the relative cost between the methods is given. Also, the percentage of computational time saved as compared to the SQP method is shown in Table 3.2.

The results from this larger, more complicated problem further support the claim that the second order methods can save a significant number of function calls as compared to the first order methods. Each variable fidelity method also used less high fidelity function calls than did the standard SQP optimizer; the highest being 42.8% using the second order SR1 method.
### TABLE 3.2

VARIABLE FIDELITY OPTIMIZATION TAYLOR SERIES BASED
RESULTS FOR THE AEET PROBLEM.

<table>
<thead>
<tr>
<th>Method</th>
<th>High</th>
<th>Low</th>
<th>Iter.</th>
<th>Final ∆</th>
<th>Relative Cost</th>
<th>Savings/SQP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adaptive Hybrid Scaling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1st Order</td>
<td>35</td>
<td>279</td>
<td>34</td>
<td>0.000220</td>
<td>134.0 hr</td>
<td>12.1%</td>
</tr>
<tr>
<td>2nd Order, BFGS</td>
<td>21</td>
<td>295</td>
<td>20</td>
<td>0.000103</td>
<td>101.7 hr</td>
<td>33.3%</td>
</tr>
<tr>
<td>2nd Order, SR1</td>
<td>15</td>
<td>298</td>
<td>14</td>
<td>0.000412</td>
<td>87.2 hr</td>
<td>42.8%</td>
</tr>
<tr>
<td>SQP</td>
<td>61</td>
<td>-</td>
<td>20</td>
<td>-</td>
<td>152.5 hr</td>
<td>0%</td>
</tr>
</tbody>
</table>

#### 3.4 Summary

In this chapter, many different Taylor series based scaling techniques were presented for use in variable fidelity optimization including: first order methods, second order methods, approximate second order methods, and an adaptive hybrid method. The methods were presented theoretically and then applied to two demonstration problems. The results of the problems indicated that the approximate second order methods can save substantial computational cost over the first order methods by lowering the number of high fidelity function calls required for optimization. The adaptive hybrid method showed the ability to adjust to an appropriate weighting between the additive and multiplicative methods so as to eliminate the need to determine \textit{a priori} which method should be used. The
only drawback observed at this point for the use of the hybrid method is that it requires twice as much memory to store both the additive and multiplicative models; however, this is presumably offset by the savings in high fidelity calls required for convergence.
The scaling models developed in the previous chapter, namely the first and second order Taylor series approaches, are only local to the current design point and do not use past information, except for the construction of the second order information using the BFGS or SR1 methods. When using variable fidelity physics-based models, the low fidelity model typically is a global model. Likewise, a global scaling function may be better at approximating the high fidelity response. In this chapter a new kriging-based scaling function is developed to improve the scaling between the different fidelity models on a more global scale. This approach allows the use of all information calculated throughout the course of the optimization, even when the trust region test fails. Figure 4.1 depicts the difference between the Taylor series and kriging-based approaches. The figure illustrates how past information can be used to generate a better scaling model. The kriging model can be constructed for any type of scaling function: for example, the additive or multiplicative methods already discussed.

It is important to stress that the savings of modeling the error, as opposed to simply modeling the high fidelity function itself, comes from the fact that the error model can be applied to a physics-based low fidelity model. This scaled low fidelity model provides a better approximation of the high fidelity model than
simply using a mathematically constructed interpolation or data fitting model. Furthermore, creating a model of the error, which is used as a correction, requires few high fidelity samples as compared to building a response surface from high fidelity samples alone.

![Figure 4.1. Drawings depicting different scaling techniques for matching low and high fidelity models.](image)

The kriging model gives exact responses at sample points, as it is an interpolating function. This insures that at least zero order matching is obtained. With the
inclusion of gradient information, first order matching is achieved. Combining first order matching, a trust region model management strategy, and an optimizer that converges to a KKT point of the scaled low fidelity problem, provides for a provably convergent framework\textsuperscript{11}.

Building and rebuilding the kriging models take extra time and memory storage. This added computational time and resource expenditure may be negligent, however, compared to the evaluation of the high fidelity model. The method used to build the kriging models is described in the following subsection.

Another benefit of using a kriging-based scaling approach is that past data can be easily incorporated into the scaling model to further increase the convergence rate. Often, a model is evaluated for various purposes before an optimization is performed. These results can be included in the kriging model to improve its matching capabilities.

4.1 Kriging Overview

Kriging was developed for use in the field of geological statistics and was used in estimating unknown values from known values obtained via a semivariogram. This helped give a best guess global view of desired parameters from a set of known values. The term \textit{kriging} was named after D. G. Krige from South Africa who used this method with much success. (See Cressie\textsuperscript{41} for details.) The method was originally developed for use in the computer science and engineering fields as design and analysis of computer experiments or DACE modeling by Sacks \textit{et al.}\textsuperscript{135}. 

66
Kriging methods have been used to model the response of many engineering systems\cite{24,78,136,140}, including the design of a low-boom business jet by Chung and Alonso\cite{38} and design of an aerospike nozzle by Simpson et al.\cite{139}. Martin and Simpson conducted a study on using kriging models to approximate deterministic computer models and discussed the applicability of various kriging variants\cite{99,102}.

The kriging scaling functions are used in the same manner as the Taylor series based methods: to scale the low fidelity model to match the high fidelity model. However, the kriging model is built using all of the points at which the models have been evaluated, including gradient information or finite difference samples. The method for constructing the kriging scaling functions is briefly explained in the rest of this section.

Kriging begins by modeling an unknown function\cite{44,144}, $y$, with the form

$$y(x) = B + Z(x).$$ (4.1)

The term $B$ is an unknown constant or global trend function, typically linear or quadratic, which is estimated based on the number of sample points, $n_s$. $Z(x)$ is the modeling of a stochastic process with a mean of zero, a variance $\sigma^2$, and a co-variance that is not zero. The covariance of $Z(x)$ is given by

$$\text{Cov}[Z(x^i), Z(x^j)] = \sigma^2 R[R(x^i, x^j)],$$ (4.2)

67
where \( R \) is the correlation matrix, \( R \) is the correlation function which is selected by the user, and \( i \) and \( j \) run from 1 to \( n_s \). It is important to notice that \( R \) is symmetric and has unit values along the diagonal.

The user selects the correlation function when generating the kriging model. In the statistical and engineering literature\(^{[59,139]}\) the Gaussian function is by far the most popular and is also used in this work. It is defined as

\[
R(x^i, x^j) = e^{-\sum_{k=1}^{n_v} \theta_k |x_k^i - x_k^j|^2},
\]

(4.3)

where \( \theta_k \) is the vector of unknown correlation parameters which is of length \( n_v \), the number of design variables. Also, \( x_k^i \) and \( x_k^j \) are the \( k \)th elements of the sample points \( x^i \) and \( x^j \).

The kriging model estimates values of \( y(x) \) at untried values of \( x \) based on a given set of expected values or sample points. This work distinguishes between the true value and an estimated value by using a hat to denote the estimated values, such as \( y \) is the true response value and \( \hat{y} \) is the estimated value. The mean squared error is defined as the square of the expected value of the difference between the real response and the approximated one at any point. Mathematically this is stated as

\[
MSE = E(y(x) - \hat{y}(x))^2.
\]

(4.4)
Because kriging is an interpolation process, the model will have no \( \text{MSE} \) at a sample point. If the \( \text{MSE} \) is minimized, then the kriging predictor is

\[
\hat{y} = \hat{B} + r^T(x)R^{-1}(y - f\hat{B}),
\]

(4.5)

where \( y \) is the vector of responses (objective values) to the sample locations \( \{x^1, \ldots, x^n_s\} \) and \( f \) is a constant vector of all ones with length \( n_s \). In Equation (4.5) the correlation vector, \( r(x) \), is the correlation between the value at a new location \( x \) and the values at the sampled locations. To use this predictor, \( \hat{B} \) and \( \theta_k \) must both be found, as \( r \) and \( R \) depend on \( \theta_k \). The correlation vector is

\[
r(x)^T = [R(x, x^1), \ldots, R(x, x^{n_s})].
\]

(4.6)

The unknown parameters \( \theta_k \) are found using maximum likelihood estimation\(^{[59]}\). This approach uses the likelihood of the assumed Gaussian computer model with kriging parameters of \( \theta_k \) given observations \( y \). This likelihood is defined as,

\[
L(\theta_k | y) = e^\frac{- (y - f\hat{B})^T R(\theta_k)^{-1} (y - f\hat{B})}{2\hat{\sigma}^2} \frac{1}{\sqrt{2\pi\hat{\sigma}^2}^{n_s}|R(\theta_k)|}.
\]

(4.7)

It is more common to use the log of the likelihood for mathematical convenience, which simplifies Equation (4.7) to

\[
\mathcal{L}(\theta_k | y) = -\frac{(y - f\hat{B})^T R(\theta_k)^{-1}(y - f\hat{B})}{2\hat{\sigma}^2} - \frac{n_s\ln(2\pi\hat{\sigma}^2) + \ln(|R(\theta_k)|)}{2}
\]

(4.8)
By setting the derivatives of Equation (4.8) with respect to both $\hat{B}$ and $\hat{\sigma}^2$ to zero, a closed-form solution for the optimal values of the trend function, $\hat{B}$, and estimated variance, $\hat{\sigma}^2$, are found:

$$\hat{B} = (f^T R^{-1} f)^{-1} f^T R^{-1} y,$$  \hspace{1cm} (4.9)

$$\hat{\sigma}^2 = \frac{1}{n_s} \left( (y - f\hat{B})^T R^{-1} (y - f\hat{B}) \right).$$  \hspace{1cm} (4.10)

Finally, $\theta_k$ is found by maximizing the log-likelihood function, known as the maximum likelihood estimation (MLE) problem, which can be reduced to:

$$\text{Maximize} \quad - \frac{n_s \ln(\hat{\sigma}^2) + \ln(|R|)}{2},$$  \hspace{1cm} (4.11)

subject to: $0 < \theta_k \leq \infty$.

This is a $n_v$ dimensional optimization problem which is well posed. Notice that a kriging model can be built for any values of $\theta_k$; this optimization ensures the best choice. In practice, when solving for $\theta_k$, $R$ may become badly scaled; this is overcome using various numerical techniques as described by Lophaven et al.\[95\] [96].

4.2 Warm Starting the Kriging Method

A possible downside to using this kriging method to model the scaling between the variable fidelity models is the fact that during the first few iterations, the kriging
model may not have enough points to accurately model this scaling function. In order to reduce this possible dilemma a warm-started kriging based method is developed. As illustrated in Figure 4.2 this method starts using a Taylor series based method, either first or second order, and then switches to the kriging model. When the Taylor series based method is used, all information is saved, which allows the kriging model to be constructed initially from many more sample points and, therefore, with higher precision. This warm-started approach is not limited by the scaling function used.

![Figure 4.2. Scaling function generation using the warm-started kriging based method.](image)

One other item that should be noted about this method is that a switch between methods only occurs if the point has been accepted in the previous iteration; otherwise, the methodology shrinks the trust region space and uses the same model until a new point is accepted, and then the model is switched.
4.3 Supplying Gradient Information

If low fidelity gradient information is readily available and a gradient based inner optimizer is used, finding the gradients of the scaled low fidelity model is a straightforward process. Having the scaled low fidelity gradients allows for a decrease in function calls by the optimizer since finite differencing techniques are not required. This is especially important if the low fidelity model has a significant computational cost. The gradient information is included in the kriging model using a method called indirect cokriging\textsuperscript{37} which is similar to the gradient enhanced kriging model developed by Lui and Batill\textsuperscript{93}. Gradient information could also be included in the kriging model using other methods such as direct cokriging\textsuperscript{38}.

To find the gradients of the scaled low fidelity models, the appropriate scaling method, additive or multiplicative for example, must be differentiated. Then either the first order, second order, or kriging models must be differentiated with respect to their expansion variables. The kriging model is a special case; its gradients can be found by other means as described by Lophaven\textsuperscript{95} et al.

4.4 Numerical Implementation Studies

To demonstrate the savings of kriging based scaling and the warm-started methods developed in this chapter, two problems are solved. The two dimensional analytic problem and the advanced energy efficient transport (AEET) high-lift airfoil design problems, introduced in Chapter 3, are solved again to allow for comparison of the various scaling techniques.
The goal of these demonstrations is to compare the kriging scaling methods to the Taylor series based scaling methods presented in the previous chapter. In comparing the different methods, the number of high fidelity function calls is the most important factor. Other aspects of the methods that will be compared are the number of low fidelity function calls, the final trust region size, and the number of iterations performed. The number of low fidelity function calls may be important, depending on the computational expense of the low fidelity model relative to the high fidelity model. The trust region size at the end of the optimization gives an insight into how well the scaling function performed; the larger the region the better. For comparison purposes, the number of function calls needed for a standard sequential quadratic programming (SQP) optimization performed on the high fidelity model alone is also presented for each problem. The SQP solver used was Matlab’s \textit{fmincon}[^3], which is included in the optimization toolbox.

4.4.1 2D Analytic Problem

This problem was introduced in Section 3.3.1. For convenience the problem statement is repeated in Equations 4.12 through 4.15. The initial design point was again taken to be $[1.5, 1.5]^T$ and both design variables were bounded between 0.1 and 10. All gradient information was obtained using finite differencing.
\[ f_{\text{high}} = 4x_1^2 + x_2^3 + x_1x_2 \]  
\[ g_{\text{high}} = \frac{1}{x_1} + \frac{1}{x_2} - 2 \]  
\[ f_{\text{low}} = 4(x_1 + 0.1)^2 + (x_2 - 0.1)^3 + x_1x_2 + 0.1 \]  
\[ g_{\text{low}} = \frac{1}{x_1} + \frac{1}{x_2 + 0.1} - 2 - 0.001 \]

All of the variable fidelity methods used converged to the same solution, the optima of the high fidelity problem, and all used the same stopping criteria of \( \epsilon_f = \epsilon_x = 0.0001 \). The contour plots at an intermediate iteration of the optimization are given in Figure 4.3 for the multiplicative first order and kriging methods. The kriging model better approximates the objective and constraints over a larger range. Table 4.1 shows the results of the variable fidelity optimization using combinations of first order, second order BFGS, second order SR1, second order full hessian updates, and kriging models for the various methods. Also included in the table for each method are the results from the warm-started kriging approach. For these trials, the corresponding method was used for the first two iterations. Then a kriging scaling model was used until convergence. The results all compare the the number of high and low fidelity function calls, total number of iterations, and final trust region size \( \Delta \). The last column of the table shows the savings, in percent of high fidelity function calls, of using the warm-started kriging method. The SQP optimization results are also given at the end of the table.

The plots in Figure 4.3 depict some of the differences between using a Taylor series approach and kriging based approach. The most noticeable difference between
Figure 4.3. Comparing the design spaces of the kriging and 1st order multiplicative scaling method after the same number of iterations.
these plots is that the kriging model does a much better job of approximating
the objective function and the constraint boundary; this would imply a faster
convergence rate. Another major difference is that the trust region size is much
larger for the kriging method. This larger trust region is a better qualitative
indication that the kriging method is doing a superior job scaling the low fidelity
model to match the high fidelity model.

Many observations can be made from comparing the results in Table 4.1 A
significant trend observed is that the warm-started kriging method improved
the performance of most of the scaling methods. For this problem, the additive
method worked much better than the multiplicative method; however, the hybrid
scaling method did very well adapting to match the additive method. In this
problem the BFGS and SR1 second order methods generally performed better
than the first order methods. And there was no significant difference between
the BFGS and SR1 methods. The second order method using the full Hessian
calculation had mixed results in terms of having fewer high fidelity function calls
than the first order model. However, with an increase in the number of design
variables the second order information would become much more expensive to
compute. It is also important to note that the kriging method used alone didn’t
show any significant savings over the second order methods. This is because the
kriging model doesn’t include any second order information. Future work could
be done to try to add second order information into the kriging model to improve
its performance when used alone or in the warm-started mode.
<table>
<thead>
<tr>
<th>Method</th>
<th>High</th>
<th>Low</th>
<th>Iter.</th>
<th>Final $\Delta$</th>
<th>High</th>
<th>Low</th>
<th>Iter.</th>
<th>Final $\Delta$</th>
<th>Savings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiplicative Scaling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1st Order</td>
<td>35</td>
<td>187</td>
<td>14</td>
<td>0.0002</td>
<td>25</td>
<td>152</td>
<td>8</td>
<td>0.0156</td>
<td>29%</td>
</tr>
<tr>
<td>2nd Order, BFGS</td>
<td>18</td>
<td>145</td>
<td>7</td>
<td>0.0625</td>
<td>14</td>
<td>137</td>
<td>5</td>
<td>0.0625</td>
<td>22%</td>
</tr>
<tr>
<td>2nd Order, SR1</td>
<td>18</td>
<td>147</td>
<td>7</td>
<td>0.1875</td>
<td>14</td>
<td>137</td>
<td>5</td>
<td>0.0625</td>
<td>22%</td>
</tr>
<tr>
<td>2nd Order, FULL</td>
<td>25</td>
<td>115</td>
<td>4</td>
<td>1.0000</td>
<td>14</td>
<td>99</td>
<td>4</td>
<td>0.0625</td>
<td>44%</td>
</tr>
<tr>
<td>Kriging</td>
<td>29</td>
<td>244</td>
<td>12</td>
<td>0.0264</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Additive Scaling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1st Order</td>
<td>13</td>
<td>89</td>
<td>4</td>
<td>0.2500</td>
<td>13</td>
<td>112</td>
<td>4</td>
<td>0.2500</td>
<td>0%</td>
</tr>
<tr>
<td>2nd Order, BFGS</td>
<td>13</td>
<td>94</td>
<td>4</td>
<td>1.0000</td>
<td>13</td>
<td>112</td>
<td>4</td>
<td>0.2500</td>
<td>0%</td>
</tr>
<tr>
<td>2nd Order, SR1</td>
<td>16</td>
<td>110</td>
<td>5</td>
<td>1.0000</td>
<td>13</td>
<td>112</td>
<td>4</td>
<td>0.2500</td>
<td>19%</td>
</tr>
<tr>
<td>2nd Order, FULL</td>
<td>25</td>
<td>88</td>
<td>4</td>
<td>1.0000</td>
<td>17</td>
<td>97</td>
<td>5</td>
<td>0.2500</td>
<td>32%</td>
</tr>
<tr>
<td>Kriging</td>
<td>13</td>
<td>92</td>
<td>4</td>
<td>1.0000</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Adaptive Hybrid Scaling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1st Order</td>
<td>20</td>
<td>159</td>
<td>9</td>
<td>0.0010</td>
<td>17</td>
<td>190</td>
<td>6</td>
<td>0.2500</td>
<td>15%</td>
</tr>
<tr>
<td>2nd Order, BFGS</td>
<td>13</td>
<td>96</td>
<td>4</td>
<td>1.0000</td>
<td>10</td>
<td>69</td>
<td>3</td>
<td>0.2500</td>
<td>23%</td>
</tr>
<tr>
<td>2nd Order, SR1</td>
<td>13</td>
<td>93</td>
<td>4</td>
<td>1.0000</td>
<td>13</td>
<td>93</td>
<td>5</td>
<td>0.2500</td>
<td>0%</td>
</tr>
<tr>
<td>2nd Order, FULL</td>
<td>25</td>
<td>97</td>
<td>4</td>
<td>1.0000</td>
<td>20</td>
<td>129</td>
<td>6</td>
<td>0.0156</td>
<td>20%</td>
</tr>
<tr>
<td>Kriging</td>
<td>22</td>
<td>238</td>
<td>7</td>
<td>1.0000</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SQP</td>
<td>44</td>
<td>-</td>
<td>8</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
4.4.2 Energy Efficient Transport High-Lift Airfoil Design

The advanced energy efficient transport (AEET) design problem was introduced in Section 3.3.2. The design problem is: given a high-lift airfoil, find the optimal placement of its slat, vane, and flap to provide maximum lift for takeoff or landing configurations. The problem consists of nine design variables which control the horizontal, vertical, and rotational orientation of the slat vane and flap, relative to their cruise configuration. Figure 3.8 shows the layout of the supercritical airfoil and the control surface movements. The constraints placed on the system are that the gaps between control surfaces must be larger than zero for gridding purposes.

The flow conditions and computational grids, as shown in Figure 3.9, are identical to the values used previously. The CFD runs using FUN2D took approximately 9 minutes and 2.5 hours for the low and high fidelity models, respectively.

The results of the optimization trials are presented in Table 4.2. Based on the results from the analytic case, only the adaptive hybrid method was used on this problem. Using the same convergence criteria as in the 2D analytic problem, all of the trials converged to the same solution.

The results from this larger, more complicated problem further support the claim that the warm-started kriging-based approach can save a significant number of function calls. Each variable fidelity method also used fewer high fidelity function calls than did the standard SQP optimizer. For this problem, the second order and kriging methods outperformed the first order methods, and SR1 was the best.
TABLE 4.2
VARIABLE FIDELITY OPTIMIZATION RESULTS FOR THE AEET PROBLEM.

<table>
<thead>
<tr>
<th>Method</th>
<th>High</th>
<th>Low</th>
<th>Iter.</th>
<th>Final ∆</th>
<th>High</th>
<th>Low</th>
<th>Iter.</th>
<th>Final ∆</th>
<th>Savings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adaptive Hybrid Scaling</td>
<td>35</td>
<td>279</td>
<td>34</td>
<td>0.0002</td>
<td>14</td>
<td>352</td>
<td>13</td>
<td>0.0049</td>
<td>60%</td>
</tr>
<tr>
<td>1st Order</td>
<td>21</td>
<td>295</td>
<td>20</td>
<td>0.0001</td>
<td>12</td>
<td>295</td>
<td>11</td>
<td>0.0029</td>
<td>43%</td>
</tr>
<tr>
<td>2nd Order, BFGS</td>
<td>15</td>
<td>298</td>
<td>14</td>
<td>0.0004</td>
<td>12</td>
<td>378</td>
<td>11</td>
<td>0.0029</td>
<td>20%</td>
</tr>
<tr>
<td>2nd Order, SR1</td>
<td>15</td>
<td>462</td>
<td>14</td>
<td>0.1875</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Kriging</td>
<td>61</td>
<td>-</td>
<td>20</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

of the second order methods. One interesting side effect observed when using the kriging-based methods was an increase in low fidelity function calls. A possible reason for this may be that the scaled kriging model becomes very flat near the optima.

4.5 Summary and Conclusions

In this chapter a kriging-based scaling method and a warm started variant were presented theoretically and then applied to two demonstration problems. Kriging based scaling models are able to incorporate all design points that have been evaluated either in the design process or beforehand. The inclusion of more data points allows for a more global scaling function, as compared to the Taylor series based methods, which can decrease the number of variable fidelity iterations needed for convergence. In design cases where little or no information is
available before the optimization process, kriging models are initially inefficient. To improve convergence rates in these situations, a warm started kriging based scaling method was introduced. The warm started method starts the design process using a Taylor series based scaling function; after a number of data points have been evaluated, a kriging scaling function is then used. Both kriging-based scaling method were demonstrated using a two-dimensional analytic problem and a high-lift airfoil design problem.

The results of both demonstration problems indicated that the warm-started kriging based method can save substantial computational cost by lowering the number of high fidelity function calls required for optimization. In the analytic problem the warm-started method used 19% fewer high fidelity function calls on average for the various scaling methods. For the high-lift example the warm-started method reduced the number of expensive function calls by 60% using first order information and by an average of over 30% for the different second order scaling methods.

It was demonstrated that the adaptive hybrid method has the ability to adjust to an appropriate weighting between the additive and multiplicative methods so as to eliminate the need to determine \textit{a priori} which method should be used. The hybrid method did not improve performance over using either the multiplicative or additive scaling methods; it just alleviated the need for the designer to select the appropriate method. The adaptive hybrid method, therefore, can be used to save computational resources prior to performing the optimization.
A few drawbacks were identified with the kriging scaling models. They tended to increase the number of low fidelity function calls, which would be an issue if the low fidelity model required significant computational time. More work is needed to identify why this occurs and possible corrections. Also, the kriging models may be expensive to build and update after each iteration, depending on the number of design variables and sample points used; though, this cost may be insignificant, depending on the relative cost of the high fidelity model. A possible scaling limitation is that the adaptive hybrid method requires more memory to store both the additive and multiplicative models. With sufficient memory, the extra processing times for handling both models is offset by the savings in high fidelity function calls.
Kriging models have mostly served as metamodels in the optimization field. In Chapter 4 kriging models were used in a different manner to aid the variable fidelity optimization algorithm in reducing the cost of finding optima of computationally expensive simulations. Specifically, kriging-based scaling functions were used to match low fidelity models to high fidelity models so that the low fidelity model, augmented with the scaling function, would approximate the high fidelity model. The kriging-based scaling allows the approximation to be more global as compared to the Taylor series based methods, presented in earlier chapters, that use only local information. The kriging scaling approach combined with the trust region management framework provably converges to the solution of the higher fidelity model with the intent of saving computational resources.

Kriging is typically used as an interpolation process; however, there are a number of model parameters that must be chosen to control what effect nearby sample points have on the prediction of intermediate values. The selection of these parameters can be costly and depends on the dimensionality of the problem as well as the number of sample points used to construct the kriging model. There are two main methods of finding these parameters\textsuperscript{102}: cross validation and
maximum likelihood estimation (MLE). In this work only the MLE method is
used because it provides an estimate of the variance and the likelihood gives a
measure of the model fitness. As its name implies, the MLE method involves
an optimization to find the model parameters that maximize the likelihood es-
timate. The optimization can be performed with either a local optimizer, for
example gradient-based or pattern search methods, or using a global stochastic
optimization scheme. Ripley\[120\], Warnes and Ripley\[153\], and Martin et al.\[99\]
discuss problems with this optimization. They state two main difficulties: the
function is prone to have multiple maxima and the function may have long,
almost flat ridges of near optimal values. To overcome these difficulties Martin
and Simpson\[100\] suggested the use of simulated annealing, which is a monte
carlo global optimization method. Simulated annealing, however, is much more
expensive than a local-based method. Is this extra expense to possibly build a
better kriging model worthwhile in terms of the upper level optimization scheme
(i.e., variable fidelity methods) in which it is used? This tradeoff is addressed in
this chapter.

Because of the sequential nature of the variable fidelity optimization process, the
kriging models must be rebuilt, including new information at each iteration. This
process of rebuilding the kriging model by refitting the model parameters can be
quite costly, as shown in Figure 5.1. The figure shows the exponential cost trend
of constructing a kriging model with increasing numbers of samples and design
variables; the CPU time on the vertical axis should be used as a relative scale due
to the wide variation in computer processing speeds. Figure 5.1a shows the trends
for constructing a kriging model with fixed model parameters, while Figure 5.1b
accounts for optimizing the model parameters using a pattern search optimizer. For large problems with many design variables or samples, the cost of rebuilding the kriging model could reduce or outweigh the savings of using such methods. When the kriging models are rebuilt a maximum likelihood estimator is used to find optimal parameter values. However, the interpolative nature of kriging does not depend on the model parameters used; therefore, the kriging models may not need to be rebuilt, parameters refit, each time a new data point is added. In this

Figure 5.1. Relative cost of building kriging models with different numbers of samples and design variables.
chapter two metamodel update management schemes (MUMS) are proposed to determine when the kriging model parameters should be updated; the methods are based on the ratio of successive likelihood values (L-MUMS) and on using the trust region ratio value (TR-MUMS). This research also determines if such update management schemes can predict whether the kriging model parameters, when updated, should be found using a local optimization update or with a more expensive but more robust stochastic method.

This chapter first gives the different local and global optimization approaches used to fit the kriging model parameters (Section 5.1). Then a description of the update schemes is provided (Section 5.2), followed by the numerical procedure for evaluating them (Section 5.3). The update strategies are then demonstrated on an internal combustion engine design problem and a control-augmented structural design problem (Section 5.4).

5.1 Optimization Methods for Finding the Kriging Model Parameters

This section discusses three optimizers that can be used in finding the optimal kriging model parameters. The first two, Quasi-Newton and pattern search, are local optimizers, while the third, adaptive simulated annealing, is a global optimizer.

5.1.1 Constructing the Kriging Model Using a Quasi-Newton Method

To solve the MLE problem in Equation (4.11), a gradient-based or local optimization approach can be used. The main benefit of using such methods over
stochastic ones, is that they typically require many fewer function calls to converge. These methods make no claim or attempt at finding global solutions, only local optima, and have difficulty in large flat regions that may be occur in the likelihood function.

For this study a Quasi-Newton Broyden-Fletcher-Goldfarb-Shanno (BFGS) method is used in the form of MATLAB’s \textit{fminunc} function. Quasi-Newton methods successively minimize a second-order model of the function of the form

\[
\text{Minimize } \frac{1}{2} x^T \tilde{H} x + c^T x + b,
\]

where \( \tilde{H} \) is the current estimate of the Hessian. This estimate of the Hessian is approximated using successive gradient information, calculated via finite differencing, using the BFGS update:

\[
\tilde{H}_{k+1} = \tilde{H}_k + \frac{q_k q_k^T}{q_k^T s_k} - \frac{\tilde{H}_k^T s_k s_k^T \tilde{H}_k}{s_k^T \tilde{H}_k s_k},
\]

where

\[
s_k = x_{k+1} - x_k,
\]

\[
q_k = \nabla f(x_{k+1}) - \nabla f(x_k).
\]

5.1.2 Constructing the Kriging Model Using Pattern Search

In the DACE toolbox for MATLAB developed by Lophaven et al. a pattern search method is used to find the optimal kriging parameters. They specifically use the Hooke and Jeeves method, as described by Kowalik and Osborne.
using relative changes in the parameters instead of absolute changes. While this method does not use gradient information directly, it is still a local optimization method. Pattern search methods tend to be more suited for highly nonlinear or discontinuous functions, which may give this method an edge over a gradient-based approach, but tend to be less efficient when the function is first-order continuous. The pattern search will be compared to the gradient-based method (Quasi-Newton) and the simulated annealing approach, described next, in the numerical experiments.

5.1.3 Constructing the Kriging Model Using Adaptive Simulated Annealing

In order to provide a more robust optimization of the likelihood function than the gradient-based method, given the multimodal and long near-optimal ridge features of the likelihood function¹⁰², adaptive simulated annealing (ASA) is used. Adaptive simulated annealing was developed by Ingber⁷⁴,⁷⁵,⁷⁶,⁷⁷ and is a more efficient and more robust implementation of simulated annealing. Simulated annealing generalizes the Metropolis Monte Carlo integration algorithm¹⁰⁶ by including a temperature schedule for efficient searching by Kirkpatrick et al.⁸². Other researchers have also independently developed a similar method³¹,¹²⁰. Adaptive simulated annealing differs in the fact that typically fixed algorithmic parameters are allowed to adapt to each problem.

The metallurgical annealing process is used to remove stress from hardened metal or glass parts making them less brittle and less prone to failure. The reheating process and slow cooling associated with annealing is mimicked in the simulated
annealing to allow the optimizer to escape being trapped by local optima, traditionally a minima. This is done by using a quenching parameter, $T$, which is analogous to temperature and slowly reducing this value. Metropolis noted a larger probability of high energy exists at higher temperatures than at lower temperatures. Therefore, when the temperature is high, the method allows for larger jumps and increases in the objective function. As the temperature cools smaller jumps are allowed, and there is a smaller probability of allowing the objective function to increase. Simulated annealing uses the physical law given by Boltzmann’s probability distribution function,

$$p(\Delta E) = e^{-\frac{\Delta E}{kT}},$$

(5.5)

in determining when to accept functional values that are worse than the current point. In Equation (5.5) $k$ is the Boltzmann constant typically taken as one. The algorithm accepts all steps when the function is improved and accepts an increase with a probability of $p(\Delta f)$ using a random number generator. For more details Belegundu\cite{22} presents a good overview of the method, while Robert\cite{127} provides more background; for implementation details as used in this work see Ingber\cite{77}.

Simulated annealing, while more robust in finding the solution to the MLE problem, is also much more expensive in terms of function calls as it is a Monte Carlo method\cite{139}. The expense of each function call in constructing the kriging model involves finding the inverse and/or determinant of the correlation matrix.
5.1.4 Comparison Between Optimizers When Building a Kriging Model: 1D Example

Using a simple one-dimensional function allows for comparing the kriging models produced with the different optimizers when solving the MLE problem. The sine function is used as the target function to be approximated. The kriging models are built using four sample points, \( x = \{1.0, 1.05, 3.0, 3.05\} \) and \( y \) is the set or responses of \( f(x) = \sin(x) \). The initial starting point was taken to be \( \theta_0 = 5.5 \). This is not a very good initial guess given the scale of the problem; however, this is just a demonstration of what could happen in a larger problem where little is known about choosing a good initial point. Figure 5.2 shows the likelihood function given the set of sample points. The figure shows the likelihood is very flat in the range \( \theta = [1.5, 5.5] \) with a local maximum near \( \theta = 3 \) and a global maximum near \( \theta = 0.4 \).

The resulting kriging models constructed using no optimization and the three different optimizers are shown in Figure 5.3. The model that used no optimization produced the worst approximation, as expected. The two local optimization methods, Quasi-Newton and pattern search, produced very similar approximations. The best approximation of the sine function was produced by using the ASA optimizer.

In Table 5.1 the resulting value of \( \theta \) from each optimization is given along with the final likelihood value and the number of likelihood evaluations needed for convergence (L Evals). The results in the table show that the two local optimizers converged to the local maxima, which is why their kriging models were similar.
The ASA method converged to the global solution, giving a better approximation to the sine function. However, the ASA method used orders of magnitude more likelihood evaluations to converge, so the better approximation was obtained at a much higher computational cost. The Quasi-Newton and pattern search had similar numbers of likelihood evaluations. The Quasi-Newton optimizer required a few more evaluations because it used finite differencing to determine needed gradient information.

In this example if the starting point was a little better, for example around $\theta = 1$, then all methods would have converged to the same point, the global optima. The relative number of function evaluations would have still been roughly the same; therefore, one would wonder why ASA should be used. This observation raises an interesting point: can it somehow be gauged whether or not an initial start point
Figure 5.3. One-dimensional example comparing the kriging models produced by the different optimization methods.

for $\theta$ is close to the global optima? Then it would be possible to choose either a local optimizer or a global one to solve the MLE problem in order to construct the best approximation using the least amount of likelihood evaluations. This idea could be used in design algorithms where kriging models are built sequentially, as in variable fidelity optimization, where only a few points are added to the sample space each time the kriging model is reconstructed. It is also possible that no optimization is needed if the best $\theta$ values do not change much when adding a few more samples. More rigorous frameworks are proposed in the next section drawing from these observations. These frameworks are then evaluated using two design problems.
### TABLE 5.1
OPTIMIZATION RESULTS FOR THE ONE DIMENSIONAL EXAMPLE.

| Optimizer     | $\theta$ | $L(\theta|y)$ | Evals |
|---------------|----------|----------------|-------|
| None          | 5.50     | 0.4894         | 1     |
| Pattern Search| 3.09     | 0.5328         | 9     |
| Quasi-Newton  | 2.97     | 0.5329         | 20    |
| ASA           | 0.35     | 0.7190         | 364   |

#### 5.2 Metamodel Update Management Strategies For Sequentially Building Kriging Models

This section discusses the update strategies used to reduce the cost of having to optimize the kriging model parameters at each iteration of sequential approximation optimization (SAO) method like-variable fidelity optimization. Both methods measure, using different indicators, how good a kriging model is and if it should be updated or not after new samples are included. The first method, L-MUMS, uses the likelihood value which is measured directly from the kriging model itself. The second method, TR-MUMS, uses the trust region ratio value which gives a measure of how well the approximation matches the true model.
5.2.1 L-MUMS

The likelihood value obtained from Equation (4.7) when constructing a kriging model gives a measure of how likely it is that the values used for the model parameters, $\theta$, are the best values. To compare different kriging models, a ratio of their likelihood values can be used. If, for instance, the optimal model parameters are found for a set of data samples and then later a few more data samples are included, the new likelihood could be compared to the previous value to determine if the same model parameters can be used. This assumes that the inclusion of new data points does not significantly change the likelihood function.

To save computational cost of optimizing the kriging model parameters for each iteration of the variable fidelity framework a likelihood ratio of the following form is proposed:

$$\rho_n = \frac{L_n(\theta_n|y_n)}{L_n(\theta_n|y_{n-1})}, \quad (5.6)$$

where $y_{n-1}$ is the data set of the previous kriging model update and $y_n$ is the current data set. If the likelihood ratio is relatively near unity then the model parameters do not need to be updated, but if the ratio indicates a small change in likelihood a local optimization method starting from the current model parameters could easily move to the new optima relatively inexpensively. On the other hand, if the likelihood ratio shows a large change in likelihoods after the new points have been included, then a stochastic optimizer (e.g., simulated annealing) might be required to account for the multi-modal behavior of the likelihood function. This updating management scheme can be mathematically stated as:
\[ K_n = \begin{cases} 
\text{global} : & \varrho_n \leq \frac{1}{A_1} \lor A_1 < \varrho_n \\
\text{local} : & \frac{1}{A_1} < \varrho_n < \frac{1}{A_2} \lor A_2 < \varrho_n < A_1 \\
\text{none} : & \frac{1}{A_2} < \varrho_n < A_2 
\end{cases}, \quad (5.7) \]

where \( K_n \) is the optimization scheme used to determine the kriging model parameters for the \( n \)th iteration. This scheme has only two parameters which need to be tuned, \( A_1 \) and \( A_1 \). For the first iteration the kriging model parameters are found using any optimization approach.

### 5.2.2 TR-MUMS

An alternative approach to managing the updating of the kriging model parameters is to use the trust region ratio, \( \rho_n \) defined in Equation (2.5), that is already calculated in variable fidelity optimization as it is in most SAO methods. The trust region ratio provides a measure of how well the approximation represents the true model. It therefore can be used to estimate when the kriging model parameters need to be updated — when the kriging model is producing a poor approximation. Unlike the likelihood ratio this method only works after a bad approximation is produced and does nothing to determine if a kriging model is poor before it is used. However, this approach is more physically intuitive and is simple. The update scheme, in its most general form is as follows:

\[ K_{n+1} = \begin{cases} 
\text{global} : & \rho_n \leq A_1 \\
\text{local} : & A_1 < \rho_n < A_2 \\
\text{none} : & A_2 \leq \rho_n \leq A_3 
\end{cases}. \quad (5.8) \]
The parameters $A_1$, $A_2$, and $A_3$ do not necessarily have to correspond to the values used in updating the trust region size in Equation (2.6). This update scheme uses a global optimizer when the trust region ratio is far from one, a local optimizer when the ratio is moderately close to one, and no update when the ratio is near one. A simpler version of this scheme is one that only updates using a single optimizer. This reduces to a simpler scheme given below:

$$K_{n+1} = \begin{cases} 
\text{update} : & \rho_n \leq A_1 \\
\text{none} : & \rho_n > A_1 
\end{cases} \quad (5.9)$$

5.3 Numerical Experimental Procedure and Demonstration Problems

This section describes the numerical procedure used for evaluating the kriging model updating strategies. The goal is to determine if such updating strategies can reduce the cost of building the kriging models over the course of the optimization process without unduely increasing the number of iterations and high fidelity function calls required for convergence. Two engineering design problems were used to study the kriging update strategies: an internal combustion engine sizing problem and a control-augmented structure design problem.

Three main steps were used to evaluate the management of updating the kriging model parameters. First, the test problems were optimized using the variable fidelity method in which the kriging model parameters were updated at each iteration; this was repeated using the various likelihood optimization methods. Second, the variable fidelity framework was run using the the same kriging model parameters for the entire optimization process. The fixed values for these parameters corresponded to using each of the likelihood optimizers after the first
iteration. These results allow for a comparison between always and never updating the kriging model and to show how this affects the convergence of the variable fidelity algorithm. Finally, the different MUMS were used and compared to the first two studies.

The two demonstration problems are described in the next two sections. Quadratic response surfaces were used as the low fidelity objectives and constraints for both problems. The were generated using a high fidelity Latin hypercube sampling using $m^2$ data points. The computational costs of evaluating of the high and low fidelity models for the engine design problem were comparable and are used solely to evaluate the savings of the kriging update schemes. For the control-augmented structure problem the high fidelity model required 150 times the computational cost of the low fidelity model to evaluate.

5.3.1 Internal Combustion Engine Design

In this problem the geometry for a flat head internal combustion chamber is sought to provide maximal specific power. The design must also satisfy a number of constraints including packaging, fuel economy, and knock limitations. The problem was originally posed by the Ford Motor Corporation, and a robust variation of the problem was solved by McAllister and Simpson. The engine analysis parameters were determined by Ford’s Engine Assessment Model (ESA). A schematic for the flat head combustion chamber is shown in Figure 5.4. The design variables for this problem are the cylinder bore $b$, the compression ratio $c_r$, exhaust value diameter $d_E$, intake value diameter $d_I$, and the revolutions per
minute at peak power $w$. The design problem is mathematically posed below.

\begin{align}
\text{minimize} \quad f &= K_0\left(\frac{\rho Q}{A_f} \eta_V - FMEP\right)w \\
\text{subject to} \quad & K_1 N_c b - L_1 \leq 0, \quad \text{[min bore wall thick.]} \quad (5.11) \\
& \sqrt{4K_5 V_{\pi N_c L_2}} - b \leq 0, \quad \text{[max engine height]} \quad (5.12) \\
& d_I + d_E - K_3 b \leq 0, \quad \text{[valve structure]} \quad (5.13) \\
& K_4 d_I - d_E \leq 0, \quad \text{[min valve diam. ratio]} \quad (5.14) \\
& d_E - K_5 d_I \leq 0, \quad \text{[max valve diam. ratio]} \quad (5.15) \\
& 9.428 \times 10^{-5} \frac{4V}{\pi N_c} \frac{w}{d_I^2} - K_6 C_s \leq 0, \quad \text{[max Mach index]} \quad (5.16) \\
& c_r - 13.2 + 0.045 b \leq 0, \quad \text{[knock-limited ratio]} \quad (5.17) \\
& w - K_7 \leq 0, \quad \text{[max torq. conv. rpm]} \quad (5.18) \\
& 3.6 \times 10^6 - K_8 Q \eta_{tw} \leq 0. \quad \text{[fuel economy]} \quad (5.19)
\end{align}

The thermal efficiency, $\eta_t$, volumetric efficiency, $\eta_v$, and thermal at partial loading efficiency, $\eta_{tw}$, are all functions of the design variables and are given in the original paper[151]. The original paper also includes expressions for the friction mean effective pressure (FMEP), density of the inlet charge $\rho$, lower heating value $Q$, air-to-fuel ratio $A_f$, number of cylinders $N_c$, displacement volume $V$, port discharge coefficient $C_s$, and the parameters $K_i, i = \{0...8\}$ and $L_i, i = \{1, 2\}$. The starting and optimal designs are given in Table 5.2 along with their respective objective function values.
5.3.2 Control-Augmented Structure Design

The control-augmented structure design problem shown in Figure 5.5 was introduced by Sobieszczanski-Sobieski et al.\cite{143}. The problem comprises a total of 11 design variables and 43 states. The physical problem consists of a cantilever beam subjected to static loads along the beam and to a dynamic excitation force applied at the free end. Two sets of actuators are placed at the free end of the beam to control both the lateral and rotational displacement.

The system analysis is comprised of two coupled contributing analyses as shown in Figure 5.6. The structures subsystem, CA\textsubscript{s}, consists of a finite element model of the beam where the natural frequencies and modes of the cantilever beam are computed. CA\textsubscript{s} requires, in addition to the characteristics of the beam, the weight
TABLE 5.2
STARTING AND OPTIMUM DESIGNS FOR THE INTERNAL COMBUSTION ENGINE DESIGN PROBLEM.

<table>
<thead>
<tr>
<th>Design Variable</th>
<th>Starting Design</th>
<th>Optimum Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>( b ) (mm)</td>
<td>75</td>
<td>83.33</td>
</tr>
<tr>
<td>( c_r ) (L/L)</td>
<td>6.42</td>
<td>9.45</td>
</tr>
<tr>
<td>( d_E ) (mm)</td>
<td>26</td>
<td>30.99</td>
</tr>
<tr>
<td>( d_I ) (mm)</td>
<td>39</td>
<td>37.34</td>
</tr>
<tr>
<td>( w ) (rpm)</td>
<td>7500</td>
<td>6070</td>
</tr>
<tr>
<td>( f ) (kW/L)</td>
<td>30.28</td>
<td>55.67</td>
</tr>
</tbody>
</table>

of the control system as input. The weight of the control system is calculated in the controls CA, CA\(_c\). The weight of the control system is a function of the dynamic displacements and rotations of the free end of the beam. These dynamic displacements and rotations are functions of the natural frequencies and modes obtained in the structures CA, thus subjecting these CAs to coupling.

The objective of the optimization is to minimize the total weight of the system \( W_t \), composed of the weight of the beam \( W_s \) plus the weight of the control system \( W_c \). The minimization is subjected to seven constraints on the static stresses, lateral and rotational displacements, natural frequencies and dynamic lateral and rotational displacements at the free end of the beam. The problem is posed as:
minimize \[ W_t = W_s + W_c \]
subject to \[ 1 - \frac{dt}{dt_a} \geq 0, \]
\[ 1 - \frac{dr}{dr_a} \geq 0, \]
\[ \frac{\omega_1}{\omega_{1a}} - 1 \geq 0, \]
\[ \frac{\omega_2}{\omega_{2a}} - 1 \geq 0, \]
\[ 1 - \frac{\sigma}{\sigma_a} \geq 0, \]
\[ 1 - \frac{ddl}{ddl_a} \geq 0, \]
\[ 1 - \frac{ddr}{ddr_a} \geq 0, \]
### TABLE 5.3
STARTING AND OPTIMUM DESIGNS FOR THE CONTROLS-AUGMENTED STRUCTURE PROBLEM.

<table>
<thead>
<tr>
<th>Design Variable</th>
<th>Starting Design</th>
<th>Optimum Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_1 \ (\text{in})$</td>
<td>10.0</td>
<td>3.0</td>
</tr>
<tr>
<td>$b_2 \ (\text{in})$</td>
<td>10.0</td>
<td>3.0</td>
</tr>
<tr>
<td>$b_3 \ (\text{in})$</td>
<td>10.0</td>
<td>3.0</td>
</tr>
<tr>
<td>$b_4 \ (\text{in})$</td>
<td>10.0</td>
<td>3.0</td>
</tr>
<tr>
<td>$b_5 \ (\text{in})$</td>
<td>10.0</td>
<td>3.0</td>
</tr>
<tr>
<td>$h_1 \ (\text{in})$</td>
<td>10.0</td>
<td>13.85</td>
</tr>
<tr>
<td>$h_2 \ (\text{in})$</td>
<td>10.0</td>
<td>11.96</td>
</tr>
<tr>
<td>$h_3 \ (\text{in})$</td>
<td>10.0</td>
<td>9.78</td>
</tr>
<tr>
<td>$h_4 \ (\text{in})$</td>
<td>10.0</td>
<td>7.06</td>
</tr>
<tr>
<td>$h_5 \ (\text{in})$</td>
<td>10.0</td>
<td>3.75</td>
</tr>
<tr>
<td>$c$</td>
<td>0.01</td>
<td>0.06</td>
</tr>
</tbody>
</table>

where $dl$ is the static lateral displacement, $dr$ is the static rotational displacement, $ddl$ is the dynamic lateral displacement, $ddr$ is the dynamic rotational displacement, $\omega_1$ is the first natural frequency, $\omega_2$ is the second natural frequency, and $\sigma$ is the static stress. The subscript $a$ stands for the allowed value. The optimum for this problem is depicted in Table 5.3. The minimum weight, $W = 1493.6 \text{ lbs}$ occurs where 6 design variables are at their bounds.
5.4 Results

The results found from optimizing the two demonstration problems using the numerical procedure outlined in the previous section are given in this section. It was observed that when using L-MUMS for problems with nonlinear design spaces larger than a couple of design variables, the magnitude of the likelihood fluctuated by many orders of magnitude when just a few additional samples were added; this violated the assumption from which the update method was derived. This caused the kriging model to be updated every iteration. It was, therefore, concluded that the likelihood ratio was not a good indicator of when to update the kriging model parameters. Only the TR-MUMS results are given in the numerical studies for this reason.

5.4.1 Internal Combustion Engine Design Results

A summary of all of the results of optimizing the combustion chamber of an internal engine are given in Table 5.4. The number of iterations, high fidelity function calls, low fidelity function calls, likelihood evaluations, and the number of times the kriging model parameters were updated are all summarized in the table.

The first section of Table 5.4 shows the results for the variable fidelity optimization when the kriging model parameters were updated each iteration. For comparison, the number of function calls required to optimize the problem using a standard SQP solver is also given at the bottom of the table, and for each case the variable fidelity optimization requires fewer high fidelity function calls. The results show
that updating the kriging models using ASA actually resulted in a higher number of high fidelity function calls. This suggests that the highest likelihood value may not always produce the best approximation. The number of low fidelity function calls was also higher using ASA. The largest difference between the results from updating the kriging models each iteration is the number of likelihood evaluations required; ASA required two orders of magnitude more evaluations than did the Quasi-Newton optimizer, which in turn required another order of magnitude more evaluations than did the pattern search approach.

The next study repeated the same problem setup but used the value of the kriging model parameters found in the first iteration for the entire optimization process. These results are given in the second section of Table 5.4. As expected, the number of likelihood evaluations for each method significantly decreased, but still retained the same relative scale between the different optimizers. Unexpectedly, these results are not worse than when the kriging models were updated after each iteration. In some cases the results are even better. This provides evidence that sequential optimization techniques, such as VFO, may have low sensitivity to the values used in computing the kriging model and may not need to be updated as often. The pattern search method actually converged faster than in the previous trial, while all three methods required fewer low fidelity function calls. The unexpected result may be related to the theoretical findings of Lim et al. [90]; they noted the kriging best linear unbiased predictor (BLUP) has special asymptotic properties when the output of a computer model is highly correlated over the design space.
Table 5.4: Internal Combustion Engine Optimization Results.

<table>
<thead>
<tr>
<th>Optimizer</th>
<th>Iter</th>
<th>High Fn Calls</th>
<th>Low Fn Calls</th>
<th>L Evals</th>
<th>θ Updates</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASA</td>
<td>7</td>
<td>42</td>
<td>911</td>
<td>344,307</td>
<td>7</td>
</tr>
<tr>
<td>Quasi-Newton</td>
<td>6</td>
<td>31</td>
<td>308</td>
<td>8,668</td>
<td>6</td>
</tr>
<tr>
<td>Pattern Search</td>
<td>7</td>
<td>32</td>
<td>453</td>
<td>407</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASA</td>
<td>7</td>
<td>42</td>
<td>698</td>
<td>56,701</td>
<td>1</td>
</tr>
<tr>
<td>Quasi-Newton</td>
<td>6</td>
<td>31</td>
<td>247</td>
<td>347</td>
<td>1</td>
</tr>
<tr>
<td>Pattern Search</td>
<td>5</td>
<td>25</td>
<td>364</td>
<td>62</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASA</td>
<td>7</td>
<td>42</td>
<td>698</td>
<td>56,701</td>
<td>1</td>
</tr>
<tr>
<td>Quasi-Newton</td>
<td>5</td>
<td>25</td>
<td>274</td>
<td>2,852</td>
<td>2</td>
</tr>
<tr>
<td>Pattern Search</td>
<td>5</td>
<td>25</td>
<td>462</td>
<td>118</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TR-MUMS (θ updated when $\rho_n &lt; 0.25$)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASA</td>
<td>7</td>
<td>42</td>
<td>698</td>
<td>56,701</td>
<td>1</td>
</tr>
<tr>
<td>Quasi-Newton</td>
<td>5</td>
<td>25</td>
<td>274</td>
<td>2,852</td>
<td>2</td>
</tr>
<tr>
<td>Pattern Search</td>
<td>5</td>
<td>25</td>
<td>462</td>
<td>118</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standard single fidelity optimization</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQP</td>
<td>9</td>
<td>64</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
In the last trial the TR-MUMS was used. In this strategy the kriging model was
updated using different optimizers when the trust region ratio was below 0.25
\( \rho_n < 0.25 \). The results are given in the third section of Table 5.4. The results
show a decrease in the number of high fidelity function calls using Quasi-Newton
and the pattern search approaches as compared to the initial study. The pattern
search used the same number of high fidelity function calls as in the second
trial. Both of these methods updated the kriging model parameters one extra
time as compared to using the same kriging models for the entire optimization,
allowing for the improved performance; and as expected the number of likelihood
function evaluations increased accordingly but was still much less than updating
the parameters after each iteration. These results show that the TR-MUMS can
decrease the cost of variable fidelity optimization.

5.4.2 Control-Augmented Structure Design Results

The computational expense of the optimization process using the different krig-
ing model update schemes is more closely studied using the control-augmented
structures problem. The amount of computational time required was added to
the summary of results for this problem which is given in Table 5.5. A high
fidelity function call required 1.5s to evaluate, the low fidelity model took 0.01s
to evaluate, and to optimize the kriging models, on average, took 3s and 900s
for the pattern search and ASA approaches respectively. Also, the Quasi-Newton
optimizer was not used in this problem for finding the kriging models. This is
done without loss of generality because it is a local optimizer like the pattern
search and both had similar results in the previous problem relative to ASA.
The first section of Table 5.5 shows the results when the kriging model parameters were updated each iteration. The ASA approach used slightly fewer high fidelity function calls than did the pattern search, and it also used fewer low fidelity function calls as well. The main difference between the two optimizers was in the number of likelihood evaluations required. The simulated annealing method used over a million likelihood evaluations; three orders of magnitude more than the pattern search method. This huge difference of likelihood evaluations is the reason the execution time for using the simulated annealing was much higher, even higher than performing standard optimization on the high fidelity function alone. The ASA approach could have been less computationally expensive than the standard SQP optimization if the computational cost of the high fidelity model was much greater, as it would be in some engineering design problems.

The second section of Table 5.5 shows the results from using a single set of values for computing the kriging approximation throughout the optimization process. In this case the variable fidelity optimization process failed to converge to the optimal solution for both methods. The premature convergence was due to the fact that the trust region size became too small, which is an indication that the kriging approximation became unusable and needed to be updated. This result is contrary to the results found in the internal engine design problem, showing that there is a need for updating the kriging model parameters during the VFO process.

The TR-MUMS scheme for updating the kriging model parameters converged to the optimal solution using fewer than half of the number of kriging model updates.
### Table 5.5

**Control-Augmented Structure Optimization Results.**

<table>
<thead>
<tr>
<th>Optimizer</th>
<th>Iter</th>
<th>High Fn Calls</th>
<th>Low Fn Calls</th>
<th>L Evals</th>
<th>θ Updates</th>
<th>Wall Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASA</td>
<td>12</td>
<td>123</td>
<td>1,573</td>
<td>1,070,000</td>
<td>12</td>
<td>11,013</td>
</tr>
<tr>
<td>Pattern</td>
<td>16</td>
<td>127</td>
<td>2,093</td>
<td>1,594</td>
<td>16</td>
<td>259</td>
</tr>
<tr>
<td>ASA*</td>
<td>17</td>
<td>117</td>
<td>1,430</td>
<td>100,016</td>
<td>1</td>
<td>1,090</td>
</tr>
<tr>
<td>Pattern*</td>
<td>35</td>
<td>333</td>
<td>4,214</td>
<td>140</td>
<td>1</td>
<td>545</td>
</tr>
<tr>
<td>TR-MUMS (θ updated when $\rho_n &lt; 0.25$)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASA</td>
<td>14</td>
<td>125</td>
<td>1,894</td>
<td>410,009</td>
<td>5</td>
<td>4,706</td>
</tr>
<tr>
<td>Pattern</td>
<td>17</td>
<td>128</td>
<td>2,149</td>
<td>839</td>
<td>8</td>
<td>237</td>
</tr>
<tr>
<td>Standard single fidelity optimization</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQP</td>
<td>26</td>
<td>340</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>510</td>
</tr>
</tbody>
</table>

* Did not converge to the optimum; the trust region size became too small.
than updating the model parameters every iteration. The results are given in the third section of Table 5.5. The reduction in likelihood evaluations, however, came at a cost of slightly increasing the number of high fidelity and low fidelity function calls; at least for this problem. These results indicate that for problems in which the cost of updating the kriging model parameters is significantly high relative to a high fidelity function call, the trust region ratio can be used as a good indicator as to when the kriging parameters should be updated.

5.5 Summary and Conclusions

Optimizing the kriging model parameters can be computationally expensive, especially when the number of design variables and/or sample points used are large. This cost could outweigh the benefits of using kriging models as approximations in trust region managed sequential approximate optimization methods, such as variable fidelity optimization which attempt to decrease the computational cost of simulation-based design. In this chapter an overview of different ways of optimizing the kriging model parameters was given. It was pointed out that the likelihood function, which is the objective in determining the kriging model parameters, can be multi-modal and have long flat ridges making it tough for local optimization methods such as Quasi-Newton or pattern search to converge to the global optima. Global optimization approaches, like adaptive simulated annealing, have been used by researchers to address this problem. However, such methods required many orders of magnitude more likelihood evaluations to converge. Two metamodel update management schemes were introduced in this chapter to reduce the cost of using kriging models that are sequentially updated.
The first scheme, L-MUMS, used likelihood ratios to determine when to update the kriging model parameters. The second, TR-MUMS, used the trust region ratio value to determine when the kriging model was not doing an adequate job of approximating the true model and needed to be updated.

Two engineering design problems were solved to study the sensitivity of the variable fidelity optimization framework to different kriging model parameter updating schemes. The test problems included an internal engine combustion chamber sizing problem and a control-augmented structural design problem. It was found that the variable fidelity method was insensitive to what optimizer was used; in fact, the simulated annealing optimizer required many more likelihood evaluations and did not improve performance. Local optimization methods like Quasi-Newton and especially the pattern search method performed well with many fewer likelihood evaluations. It was also found that the likelihood ratio was not a good measure of when the kriging model parameters need to be updated. However, the trust region ratio was found to be a good indicator, and its use resulted in fewer kriging model parameter updates and a lower total cost of optimization.
CHAPTER 6
RELIABILITY BASED DESIGN USING VARIABLE FIDELITY OPTIMIZATION

In previous chapters the main focus was on refining the variable fidelity optimization method. More specifically, Chapters 2, 3, and 4 looked at new forms of the scaling function that scaled the low fidelity model to match the high fidelity model. Chapter 5 discussed possible ways to reduce the cost of constructing the kriging based scaling function that was presented in Chapter 4. This chapter describes a specialized application of VFO framework. The framework is extended to solve reliability based design problems; this application broadens the current use of variable fidelity methods to a different class of problems.

Designs produced by deterministic optimization are often on the boundary of one or more constraints. Such designs leave no room for uncertainties such as those that may arise out of manufacturing tolerances, material property variations, or the unpredictability of external forces and loadings which may result in catastrophic failure. To address this issue engineers have typically used worst case values for the uncertain parameters; this leads to unnecessarily conservative designs. These designs are more expensive, where expense is paramount in a competitive marketplace. Furthermore, to address unknown failure modes safety factors have been traditionally used to further increase the robustness of designs.
The approach of using worst case values is also quite heuristic in nature and doesn’t guarantee that a design meets a desired probability of failure specification. Reliability based design optimization (RBDO) methods have emerged to solve such problems. Designs obtained using RBDO ensure that the probability of failure due to parameter uncertainties is below a desired level.

Conventionally, researchers have formulated RBDO as a nested optimization problem referred to as the double-loop method. This formulation is computationally expensive because of the two levels of optimization required. The upper level optimization solves the standard design problem, while for each function call a number of reliability analysis are performed; these are the inner optimization problems. Solving such nested optimization problems can be cost prohibitive, especially for large scale high fidelity multidisciplinary systems. Moreover, the computational cost associated with RBDO grows exponentially as the number of random variables and the number of critical failure modes increase. To alleviate the high computational cost, researchers have developed sequential RBDO methods. However, such techniques may lead to premature convergence and, hence, yield spurious optimal designs. In the research by Agarwal et al.\[6, 9\], a new unilevel formulation for performing RBDO was developed. The formulation provided for improved robustness and provable convergence as compared to a unilevel variant given by Kuschel and Rackwitz[85]. In this new unilevel approach, the basic idea is to replace the inverse first order reliability method (FORM) by its first order Karush-Kuhn-Tucker (KKT) necessary optimality conditions at the upper level optimization. This unilevel method was shown to be computationally equivalent to the original nested optimization problem if the lower level optimiza-
tion problem is solved by satisfying the KKT necessary condition.

RBDO, in general, is still relatively expensive when compared to deterministic optimization that doesn’t account for design uncertainty. This expense has put limits on the types of problem to which it can be applied. For instance, high fidelity simulation models which require considerable computational cost may not be able to converge under design cycle time constraints. Attempts at reducing this cost have used approximation models such as kriging and correction response surfaces. These methods, however, don’t converge to the true optimal solution of the high fidelity system. In this chapter, variable fidelity methods are used in conjunction with the double-loop method to reduce the computational cost of obtaining a reliable design, while guaranteeing convergence to the high fidelity solution, provided a suite of fidelity models are available. Using the new scaling methods described in earlier chapters of this dissertation, the efficiency of variable fidelity methods has improved, providing for a framework for using lower fidelity models to reduce computational cost of design optimization. Recalling from previous chapters, the basic concept of variable fidelity optimization is to reduce the number of high fidelity function calls by using lower fidelity models and scaling functions that update the low fidelity model to match the higher fidelity result. The bulk of the computational time is used for evaluating the low fidelity model, while using limited high fidelity calls to ensure the scaling function is accurate.

In this chapter an overview of RBDO is given first, along with brief details of the double-loop method. Next the combined variable fidelity reliability based design
optimization (VF-RBDO) approach is presented. This approach is then applied to two design problems to demonstrate the computational savings. The problems include an analytic test problem and a higher-dimensional high-lift airfoil design problem.

6.1 Deterministic Design Optimization

In solving a deterministic design optimization problem, the designer seeks the optimum values of design variables for which a merit function is the minimum and the deterministic constraints are satisfied. A standard form of the deterministic design optimization problem is:

\[
\begin{align*}
\text{minimize} & \quad f(\mathbf{x}, \mathbf{p}) \\
\text{subject to} & \quad g_i^R(\mathbf{x}, \mathbf{p}) \geq 0, \quad i = 1, \ldots, N_R, \\
& \quad g_j^D(\mathbf{x}, \mathbf{p}) \geq 0, \quad j = 1, \ldots, N_D, \\
& \quad \mathbf{x}^l \leq \mathbf{x} \leq \mathbf{x}^u,
\end{align*}
\] (6.1)

where \(\mathbf{x}\) are the design variables and \(\mathbf{p}\) are the fixed parameters of the optimization problem. \(g_i^R\) is the \(i^{th}\) hard constraint that models the \(i^{th}\) critical failure mechanism of the system (e.g., stress, deflection, loads, etc). \(g_j^D\) is the \(j^{th}\) deterministic constraint due to other design considerations that are not affected by parameters or variables that could be uncertain. The design space is bounded by \(\mathbf{x}^l\) and \(\mathbf{x}^u\). The merit function and the constraints are explicit functions of \(\mathbf{x}\) and \(\mathbf{p}\).
A deterministic optimization formulation does not account for the uncertainties in the design variables and parameters. Optimized designs based on a deterministic formulation are usually associated with a high probability of failure because of the likely violation of certain hard constraints. This is particularly true if the hard constraints are active at the deterministic optimum solution. In today’s competitive marketplace, it is very important that the resulting designs are optimal as well as reliable. This is usually achieved by replacing a deterministic optimization formulation with a reliability based design optimization formulation, where the critical hard constraints are replaced with reliability constraints.

6.2 Reliability Based Design Optimization

In the last two decades, researchers have proposed a variety of frameworks for efficiently performing reliability based design optimization. A survey of the literature reveals that the various RBDO methods can be divided into two broad categories: double-loop methods and sequential methods. Both of these approaches make use of a reliability analysis technique to determine how reliable a given design is. These methods are explained in the following sections\(^1\).

Reliability based design methods tend to move optimal designs away from failure driven constraints so that the uncertainties lead to a much lower chance of failure. This concept is shown pictorially in Figure 6.1. In the figure, point A is a deterministic design that is located on a failure driven constraint. When uncertainty is considered this single design point is replaced by a probability distribution of

---

\(^1\)Section 6.2 was co-authored by Harish Agarwal, Gano et al.\[51\]
designs produced using the design parameters of point A. This figure shows that a good deal of this distribution about point A is in the failure domain. Point B with its respective probability distribution has a much lower chance of producing designs that fail. RBDO locates designs, like point B, that have a given measure of reliability while also trying to optimize an objective or cost function. In other words, RBDO methods attain the target reliably while minimizing the tradeoff from the objective.

Figure 6.1. Reliability based optimization produce designs away from failure driven constraints.

6.2.1 Double Loop Methods for RBDO

Traditionally, the reliability based optimization problem has been formulated as a double loop optimization problem. In a typical RBDO formulation, the critical
hard constraints from the deterministic formulation are replaced by reliability constraints, as in

\[
\begin{align*}
\text{minimize} \quad & f(x, p) \\
\text{subject to} \quad & g^{rc}(V, \eta) \geq 0, \quad (6.5) \\
& g^D_j(x, p) \geq 0, \quad j = 1, \ldots, N_D, \quad (6.6) \\
& x^l \leq x \leq x^u, \quad (6.7)
\end{align*}
\]

where \(g^{rc}\) are the reliability constraints. They are either constraints on probabilities of failure corresponding to each hard constraint or are a single constraint on the overall system’s probability of failure. In this research, only component failure modes are considered. It should be noted that the reliability constraints depend on the random variables \(V\) and limit state parameters \(\eta\). The distribution parameters of the random variables are obtained from the design variables \(x\) and the fixed parameters \(p\). (See the section on reliability analysis below.) \(g^{rc}\) can be formulated as

\[
g_i^{rc} = P_i - P^*_i, \quad i = 1, \ldots, N_R, \quad (6.9)
\]

where \(P_i\) is the failure probability of the hard constraint \(g^R_i\) at a given design, and \(P^*_i\) is the target allowable probability of failure for this failure mode. The probability of failure is usually estimated by employing standard reliability techniques. A brief description of standard reliability methods is given in the next section. It has to be noted that the RBDO formulation given above (Equations (6.5)-(6.8))
assumes that the violation of soft constraints due to variational uncertainties is permissible and can be traded off for more reliable designs. For practical problems, design robustness represented by the merit function, and the soft constraints could be a significant issue, one that would require the solution to a hybrid robustness and reliability based design optimization formulation.

6.2.2 Reliability Analysis

Reliability analysis is a tool to compute the reliability index or the probability of failure corresponding to a given failure mode or for the entire system [66]. The uncertainties are modeled as continuous random variables, \( V = (V_1, V_2, ..., V_n)^T \), with a known (or assumed) joint cumulative distribution function (CDF), \( F_V(v) \). The design variables, \( x \), consist of either distribution parameters \( \theta \) of the random variables \( V \), such as means, modes, standard deviations, and coefficients of variation, or deterministic parameters, also called limit state parameters, denoted by \( \eta \). The design parameters \( p \) consist of either the means, the modes, or any first order distribution quantities of certain random variables. Mathematically this can be represented by the statements:

\[
[p, x] = [\theta, \eta], \tag{6.10}
\]

\[ p \text{ is a subvector of } \theta. \tag{6.11} \]

Random variables can be consistently denoted as \( V(\theta) \), and the \( i^{th} \) failure mode can be denoted as \( g^R_i(V, \eta) \). In the following, \( v \) denotes a realization of the random variables \( V \), and the subscript \( i \) is dropped without loss of clarity. Letting
$g^R(v, \eta) \leq 0$ represent the failure domain, and $g^R(v, \eta) = 0$ be the so-called limit state function, the time-invariant probability of failure for the hard constraint is given by

$$P(\theta, \eta) = \int_{g^R(v, \eta) \leq 0} f_V(v) \, dv,$$  \hspace{1cm} (6.12)

where $f_V(v)$ is the joint probability density function (PDF) of $V$. It is usually impossible to find an analytical expression for the above integral. In standard reliability techniques, a probability distribution transformation $T: \mathbb{R}^n \rightarrow \mathbb{R}^n$ is usually employed. An arbitrary $n$-dimensional random vector $V = (V_1, V_2, ..., V_n)^T$ is mapped into an independent standard normal vector $U = (U_1, U_2, ..., U_n)^T$. This transformation is known as the Rosenblatt Transformation \cite{133}. This transformation is depicted in Figure 6.2. The standard normal random variables are characterized by a zero mean and unit variance. The limit state function in $U$-space can be obtained as $g^R(v, \eta) = g^R(T^{-1}(u), \eta) = G^R(u, \eta) = 0$. The failure domain in $U$-space is $G^R(u, \eta) \leq 0$. Equation (6.12) thus transforms to

$$P_i(\theta, \eta) = \int_{G^R(u, \eta) \leq 0} \phi_U(u) \, du,$$  \hspace{1cm} (6.13)

where $\phi_U(u)$ is the standard normal density. If the limit state function in $U$-space is affine, i.e., if $G^R(u, \eta) = \alpha^T u + \beta$, then an exact result for the probability of failure is $P_f = \Phi(-\frac{\beta}{\|\alpha\|})$, where $\Phi(\cdot)$ is the cumulative Gaussian distribution function. If the limit state function is close to being affine, i.e., if $G^R(u, \eta) \approx \alpha^T u + \beta$ with $\beta = -\alpha^T u^*$, where $u^*$ is the solution of the following optimization problem,
Figure 6.2. Transformation of the failure domain to a normalize space makes it possible to locate the MPP.

\[
\begin{align*}
\text{min } & \quad ||u|| \\
\text{subject to } & \quad G^R(u, \eta) = 0,
\end{align*}
\]

(6.14) (6.15)

then the first order estimate of the probability of failure is \(P_f = \Phi(-\beta ||\alpha||)\), where \(\alpha\) represents a normal to the manifold (6.15) at the solution point. The solution \(u^*\) of the above optimization problem, the so-called design point, \(\beta\)-point or the most probable point (MPP) of failure, defines the reliability index \(\beta_p = -\frac{\alpha^T u^*}{||\alpha||}\). This method of estimating the probability of failure is known as the first order reliability method (FORM) [66].

In the second order reliability method (SORM), the limit state function is approximated as a quadratic surface. A simple closed form solution for the probability computation using a second order approximation was given by Breitung[28] using the theory of asymptotic approximations as
\[ P_f(\theta, \eta) = \int_{G^R(u, \eta) \leq 0} \phi_U(u) \, du \]
\[ \approx \Phi(-\beta_p) \prod_{l=1}^{n-1} (1 - \kappa_l)^{-1/2}, \quad (6.16) \]

where \( \kappa_l \) is related to the principal curvatures of the limit state function at the minimum distance point \( u^* \), and \( \beta_p \) is the reliability index using FORM. Breitung\[28\] showed that the second-order probability estimate asymptotically approaches the first order estimate as \( \beta_p \) approaches infinity, if \( \beta_p \kappa_l \) remains constant.

The first order approximation, \( P_f \approx \Phi(-\beta_p) \), is sufficiently accurate for most practical cases. Thus, only first order approximations of the probability of failure are used in practice. Using the FORM estimate, the reliability constraints in Equation (6.9) can be written in terms of reliability indices as

\[ g_{rc}^i = \beta_i - \beta_{t_i}, \quad (6.17) \]

where \( \beta_i \) is the first order reliability index, and \( \beta_{t_i} = -\Phi^{-1}(P_{allow_i}) \) is the desired reliability index for the \( i^{th} \) hard constraint. When the reliability constraints are formulated as given in Equation (6.17), the approach is referred to as the reliability index approach (RIA).

It should be noted that the reliability analysis involves a probability distribution transformation, the search for the MPP, and the evaluation of the cumulative
Gaussian distribution function. To solve the FORM problem (Equations 6.14-6.15), various algorithms have been reported in the literature\cite{92}. The solution typically requires many system analysis evaluations. Moreover, there might be cases where the optimizer may fail to provide a solution to the FORM problem, especially when the limit state surface is far from the origin in $U$-space or when the case $G^R(u, \eta) = 0$ never occurs at a particular design variable setting.

In design automation it is not known \textit{a priori} what design points the upper level optimizer will visit; therefore, it is not known if the optimizer for the FORM problem will provide a consistent result. This problem was addressed recently by Padmanabhan and Batill\cite{114} by using a trust region algorithm for equality constrained problems. For cases when $G^R(u, \eta) = 0$ does not occur, the algorithm provided the best possible solution for the problem through

$$\min \|u\| \quad \text{subject to} \quad G^R(u, \eta) = c. \quad (6.18)$$

The reliability constraints formulated by the RIA are, therefore, not robust. RIA is usually more effective if the probabilistic constraint is violated, but it yields a singularity if the design has zero failure probability\cite{147}. To overcome this difficulty, Tu \textit{et al.}\cite{147} provided an improved formulation to solve the RBDO problem. In this method, known as the \textit{performance measure approach} (PMA), the reliability constraints are stated by an inverse formulation as

$$g^{rc}_i = G^R_{\beta_i}(u^{\ast}_{\beta_i} + \eta) \quad i = 1, ..., N_R. \quad (6.20)$$
\( \mathbf{u}_i^* \) is the solution to the inverse reliability analysis (IRA) optimization problem

\[
\begin{align*}
\text{min} & \quad G^R_i(\mathbf{u}, \eta) \\
\text{subject to} & \quad \|\mathbf{u}\| = \beta_t,
\end{align*}
\]

(6.21) (6.22)

where the optimum solution \( \mathbf{u}_i^{*\beta = \beta_t} \) corresponds to MPP in IRA of the \( i^{th} \) hard constraint. Solving RBDO by the PMA formulation is usually more efficient and robust than the RIA formulation where the reliability is evaluated directly. The efficiency lies in the fact that the search for the MPP of an inverse reliability problem is easier to solve than the search for the MPP corresponding to an actual reliability. The RIA and the PMA approaches for RBDO are essentially inverse of one another and would yield the same solution if the constraints are active at the optimum\[147]. If the constraint on the reliability index (as in the RIA formulation) or the constraint on the optimum value of the limit-state function (as in the PMA formulation) is not active at the solution, the reliable solution obtained from the two approaches might differ.

Similar RBDO formulations were independently developed by other researchers \[81, 122, 134\]. In these RBDO formulations, constraint (6.22) is considered as an inequality constraint (\( \|\mathbf{u}\| \leq \beta_t \)), which is a more robust way of handling the constraint on the reliability index. The major difference lies in the fact that in these papers’ semi-infinite optimization algorithms were employed to solve the RBDO problem. Semi-infinite optimization algorithms solve the inner optimization problem approximately. However, the overall RBDO is still a nested double-loop optimization procedure. As mentioned earlier, such formulations are computationally
intensive for problems where the function evaluations are expensive. Moreover, the formulation becomes impractical when the number of hard constraints increase, which is often the case in real-life design problems. To alleviate the computational cost associated with the nested formulation, sequential RBDO methods have been developed.

6.2.3 Sequential Methods for RBDO

Sequential RBDO methods include a variety of different approaches proposed by different researchers. Chen and Du\cite{34} developed a decoupled sequential probabilistic design methodology. In this framework, the deterministic optimization and the reliability assessment are decoupled from one another. During each cycle, a deterministic optimization problem is solved, followed by reliability assessment and a convergence check. Chen et al.\cite{35} also developed a sequential RBDO methodology that was recently generalized for non-normal distributions by Wang and Kodiyalam\cite{152} and extended for multidisciplinary systems by Agarwal et al.\cite{7}. In this methodology, the lower-level optimization is eliminated, and the MPP of failure corresponding to the probabilistic constraints is estimated implicitly by using a nonlinear transformation based on the direction cosines of the hard constraints at the mean values of the random variables. This methodology is shown to be extremely efficient. However, for highly nonlinear limit state functions, the estimate of the MPP of failure given by the nonlinear transformation might be very different from the actual MPP of failure, and the framework may fail to converge to the true solution. The drawback of sequential RBDO methodologies is that a local optimum cannot be guaranteed. Such methodologies can lead to spurious optimal designs.
It has been noted that the traditional reliability based optimization problem is a nested optimization problem. Solving such nested optimization problems for a large number of failure driven constraints and/or nondeterministic variables is extremely expensive. Researchers have developed sequential approaches to speed up the optimization process and to obtain a consistent reliability based design. To address the issue of obtaining spurious optimal designs, a new sequential optimization strategy for reliability based design is developed in Agarwal et al.\cite{4}.

At this point the ground work for doing RBDO has been described. The RBDO methods inherently have a much higher computational cost than doing deterministic optimization. In order to make RBDO computationally trackable for expensive problems, variable fidelity optimization methods are employed in this investigation to reduce the computational cost. The ability to use such methods assumes that a suite of fidelity models can be provided. In the next section variable fidelity methods are discussed.

6.3 Reliability Based Design Using Variable Fidelity Optimization

This section describes how the variable fidelity framework is used to reduce the computational expense of high fidelity reliability based design. This combined method will be henceforth referred to as variable fidelity reliability based design optimization (VF-RBDO).

Combining these two methods is a straightforward process; though, combining
the two methods is a novel approach to lower the cost of RBDO. The variable fidelity framework is setup to do deterministic as opposed to stochastic reliability optimization. In the reliability case the upper level constraints, which insure a specific level of reliability, involve a sub-optimization process of finding the MPP for each of the deterministic constraints. The objective function is not typically altered, unless a robust design formulation is desired. Therefore, the objective function and the upper level constraints can be included into the variable fidelity framework directly as long as there exist at least two levels of fidelity models used in computing the system responses. A diagram for combining these two methods is given in Figure 6.3.

Figure 6.3. VF-RBDO framework

Another emerging RBDO method is to remove the sub-optimization problems of finding the MPP for each constraint by instead including their first order Karush Kuhn Tucker conditions at the top level. By removing the nested optimization calls these methods have been shown to reduce the computational expense of
RBDO\cite{6,9}. However, these methods increase the number of design variables significantly while also making the design space itself much more complicated by including many equality constraints. This complication may be augmented by using constraint relaxation methods such as homotopy\cite{8} or by using approximations to solve the infeasible trust region problem\cite{116}. Furthermore, unilevel methods require second order information which can be approximated. Because second order information is typically not available, using approximate second order information may hinder the convergence properties of the variable fidelity method. Combining the unilevel RBDO methods and the variable fidelity method would follow the same procedure as described above.

6.4 Numerical Implementation Studies

To demonstrate the savings of using the VF-RBDO method it was applied to two problems. The first problem is a specific analytic problem referred to as the Barnes Problem. This problem will provide verification of the method and is two-dimensional for easier solution visualization. The second problem is an airfoil shape optimization problem that is subject to an uncertain Mach number and angle of attack. This problem is similar to problems used by Padula et al.\cite{71,89,115} for robust design. To compute the aerodynamic forces on the airfoils, various fidelity computational fluid dynamics (CFD) simulations are used.

6.4.1 Modified Barnes Problem

In order to help visualize the VF-RBDO methodology an analytic two-dimensional problem is solved first. The problem is known as the Barnes Problem as it was
originally formulated in his master’s thesis\textsuperscript{21}. It is a highly nonlinear problem which makes it challenging to solve, even though it has only two design variables. The original formulation of the problem was deterministic. Agarwal and Renaud\textsuperscript{5} recast the problem as a reliability problem and renamed this formulation the Modified Barnes Problem. This modified version is used as the high fidelity model. The problem has four random variables, \( V \), which are statistically independent and normally distributed. The means and standard deviations for these random variables are given in the problem formulation below. Two of the three constraints for this problem are failure driven, \( g^R \). The remaining constraint, \( g^D \), is deterministic. The optimization problem is posed as follows.

\[
\min_{x_1, x_2} f(x) = a_1 + a_2 x_1 + a_3 x_2^2 + a_4 x_1^3 + a_5 x_1^4 + a_6 x_2 + a_7 x_1 x_2 + a_8 x_1^2 x_2 + a_9 x_1^2 x_2 + a_{10} x_1 x_2 + a_{11} x_2^2 + a_{12} x_2^3 + a_{13} x_2^4 + \frac{a_{14}}{x_1 + 1} + a_{15} x_1 x_2^2 + a_{16} x_1 x_2^2 + a_{17} x_1 x_2^2 + a_{18} x_1 x_2^3 + a_{19} x_2 x_2^3 + a_{20} e^{a_{21} x_1 x_2}
\]

subject to:
\[
\begin{align*}
g^R_1 &= \frac{x_1 x_2}{v_1} - v_2 \geq 0 \\
g^R_2 &= \frac{x_2}{v_3} + \frac{x_2^2}{v_4} \geq 0 \\
g^D &= \left(\frac{x_2}{50} - 1\right)^2 - \left(\frac{x_1}{500} - 0.11\right) \geq 0
\end{align*}
\]

and
\[
\begin{align*}
0 &\leq x_1 \leq 75 \\
0 &\leq x_2 \leq 65
\end{align*}
\]

where \( v_1 = N(700, 1), \ v_2 = N(1, 0.3), \ v_3 = N(5, 1), \ v_4 = N(25, 0.3) \)

The coefficients of the objective function are given in Table 6.1. The RBDO formulation for this problem is
TABLE 6.1
COEFFICIENTS FOR THE BARNES PROBLEM.

| \(a_1\) | 7.5196E1 | \(a_2\) | -3.8112E0 | \(a_3\) | 1.2694E-1 | \(a_4\) | -2.0567E-3 | \(a_5\) | 1.0345E-5 |
| \(a_6\) | -6.8306E0 | \(a_7\) | 3.0234E-2 | \(a_8\) | -1.28134E-3 | \(a_9\) | 3.5256E-5 | \(a_{10}\) | -2.2667E-7 |
| \(a_{11}\) | 2.5645E-1 | \(a_{12}\) | -3.4604E-3 | \(a_{13}\) | 1.3514E-5 | \(a_{14}\) | -2.8106E1 | \(a_{15}\) | -5.2375E-6 |
| \(a_{16}\) | -6.3000E-8 | \(a_{17}\) | 7.0000E-10 | \(a_{18}\) | 3.4054E-4 | \(a_{19}\) | -1.6638E-6 | \(a_{20}\) | -2.8673E0 |
| \(a_{21}\) | 5.0000E-4 |

minimize \(f(x)\)
subject to: \(g^c_i = \beta_i - 3 \geq 0, \ i = 1, 2\)
\(g^D \geq 0\)
and
\(0 \leq x_1 \leq 75\)
\(0 \leq x_2 \leq 65\).

A low fidelity version of this problem is created by making a few changes to the problem which significantly alter the design space. In the objective function two coefficients, \(a_5\) and \(a_{21}\), are set to zero, reducing the nonlinearity of the model. Also the first two constraints are altered, including a change of three mean values of the random variables. These modifications are summarized below.

\(a_{5\text{low}} = 0, \ a_{21\text{low}} = 0\)

\(g^R_{1\text{low}} = \frac{x_1 x_2}{v_{1\text{low}}} - v_{2\text{low}} \geq 0, \ g^R_{2\text{low}} = \frac{x_2}{v_3} + \frac{x_1}{v^2_{4\text{low}}} \geq 0\)

\(v_{1\text{low}} = N(750, 1), \ v_{2\text{low}} = N(0.5, 0.3), \ v_{4\text{low}} = N(4, 0.3)\)
Figures 6.4 and 6.5 show the design spaces for both the high and low fidelity models. The high fidelity constraints are also shown in the low fidelity model for easier comparison. The high fidelity model has four local minima while the low fidelity model has only two; these points are represented in the figure by circles. The objective function contours of the two models have significant differences in value and in orientation. One would hope that in practice the low fidelity model would be a better match; however, this demonstration problem shows the robustness of the variable fidelity methodology.

Figure 6.4. High fidelity design space of the Barnes Problem.
In both implementation studies, different variable fidelity scalings and models are compared for use in reliability based design optimization. These results are also compared using a standard sequential quadratic programming (SQP) method to solve the RBDO problem. MATLAB’s \texttt{fmincon} was used as the SQP solver. The different variable fidelity scaling methods used were the multiplicative, additive, and adaptive hybrid. Furthermore, for each scaling the first order and quasi-second order methods using BFGS and SR1 were compared. All of the trials were started at the point $x = [40, 30]^T$ with an initial trust region size of $\Delta_0 = 10$. The results are summarized in Table 6.2.

All of the methods compared in Table 6.2 converged to the same solution. The objective of using these methods was to reduce the number of high fidelity function calls. Comparing the high fidelity function calls required for convergence
TABLE 6.2
VF-RBDO DOUBLE-LOOP PMA RESULTS FOR THE MODIFIED BARNES PROBLEM.

<table>
<thead>
<tr>
<th>Method</th>
<th>High</th>
<th>Low</th>
<th>Iter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiplicative Scaling</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1st Order</td>
<td>952</td>
<td>5580</td>
<td>12</td>
</tr>
<tr>
<td>2nd Order, BFGS</td>
<td>489</td>
<td>9571</td>
<td>7</td>
</tr>
<tr>
<td>2nd Order, SR1</td>
<td>442</td>
<td>5465</td>
<td>6</td>
</tr>
<tr>
<td>Additive Scaling</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1st Order</td>
<td>440</td>
<td>4439</td>
<td>7</td>
</tr>
<tr>
<td>2nd Order, BFGS</td>
<td>405</td>
<td>3001</td>
<td>6</td>
</tr>
<tr>
<td>2nd Order, SR1</td>
<td>377</td>
<td>3001</td>
<td>6</td>
</tr>
<tr>
<td>Adaptive Hybrid Scaling</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1st Order</td>
<td>481</td>
<td>3981</td>
<td>7</td>
</tr>
<tr>
<td>2nd Order, BFGS</td>
<td>409</td>
<td>6477</td>
<td>6</td>
</tr>
<tr>
<td>2nd Order, SR1</td>
<td>390</td>
<td>6590</td>
<td>6</td>
</tr>
<tr>
<td>RBDO SQP</td>
<td>1295</td>
<td>-</td>
<td>9</td>
</tr>
</tbody>
</table>
between the standard SQP RBDO method and all of the VF-RBDO methods shows a significant savings. The 1st order multiplicative method performed the worst in this case; though, it still used 26% fewer high fidelity calls. The highest savings was obtained using the 2nd order additive method using the SR1 update to approximate the Hessian information, and it achieved a 71% reduction in high fidelity calls. All of the second order methods out-performed the first order methods. The adaptive hybrid method seemed to select the better of the two methods but didn’t perform better than either one. One other factor that should be noticed when comparing these methods is that there was a much larger number of low fidelity function calls for the multiplicative method and the second order hybrid method. This could play a important factor, depending on the relative costs between the suite of fidelity models, so it is further studied in the next problem.

6.4.2 Energy Efficient Transport High-Lift Airfoil Design

The advanced energy efficient transport (AEET) design problem was introduced in Section 3.3.2. The problem formulation in this chapter is slightly modified from the previous chapters to include failure driven constraints with random variables. The problem solved here is: given a high-lift airfoil, find the optimal placement of its slat, vane, and flap to provide maximum lift for takeoff or landing configurations. The problem consists of nine design variables which control the horizontal, vertical, and rotational orientation of the slat vane and flap relative to their cruise configuration. The rotation is measured positive counter-clockwise about each control surface’s leading edge. Figure 3.8 shows the layout of the multi-element
supercritical airfoil in both cruise and high-lift configurations. The figure also shows the three degrees of freedom the control surfaces can move. Two failure driven constraints were placed on the lift to drag ratio and on the moment produced by the airfoil configuration. A deterministic constraint was placed on the distance, or gap, between the elements for gridding purposes. The formulation for the deterministic design problem is:

\[
\begin{align*}
\text{maximize} & \quad c_l \\
\text{subject to:} & \quad \frac{c_l}{c_d} \geq 85 \\
& \quad c_m \leq 0.65 \\
& \quad \text{gaps} \geq 1 \times 10^{-6}.
\end{align*}
\]

The flow conditions for the problem consisted of a Reynolds number of 9 million, a Mach number of 0.3, and an angle of attack 3 degrees. For the RBDO problem formulations the Mach number and angle of attack were considered as random variables. The lift, drag, and moment coefficients are all functions of the 9 design variables and of the Mach number and angle of attack. The deterministic values of Mach number and angle of attack are taken to be their mean values with both being normally distributed with variances of 5% of their respective mean values.

Like the previous AEET problems, the flow was solved using the inviscid Euler’s equations for the low fidelity model. The grid consisted of about 45,000 elements, as seen in Figure 3.9a, which extended to 30 times the chord length in each direction. For the high fidelity model, a full Navier-Stokes solution was used; the grid consisted of about 100,000 elements as seen in Figure 3.9b. The CFD runs took approximately 9 minutes and 2.5 hours for the low and high fidelity models respectively. To further reduce the computation expense of the low fidelity
model, a kriging-surrogate was created, reducing the expense to approximately a second. It should also be noted that a surrogate was created and used for the high fidelity model for the purpose of reducing the cost to a tractable level for demonstration purposes, which follows from the work done by Alexandrov\textsuperscript{[15]}. Both fidelity models were solved using FUN2D \textsuperscript{[17, 18]} developed at NASA Langley. This package uses fully unstructured mesh, which were generated using the advancing-front local-reconnection method described by Marcum\textsuperscript{[97, 98]}.

In comparing the various VF-RBDO methods the main goal is to reduce the number of high fidelity function calls and ultimately the total time needed to find an optimal design. For each case the total of high and low fidelity function calls were tallied along with an estimate of computational expense required. This computational expense, or relative cost, was calculated using a weight of 2.5 time units for a high fidelity function call and 0.15 units per low fidelity function call. The precise computational expense couldn’t be determined because of the use of the kriging models in place of the true CFD models. The results from the VF-RBDO double loop method using PMA and various variable fidelity methods are given in Table \textsuperscript{6.3}. The table also includes the results from a standard RBDO method using just the high fidelity model.

The initial design is shown in Figure \textsuperscript{6.6a} and has a lift coefficient of 2.11. The deterministic solution has a lift coefficient of 2.69 while the reliable design, shown in Figure \textsuperscript{6.6b}, has a lift coefficient of 2.51. All of the trials converged to the same solution. The results given in Table \textsuperscript{6.3} show that the variable fidelity methods have significantly reduced the relative computational cost of design.
### TABLE 6.3
VF-RBDO DOUBLE-LOOP PMA RESULTS FOR THE AEET PROBLEM.

<table>
<thead>
<tr>
<th>Method</th>
<th>High</th>
<th>Low</th>
<th>Iter</th>
<th>Rel Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>MultiplicativeScaling</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1st Order</td>
<td>228</td>
<td>3726</td>
<td>23</td>
<td>1129</td>
</tr>
<tr>
<td>2nd Order, BFGS</td>
<td>163</td>
<td>5300</td>
<td>14</td>
<td>1203</td>
</tr>
<tr>
<td>2nd Order, SR1</td>
<td>178</td>
<td>6506</td>
<td>17</td>
<td>1421</td>
</tr>
<tr>
<td>AdditiveScaling</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1st Order</td>
<td>256</td>
<td>3774</td>
<td>27</td>
<td>1206</td>
</tr>
<tr>
<td>2nd Order, BFGS</td>
<td>152</td>
<td>5298</td>
<td>13</td>
<td>1175</td>
</tr>
<tr>
<td>2nd Order, SR1</td>
<td>138</td>
<td>4592</td>
<td>12</td>
<td>1034</td>
</tr>
<tr>
<td>Adaptive HybridScaling</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1st Order</td>
<td>183</td>
<td>2969</td>
<td>16</td>
<td>903</td>
</tr>
<tr>
<td>2nd Order, BFGS</td>
<td>144</td>
<td>5538</td>
<td>15</td>
<td>1191</td>
</tr>
<tr>
<td>2nd Order, SR1</td>
<td>105</td>
<td>3200</td>
<td>10</td>
<td>743</td>
</tr>
<tr>
<td>RBDO SQP</td>
<td>1932</td>
<td>-</td>
<td>25</td>
<td>4830</td>
</tr>
</tbody>
</table>
compared to the standard SQP approach. All of the 2nd order methods reduced the number of high fidelity function calls but had mixed results when comparing the overall relative costs. Multiplicative 2nd order and the hybrid second order BFGS methods were more expensive because of a large increase of low fidelity function calls required. If the computational cost gap were increased, the 2nd order methods would have become much more efficient in all cases. The adaptive methods performed the best overall; they had a lower cost than either of the multiplicative or additive methods for the 1st order and 2nd order SR1 scalings. The highest savings came from the hybrid 2nd order SR1 scaling. This method reduced the relative cost of finding a reliable design by 85%.

Figure 6.6. Initial and reliable optimum design configurations of the AEET.
6.5 Summary and Conclusions

Compared with standard deterministic design optimization methods, reliability based design problems tend to greatly increase the computation time and expense required to reach a converged solution. In this chapter variable fidelity methods were used to reduce the cost of reliability based design optimization. This combined variable fidelity reliability based design optimization approach was compared to standard reliability optimization using two design problems: a nonlinear analytic problem and a high lift airfoil design problem. In both of the demonstration problems the number of high fidelity function calls required was significantly reduced using VF-RBDO.

Many different types of scaling options exist when using the variable fidelity approach. There are two main types of scaling functions, multiplicative and additive, which can be combined into an adaptive hybrid method. Each of these scaling types can be approximated using first order or second order methods. In computing the Hessian, the second order methods use variable metric methods, BFGS and SR1, to reduce the computational expense. All of these options were also compared in the two demonstration problems. The second order variable fidelity scaling methods required a smaller number of high fidelity function calls compared to the first order models, but a penalty of an increase in low fidelity function calls was observed in some cases. The computational increase in low fidelity calls is usually offset by the savings in the high fidelity evaluations, but it is mainly dependent on the cost ratio between the two fidelity models. The results
indicate that the additive method performs better than the multiplicative scaling for RBDO. The adaptive hybrid method was efficient at selecting the appropriate method without the designer having to select between the two scaling methods. In the airfoil design case the hybrid method actually had improved performance over either additive or multiplicative alone; however, these results are not true in general.

In conclusion, the VF-RBDO combined methodology has been shown to reduce the computational cost of performing reliability based design. The method requires a set of fidelity models and is most efficient when the relative cost between the models is large. Additionally, the variable fidelity and reliability based methods are easily combined.
CHAPTER 7

SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS FOR FUTURE RESEARCH

This final chapter gives a summary and conclusions related to the accomplishments made in this dissertation. The primary goal of the research was to reduce the cost of simulation-based design for computationally intensive simulations by improving the efficiency of variable fidelity optimization. A secondary goal was to broaden the applicability of such methods to solve more types of problems. Chapters 2 through 5 focus on the primary goal. Variable fidelity optimization schemes are made more efficient by improving the scaling functions used to scale the low fidelity models resulting in a better approximation of the high fidelity models. Better approximations lead to a reduction of the number of high fidelity analysis calls required for convergence and ultimately a reduction in the total computational time. In Chapter 6 the secondary goal is addressed when the variable fidelity framework is extended to solve reliability based design problems. This chapter also provides suggestions concerning potentially useful future research in these areas.
7.1 Summary and Conclusions

7.1.1 Variable Fidelity Framework

A detailed outline of the variable fidelity optimization framework was given in Chapter 2. The framework requires a set of models of different physical or numerical fidelity. Finding the optimal solution of the high fidelity model using the fewest high fidelity function evaluations is the primary objective of this framework. The low fidelity model provides a physics based global approximation of the high fidelity model at a fraction of the computational cost. A scaling function is constructed to match the low fidelity model to the high fidelity model to help insure convergence of the high fidelity model. This sequential approximate optimization processes is managed by a trust region move-limit strategy which also aids in convergence. The variable fidelity framework is ideal for solving moderate sized design problems with the number of design variables on the order of ten to a couple hundred. The size limitations arise due to memory requirements and also the computational power required to build and manage the scaling models.

The two most dominate scaling methods used before this research for use in approximate model management schemes were described in the beginning of Chapter 3. The scaling methods included a first order Taylor series expansion of a multiplicative and additive error function. A divide-by-zero issue was identified as a problem with the Taylor series multiplicative scaling method. The problem occurs near active constraints or if the objective’s minimum value is near zero. This implementation issue was avoided by adding an extra constant to both the low fidelity objective and constraints.
7.1.2 Adaptive Hybrid Scaling

Traditionally the variable fidelity optimization process required the designer to select an appropriate scaling method before performing the optimization. Without doing numerical experimentation on the high and low fidelity models, which can be quite computationally expensive, the most efficient scaling model cannot be determined \textit{a priori}. The adaptive hybrid scaling method combines both the additive and multiplicative functions using a weighted average. The weighted average is then adjusted using the nearest previous design point to select the appropriate scaling, relieving the need for the designer to select a scaling method. This saves computational time by negating the effect of choosing a poor scaling approach or determining the scaling function by evaluating the high fidelity function unnecessarily.

The adaptive hybrid scaling was demonstrated with considerable success on a variety of demonstration design problems. It was first used with first and second order Taylor series scaling functions on a two dimensional analytic problem in Section 3.3.1. The second order results from the two dimensional problem showed that the adaptive hybrid method converged as fast or faster as the best method of either the multiplicative or additive methods. The average savings from choosing an inappropriate method was found to be 17\% fewer high fidelity function calls for the second order methods. For this same problem when the adaptive hybrid was applied to the first order scaling, the number of high fidelity function calls was 35\% higher than the best single scaling method but 75\% lower than the worst; this mixed result still shows the advantage of using this method over selecting a scaling method with no prior knowledge. In Section 4.4.1 the same general results
were observed using the same test problem but with kriging based scaling functions, including the warm-started method. The adaptive hybrid scaling method also performed well when applied to reliability based design problems. In finding a reliable design for a high-lift airfoil, the adaptive hybrid method saved 21%, on average, over the best of the single scaling methods alone. Based on these diverse trials, it can be concluded that the adaptive hybrid method is an efficient scaling method that relieves the designer from having to choose between using either multiplicative or additive methods alone and it can, therefore, reduce the overall expense of the variable fidelity design process.

7.1.3 Second Order Taylor Series Scaling

Traditionally variable fidelity or model management schemes have, in the past, used only first order scaling functions. Second order scalings can be done; though, it would be costly to construct such functions because of the required second order information. The expense of second order information is compounded when the function itself is computationally intensive. Chapter 3 introduces a scaling method that approximates the second order information using variable metric methods such as BFGS and SR1. These variable metric methods use consecutive first order information to build up an approximation of the second order information with no additional function calls. Since the second order information is approximated, full second order convergence rates are not expected; however, super linear convergence can be obtained with no additional high fidelity function calls per iteration, reducing the overall cost of the optimization.
The approximate second order scaling was found to require much fewer high fidelity function calls than the first order method on a variety of test cases. In the two dimensional analytic problem in Section 3.3.1, the approximate second order scaling methods used, on average, 24% fewer high fidelity function calls than did the first order methods. The results of the AEET or high-lift airfoil design problem, solved in Section 3.3.2, had a 49% reduction of high fidelity function calls using second order methods over the first order methods. In this problem the SR1 method was more efficient than using the BFGS approach to approximate the second order information. When applied to reliability based design problems, the second order methods again showed significant savings. For the modified Barnes problem (Section 6.4.1), the first order methods required 26%, on average, more high fidelity function calls that the approximate second order methods. In the reliability version of the AEET problem (Section 6.4.2), the total cost of optimization was studied. The BFGS second order method increased the computational expense to find the optimal design by 12%, on average, compared to using the first order methods, while the SR1 method reduced the cost by only 2%. In both the BFGS and SR1 methods the number of high fidelity function calls was reduced; however, the number of low fidelity calls was greatly increased, much more than in the other test problems. The reason for the loss of savings for this problem was due to the high ratio of computational cost of the low fidelity model to the high fidelity model. The second order method would have reduced the cost of optimization if the low fidelity model would have been cheaper to evaluate. It is important to note that for this problem the variable fidelity optimization methods did reduce the cost of optimization by 77%, compared to using a standard single fidelity approach.
Both the first order and second order scaling techniques for variable fidelity optimization use only local information to construct the scaling function. The low fidelity model is typically a physics based global approximation of the high fidelity model. It was therefore, hypothesized that a global scaling method might improve the convergence performance of the variable fidelity framework by allowing the optimizer to take larger steps toward to optimum. The kriging interpolation scheme was selected to incorporate all previous design points into the construction of the scaling function. The kriging scaling method also allows for incorporating all past evaluations of the various fidelity models to enhance the scaling model and further improve the optimization process. It was found that the kriging model alone was insufficient to provide a computational cost savings because of the lack of data points incorporated into the initial scaling functions. This problem was alleviated by starting the variable fidelity process using a Taylor series scaling method; then once a large enough design history, $O(m)$, has been accumulated, the kriging scaling functions are used. This combined approach was called the warm-started kriging method. The warm-started and standard kriging based scaling methods were introduced in Chapter 4.

The standard kriging based scaling method did not significantly lower the computational cost of the optimization over the second order methods. However, the warm-started method was demonstrated to be able to reduce the computational time required for optimization over the second order methods. When applied to
the two dimensional analytic problem (Section 4.4.1), the warm-started kriging method used 14% fewer high fidelity function calls than did approximate second order methods alone. When the warm-started method was applied to the AEET problem (Section 4.4.2), it required an average of 41% fewer high fidelity function calls than the Taylor series based methods. The number of low fidelity function calls significantly increased when kriging based scaling methods were used. This increase limits its application to certain problems. The warm-started method is most efficient when the cost of evaluating the low fidelity model is much less than the high fidelity model. When the low fidelity model evaluation cost is more than approximately 10% of the high fidelity model cost, then second order methods should be used exclusively. Otherwise, the warm-started kriging based scaling method is more efficient, as it reduces the number of high fidelity function calls required for convergence, and should be used instead of the approximate second order scaling methods.

7.1.5 Metamodel Update Management Strategies for Kriging Scaling Models

In the process of constructing a kriging interpolation model a set of correlation parameters must be selected. They are typically found by solving an optimization problem. When the number of sample points is large or the number of design variables is large, this optimization problem may itself require a lot of computational time to solve. This can become a problem when the expense of constructing the kriging models to be used as scaling functions significantly reduces or eliminates the savings of using the variable fidelity framework. In Chapter 5 two metamodel update management strategies (MUMS) were introduced to dictate whether the
kriging model parameters need to be updated every iteration. The first scheme, L-MUMS, uses the ratio between likelihoods of consecutive kriging models to determine if an update is needed; the second method, TR-MUMS, uses the trust region ratio. The goal of these schemes is to reduce the number of kriging model parameter updates during the variable fidelity optimization process, thereby reducing the overall cost.

The likelihood function is a standard objective used in determining the kriging model parameters. This function may be multi-modal or have very flat regions, which can be problematic for gradient based optimizers. Three different optimizers for updating the kriging models were also studied in conjunction with the update schemes. The optimizers included: adaptive simulated annealing (ASA), Quasi-Newton method, and pattern search.

Initial studies of the two update schemes found that the likelihood ratio was surprisingly poor at predicting when the kriging model parameters should be updated. The TR-MUMS, on the other hand, was shown to be quite effective when applied to an internal combustion engine sizing problem and a control-augmented structure problem. The results of the internal combustion engine sizing problem indicated that the kriging parameters had little effect on the overall convergence of the variable fidelity optimization. For this problem, using a constant set of kriging parameters performed just as well as updating them every iteration. The TR-MUMS was able to converge to the same solution while only requiring 1 to 2 updates of the kriging model, depending on the kriging model optimizer. This was a three-fold reduction in kriging model updates as compared to updating
every iteration. In the control-augmented structural design problem it was found that the kriging models did need to be updated. Using a constant set of kriging parameters, the variable fidelity optimization did not converge to an optimum because the kriging approximation became unusable after a few iterations. The TR-MUMS performed well by converging to the optimal solution and decreasing the number of kriging updates by half, as compared to updating every iteration. In both problems the ASA optimizer was not observed to significantly reduce the number of high fidelity function calls required for the variable fidelity framework to converge. The ASA method required two to three orders of magnitude more likelihood evaluations and is, therefore, not considered to be an effective method to find the kriging parameters in such a framework. The pattern search used the fewest number of likelihood evaluations for each of the trials and did not affect the overall convergence properties of the variable fidelity algorithm. The results from this study indicate that using the TR-MUMS in conjunction with kriging scaling functions can save computational resources by decreasing the number of kriging model update optimizations required for convergence. Furthermore, the kriging models should be updated using the pattern search optimizer rather than the Quasi-Newton method or ASA.

7.1.6 Variable Fidelity Reliability Based Design Optimization

Reliability based design problems are computationally intense because they require the evaluation of the deterministic objective and constraint functions as well as a reliability analysis of the design. Chapter 6 presented a method to reduce the computational cost of solving reliability based design optimization problems
by making use of the variable fidelity framework when a suite of fidelity models is available. A standard method of solving a reliability design problem is to reformulate the deterministic constraints as reliability constraints, requiring that the most probable point of failure associated with a given design meet a desired reliability index. This new formulation is typically solved using a double loop optimization method. This method, as its name implies, has two optimization loops: an outer loop converges the reliability formulation and an inner loop finds the most probable point of failure for each constraint. The solution is expensive because of the large number of function calls required by nested optimizers. To solve this problem using variable fidelity methods, scaling functions are built for matching both the reliability constraints as well as the objective function for the different fidelity models. This is a new application of variable fidelity methods which aids in reducing the computational expense of reliability based design problems.

The combined variable fidelity reliability based design optimization framework was successfully demonstrated on two problems: the modified Barnes problem (Section 6.4.1) and a reliability version of the advanced energy efficient transport problem (Section 6.4.2). The results of the Barnes problem show that, on average, the variable fidelity methods require 62% fewer high fidelity function calls to converge compared to a standard single fidelity SQP optimizer. The second order scaling methods used fewer high fidelity function calls but required more low fidelity calls than the first order scaling methods. The results of the transport problem showed a similar savings trend. In terms of the total computational costs, the variable fidelity methods converged, on average, a factor of 4.3 times
faster than the single fidelity SQP optimizer. The results of these demonstrations show the variable fidelity reliability based optimization method to be a viable solution in helping to reduce the computational cost of solving RBDO problems.

7.2 Recommendations for Future Research

The variable fidelity scaling methods, kriging update management schemes, and incorporation of reliability based design problems all offer significant improvement over existing optimization methods for the design of products involving expensive computer simulations. More work in these areas could certainly provide further improvements. This section discusses recommendations for future research that might prove beneficial to the methods introduced in this dissertation.

7.2.1 Adaptive Hybrid Scaling Improvements

The adaptive hybrid scaling method uses a weighted sum of the additive and multiplicative scaling functions. The weighted sum formation requires an additional matching condition to calculate the weighting parameter that controls how the two scaling functions are combined. The research done in this dissertation requires that the hybrid scaling function exactly scale a previous design point as well as the local derivatives. The previous design point is taken to be the closest point, or smallest Euclidean norm, from the current design point. Different methods of calculating the weighting parameter may increase the effectiveness of the method. A few conditions that would provide interesting comparisons with the method used in this research include: matching the previous iterate instead
of the closest, using a least squares fit incorporating all of the previous design points, and matching a projected design in the usable feasible region. The last suggested condition would require one extra high and low fidelity evaluation per iteration; however, this point would provide a better approximation for the next iteration because it uses information that is likely to be more in the direction of the optima as opposed to designs that are further away.

7.2.2 Kriging Based Scaling Improvements

The results of Chapter 4 showed that using a kriging based scaling method can reduce the number of high fidelity function calls required for convergence as compared to the first and second order Taylor series scaling methods. However, the results also indicated that the method required a larger number of low fidelity function calls because of the increased complexity of the approximate design space. When the computational time required by the low fidelity model is a significant fraction of the high fidelity model this can cause the method to become less efficient than the Taylor series methods. Future research on this method should consider strategies to reduce the number low fidelity function calls required for convergence without unduly increasing the high fidelity function count. One approach to address this problem would be to not use kriging strictly as an interpolation function. The correlation matrix can be formulated to match the current design and the gradient but allow past design points to not be matched exactly. Kriging models created using this approach would be smoother because they would be less constrained, resulting in a less complex approximate design space. Another useful study would be to compare the results obtained using the
kri
ging model to other global scaling methods such as neural networks or radial basis functions.

7.2.3 Variable Fidelity Reliability Based Design Optimization Improvements

In Chapter 6, reliability based design problems are solved using the variable fidelity optimization framework. The double loop method was used as the reliability formulation. Recent advances in reliability design methods have been proposed to reformulate the nested optimizations into a single level optimization problem. This is done by augmenting the number of design variables and adding the KKT conditions of the inner optimization as equality constraints in the outer optimization problem. This uni-level formulation, however, is a much more complicated problem because of: the increase in design variable size, inclusion of equality constraints which may hinder convergence of sequential approximate optimization methods, and the requirement of second derivatives of the constraints. Incorporating the uni-level formulation into the variable fidelity framework is difficult for these reasons; but it might be worthwhile to further reduce the number of high and low fidelity function calls required for convergence of reliability based design problems.

7.3 Concluding Remarks

The focus of this dissertation has been on developing improvements of variable fidelity optimization methods to reduce the computational cost of simulation based design. Variable fidelity methods reduce the cost of complex simulations by
using a suite of models of different physical fidelity, and then scaling the inexpensive models to approximate the expensive high fidelity models. Such sequential approximate optimization methods can currently be expected to handle design problems of moderate size ranging from design variables on the order of tens to possibly a few hundred, as long as sensitivity information can be obtained relatively inexpensively. Automated design optimization methods, such as those presented in this research, will probably never replace a critically thinking human designer. This is especially true for projects that directly affect human safety. However, such methods can be used as a tool to help solve problems efficiently and to enable designers to tackle evermore complex problems.
1. GENESIS Structural Analysis and Optimization Reference Manual. Vande-
plaats Research & Development, Inc., Colorado Springs, CO 80906, version


4. H. Agarwal and J. E. Renaud, Decoupled methodology for probabilistic de-
sign optimization. In ASCE Joint Specialty Conference on Probabilistic Me-
chanics and Structural Reliability, Albuquerque, NM (July 2004).

5. H. Agarwal and J. E. Renaud, Reliability based design optimization using
response surfaces in application to multidisciplinary systems. Engineering

method for reliability based design optimization. In Proceedings of the 45th
AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Mate-
rials Conference, number AIAA 2004-2029, Palm Springs, CA (19 - 22 April
2004).

7. H. Agarwal, J. E. Renaud and J. D. Mack, A decomposition approach for
reliability-based multidisciplinary design optimization. In Proceedings of the
44th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and
Materials Conference & Exhibit, number AIAA 2003-1778, Norfolk, Virginia
(April 7-10 2003).

8. H. Agarwal, J. E. Renaud and V. M. Pérez, Homotopy methods for constraint
relaxation in unilevel reliability based design optimization. In Proceedings
of the 10th AIAA/ISSMO Multidisciplinary Analysis & Optimization Con-
ference, number AIAA 2004-4402, Albany, NY (30 August - 1 September
2004).


90. Y. B. Lim, J. Sacks, S. W. J. and W. J. Welch, Design and analysis of computer experiments when the output is highly correlated over the input space. Technical Report 62, National Institute of Statistical Sciences, 19 T.W. Alexander Dr., PO Box 14006, Research Triangle Park, NC 27709-4006 (August 1997).


