Multifidelity Methods for Multidisciplinary System Design

by

Andrew I. March
B.S., Cornell University (2005)
S.M., Massachusetts Institute of Technology (2008)

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June 2012

Author .....
Department of Aeronautics and Astronautics
April 30, 2012

Certified by .....
Karen Willcox
Associate Professor of Aeronautics and Astronautics
Thesis Supervisor

Certified by .....
Qiqi Wang
Assistant Professor of Aeronautics and Astronautics
Thesis Committee Member

Certified by .....
Jaime Peraire
Professor of Aeronautics and Astronautics
Thesis Committee Member

Accepted by .....
Eytan Modiano
Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee

Multifidelity Methods for Multidisciplinary System Design

by

Andrew I. March

Submitted to the Department of Aeronautics and Astronautics
on April 30, 2012, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Optimization of multidisciplinary systems is critical as slight performance improvements can provide significant benefits over the system's life. However, optimization of multidisciplinary systems is often plagued by computationally expensive simulations and the need to iteratively solve a complex coupling-relationship between subsystems. These challenges are typically severe enough as to prohibit formal system optimization. A solution is to use multifidelity optimization, where other lower-fidelity simulations may be used to approximate the behavior of the higher-fidelity simulation. Low-fidelity simulations are common in practice, for instance, simplifying the numerical simulations with additional physical assumptions or coarser discretizations, or creating direct metamodels such as response surfaces or reduced order models. This thesis offers solutions to two challenges in multidisciplinary system design optimization: developing optimization methods that use the high-fidelity analysis as little as possible but ensure convergence to a high-fidelity optimal design, and developing methods that exploit multifidelity information in order to parallelize the optimization of the system and reduce the time needed to find an optimal design.

To find high-fidelity optimal designs, Bayesian model calibration is used to improve low-fidelity models and systematically reduce the use of high-fidelity simulation. The calibrated low-fidelity models are optimized and using appropriate calibration schemes convergence to a high-fidelity optimal design is established. These calibration schemes can exploit high-fidelity gradient information if available, but when not, convergence is still demonstrated for a gradient-free calibration scheme. The gradient-free calibration is novel in that it enables rigorous optimization of high-fidelity simulations that are black-boxes, may fail to provide a solution, contain some noise in the output, or are experimental. In addition, the Bayesian approach enables us to combine multiple low-fidelity simulations to best estimate the high-fidelity function without nesting. Example results show that for both aerodynamic and structural design problems this approach leads to about an 80% reduction in the number of high-fidelity evaluations compared with single-fidelity optimization methods.

To enable parallelized multidisciplinary system optimization, two approaches are developed. The first approach treats the system design problem as a bilevel programming problem and enables each subsystem to be designed concurrently. The second approach optimizes

surrogate models of each discipline that are all constructed in parallel. Both multidisciplinary approaches use multifidelity optimization and the gradient-free Bayesian model calibration technique, but will exploit gradients when they are available. The approaches are demonstrated on an aircraft wing design problem, and enable optimization of the system in reasonable time despite lack of sensitivity information and 19% of evaluations failing. For cases when comparable algorithms are available, these approaches reduce the time needed to find an optimal design by approximately 50%.

Thesis Supervisor: Karen Willcox
Title: Associate Professor of Aeronautics and Astronautics

Thesis Committee Member: Qiqi Wang
Title: Assistant Professor of Aeronautics and Astronautics

Thesis Committee Member: Jaime Peraire
Title: Professor of Aeronautics and Astronautics

Acknowledgments

I wish to thank my advisor Prof. Karen Willcox for her continued support and guidance throughout my tenure at MIT. Karen has been an indispensable resource and constant ally. She provided me with a balance of freedom to pursue research avenues and supervision to ensure quality results. Her meticulous reviews of my work ensured that my mathematics was correct and that my points were communicated clearly. I appreciate all of the time and effort she has dedicated to me over the past six years.

I also wish to thank my committee members Prof. Qiqi Wang and Prof. Jaime Peraire. Qiqi's door was always open, and his help with adjoints and in dissecting analyses was invaluable. I sincerely appreciate the copious time he spent going through my work. Jaime was a consistently jovial presence and always provided a word of encouragement, including suggesting further research avenues to pursue.

My readers, Prof. Joaquim Martins and Dr. Andrew Conn, have provided valuable advice to improve the quality of this thesis. In addition, previous work by Prof. Martins guided me to find potential research contributions. The aerostructural software developed by Prof. Martins, Graeme Kennedy, Gaetan Kenway, and Sandy Mader enabled the work in Chapter 6. I thank Graeme for the time and effort he put in to helping me use and understand that software. Dr. Conn's derivative-free optimization paper was the inspiration for much of the work in this thesis. His past research provides much of the foundation for this thesis, and his books are the two *de facto* references for my work.

I am compelled to thank other MIT faculty that contributed to my education and research at MIT. Prof. Darmofal spent a semester teaching me how to teach. Prof. Drela, my minor advisor, has been an amazing resource for all aspects of aerodynamics and aircraft design. Prof. Marzouk guided my research through many discussions on uncertainty quantification and how it could be applied to my work. Prof. de Weck's lectures on system design challenges and heuristic optimization methods were inspiration to do better.

I have many MIT friends that I would like to thank for their support, guidance and camaraderie. I thank David Lazzara for being a great person to share a cubicle with for

three years and for many fruitful research discussions. Leo Ng has also been a great office-mate, dog sitter and resource for the past year. I thank my Spanish friends, David Moro and Joel Saá-Seoane, for help with mathematics and for always having an excuse to grab a beer. Hemant Chaurasia and Xun Huan have been a source of motivation to exercise and start a business. Marcelo Buffoni, Jeremy Agte, and Rhea Liem are three good friends that have moved on from MIT, but improved both my research and quality of life while here. There are many other members of the ACDL and Flying Club that, although not named here, were good friends and supporters.

There are many members of our support staff that made my life significantly easier. I wish to thank, Marie Stuppard, Beth Marois, Sue Whitehead, Barbara Lechner, and Meghan Pepin. In addition, the custodians Sam, Norma, and Phylis always provided a welcome light-hearted moment in the afternoon. I especially wish to thank Jean Sofronas. Jean taught me the way around MIT, saved me countless hours of work, and was a great friend and confidant. It is scary to imagine having a job without her around.

I also gratefully acknowledge financial support from NASA Langley Research Center contract NNL07AA33C, technical monitor Natalia Alexandrov, AFOSR Computational Mathematics Program, Grant FA9550-06-0271, Program Manager Dr. Fariba Fahroo, and a National Science Foundation graduate research fellowship. I thank the Space Systems Lab for the use of their Nastran licenses and Mathew Smith for support with that. I also thank Michael Aftosmis and Marian Nemec for providing us with Cart3D.

To my wife Michelle, thank you for the unwavering support. I could not have done this PhD without your support. You have been a source strength, warmth, ideas, and words throughout this process.

Finally, I thank my parents, grandparents, in-laws, and extended family for their support in my pursuit for a PhD. I wish to dedicate this thesis to my grandparents, Ed Levy, Bertha March, Irene Levy, and my grandfathers that I never met, Samuel Lazarus and Irving March. My Grandpa Ed is the likely source of my love for science and engineering, my Grandmother Bertha (Grammie) is the likely source of my determination, and paid for all of my textbooks for ten years of college, and my Grandma Irene a likely source of my analytical skills.

Contents

List of Figures

List of Tables

List of Symbols

A	Active and violated constraint Jacobian
a	Sufficient decrease parameter
B	A closed and bounded set in $\mathbb{R}^n$
$\mathcal{B}$	Trust region
b	Gauss-Newton merit function
$\mathbf{C}$	Output matrix
C	Differentiability class
$C_p, \delta C_p$	Pressure coefficient, change in pressure coefficient
$c(\cdot)$	Inequality constraint requiring multifidelity methods
D	Drag
$\mathbf{d}$	High-fidelity function sample point
d	Artificial lower bound for constraint value
E	A convex constraint set
$e(\cdot)$	Error model for objective
$\bar{e}(\cdot)$	Error model for constraint
$\mathcal{F}(\cdot)$	A system performance objective function
F	Area dominated by the filter
$\mathbf{f}$	Applied force vector
$f(\cdot)$	Objective function
$\mathcal{G}(\cdot)$	An inequality constraint for a system
G	Area bound by the Pareto front of objective and constraint values
$g(\cdot)$	Inequality constraint
$\mathcal{H}(\cdot)$	An equality constraint for a system
H	Hessian of the surrogate model
$h(\cdot)$	Equality constraint
I	Identity matrix
$\mathcal{J}(\cdot)$	Multivariate objective function
$\mathbf{K}$	Stiffness matrix
k_1, k_2	Constant ratio to the l_2 norm
$K\mathrm{S}$	Kreisselmeier–Steinhauser functions
$\mathcal{L}$	Expanded level-set in $\mathbb{R}^n$
L	Level-set in $\mathbb{R}^n$, or aerodynamic lift
$\mathfrak{L}$	Lagrangian
l	Number of state variables

$L\!/\!D$	Lift-to-drag ratio
$\mathcal{M}$	Space of all fully linear models
$\mathbf{M}$	Mesh
M_∞	Mach number
$m(\cdot)$	Surrogate model of the high-fidelity function
$\bar{m}(\cdot)$	Surrogate model of the high-fidelity constraint
$\mathcal{N}(\cdot,\cdot)$	Normal distribution
m	Number of constraints or number of disciplines
n	Number of design variables
P	A nullspace projection operator
$\mathbf{p}$	Arbitrary step vector
p	Number of calibration/interpolation points
Q	A range-space projection operator
$\mathbf{q}$	A measure of primal feasibility
q	Vector component index, $q = 1, \ldots, n$
$\mathbf{R}$	Correlation matrix
$\mathcal{R}(\cdot)$	Vector of residual equations
$\mathbf{r}(\cdot)$	System response vector/collection of discipline responses
r	Radial distance between two points
$\mathbf{s}$	Trust region step
$\mathbf{t}$	Discipline targets
t	Constant between 0 and 1
U	Matrix of left singular values
$\mathbf{u}$	Vector of state variables
V	Matrix of right singular values
$\mathbf{v}$	Kriging values to interpolate
v	Constraint violation conservatism factor, or dimension of system response
w	Weight
$\mathbf{x}$	Design vector
$\tilde{\mathbf{x}}$	Subsystem level copy of the design vector
$\mathcal{Y}$	Set of calibration vectors used in an error model
$\mathbf{y}$	Vector from current iterate to a high-fidelity sample point
$\mathbf{z}$	A composite vector of ones and zeros, complete system design vector
α	Convergence tolerance multiplier
β	Convergence tolerance multiplier, or filter acceptance tolerance
Γ	Cholesky decomposed matrix
γ	Trust region expansion/contraction ratio, or penalty increase ratio
Δ	Trust region radius
$\delta(\cdot)$	Lower bound for trust region size
δ_h	Linearized step size for feasibility
δ_x	Finite difference step size
ϵ	Termination tolerance
ϵ_2	Termination tolerance for trust region size

ζ	Trust region contraction ratio used in convergence check
η	Trust region expansion/contraction criteria, or penalty increase criteria
Θ	Likelihood
θ	Positive constant, used used in surrogate model construction
$\vartheta, \delta\vartheta$	Flow angle, change in flow angle
κ	Bound related to function smoothness or range
Λ	Eigenvalue
λ	Lagrange multiplier
$\hat{\lambda}$	Lagrange multiplier for surrogate model
μ	Penalty parameter
$\tilde{\mu}$	Penalty parameter for system-subsystem design matching
ν	Coefficients of the polynomial interpolant, or Lagrange multiplier
ξ	Radial basis function correlation length
Π	Polynomial basis for $\mathbb{R}^n$
π	A component of the polynomial basis
ρ	Ratio of actual to predicted improvement
ϱ	Norm of inner product of constraint vector and constraint Jacobian
Σ	Matrix of singular values
σ^2	Mean square error or variance
$\hat{\sigma}^2$	Max. likelihood estimate of σ^2
τ	A solution tolerance
Υ	Penalty function
$\hat{\Upsilon}$	Surrogate penalty function
υ	Constraint violation conservatism factor
Φ	Correlation vector
$\phi(\cdot)$	Radial basis function/correlation function
Ψ	Adjoint variable
ψ	Kriging constant
Ω	A convex compact domain
ω	Coefficient of the radial basis function interpolants

Superscript

*	Optimal
$^+$	Active or violated inequality constraint
†	Moore-Penrose pseudoinverse
$^\perp$	Normal component
^{N}	Nullspace component
$^\parallel$	Tangential component
$^{(p)}$	Vector component p
$^{(1:p)}$	Vector components $1, \ldots, p$
^{i}	Discipline i
^{q}	q-component vector
^{T}	Transpose

Subscript

$\nabla\mathcal{H}$	Error bound for surrogate constraint Jacobian
0	Initial iterate
bhm	Bound for Hessian of the surrogate model
blg	Upper bound on the Lipschitz constant
C	Cokriging
C_p	Pressure coefficient response or target
c	Relating to constraint
cg	Relating to constraint gradient
cgm	Maximum Lipschitz constant for a constraint gradient
cm	Maximum Lipschitz constant for a constraint
Def	Deflection response or target
est	Maximum likelihood estimate
f	Relating to objective function
fg	Lipschitz constant for objective function gradient
FCD	Fraction of Cauchy Decrease
g	Relating to objective function gradient
$\mathcal{H}$	Error bound for surrogate constraint vector
high	Relating to the high-fidelity function
i	Vector with index i
j	Vector with index j
k	Index of trust-region iteration
l	Index of a point on the filter
low	Relating to a lower-fidelity function
$\bar{m}$	Related to constraint surrogate model
max	User-set maximum value of that parameter
med	Relating to an intermediate-fidelity function
min	User-set minimum value of that parameter
t	Constant related to design target sensitivity
$\tilde{x}$	Constant related to subsystem design sensitivity
τ	Sufficiently linearly independent condition

Chapter 1

Introduction

The ability to optimize complex systems using high-fidelity simulations can yield physical systems with unprecedented performance. Complex systems span a broad range from aircraft to nuclear reactors to transportation networks to nanoscale devices, and for all of these systems sophisticated simulations play an important role in design and performance assessment. However, the cost of these simulations is often too high for direct optimization to be tractable. These systems are composed of many interacting and coupled subsystems, each of which is modeled with an expensive simulation. Furthermore, the sensitivity of a system's performance to its design variables is often not available, and the system performance estimates may be corrupted by poor convergence behavior of the simulations or inherent uncertainty in the processes. Therefore, many expensive system or subsystem simulations will be required to find a system design with good performance, let alone optimal performance.

Section 1.1 motivates multifidelity optimization and system decomposition as potential ways of mitigating these system design challenges. Section 1.2 presents an introduction to multifidelity optimization, and Section 1.3 provides an introduction to multidisciplinary optimization, and synergisms between multifidelity and multidisciplinary optimization. Section 1.4 formally outlines the research objectives of this thesis intended to address the issues noted in multifidelity and multidisciplinary optimization. The chapter concludes by outlining the remainder of the thesis in Section 1.5.

1.1 Motivation

Typical early-stage or conceptual design of a complex system is done using basic physical tools or empirical methods, but these tools may not capture important physics, especially when new technologies are involved. We will use the term *fidelity* to describe how well a model estimates reality or equivalently the expected quality of the model. Our goal is to bring high-fidelity analyses earlier into the design process. Enabling conceptual design with high-fidelity simulation will provide the opportunity for systems to take advantage of complex phenomena not captured by lower-fidelity methods and will aid development of novel systems with excellent performance. Moreover, the use of high-fidelity simulations helps to reduce the need for costly redesigns due to a system not meeting performance targets or behaving unexpectedly as the system characteristics are more accurately modeled.

The problem with using high-fidelity simulation in the conceptual design phase is the time and expense. High-fidelity simulations can either be physical experiments or numerical simulations, both of which are expensive and time consuming. In the early stages of design it is important to rapidly evaluate a design so numerous configurations can be analyzed. This rapid turn-around is usually only possible with low-fidelity analyses. A multifidelity framework attempts to mitigate this challenge by using the lower-fidelity analyses to accelerate convergence towards the optimal high-fidelity design. In this thesis the requirements placed upon a low-fidelity model are minimal, simply a low-fidelity model must estimate the same metric of interest as the high-fidelity model and be a smooth function. There are no requirements that the low-fidelity model be accurate or qualitatively similar to the high-fidelity model. This means a low-fidelity model could be the same type of physical simulation as the high-fidelity one, but with additional physical approximations, it could be the same simulation as the high-fidelity but solved on a coarser discretization [5], it could be an empirical code, a response surface [36, 105, 114], or a reduced order model [6]. However, one would expect a multifidelity method to perform better when the lower-fidelity results better resemble the high-fidelity output.

The challenge to implementing multifidelity design strategies is in developing formalized

frameworks that can use the lower resource, low-fidelity methods to design complex systems that are optimal with respect to the highest fidelity simulations available. This challenge has two dominating aspects: first, developing methods to determine when recourse is needed to the high-fidelity methods in order to calibrate the low-fidelity methods, and second, how to best employ multifidelity strategies to system design when the multiple fidelity analysis methods are available for system components or disciplines as opposed to the entire system, i.e., when multiple high-fidelity simulations are coupled. Some other challenging aspects of these two overarching issues are optimizing systems in which the sensitivity of the system performance to the design variables is unknown, and how to best parallelize the system design process. The following sections summarize the state-of-the-art in multifidelity optimization, and present some state-of-the-art methods for designing multidisciplinary systems.

1.2 Multifidelity Optimization

Numerous heuristic techniques have been used to optimize a high-fidelity function using lower-fidelity information. We consider heuristic multifidelity optimization (MFO) approaches to be approaches that generally converge in practice to an optimum of the high-fidelity function, but in which there is no formal mathematical proof or guarantee. These methods vary from problem specific necessities to rigorous methods that compute a probability of finding an improved high-fidelity function value. Examples of problem specific multifidelity approaches include adding global response surface corrections to low-fidelity models [9], using the low-fidelity function gradient as the optimization direction, but performing the line search with the high-fidelity function values [10], creating a response surface using both high- and low-fidelity analysis results [53], and running higher-fidelity models when two or more lower-fidelity models disagree [22]. In contrast, we consider a non-heuristic method to be one in which given a set of requirements for the initial design(s) and behavior of the high- and low-fidelity functions, there is a mathematical guarantee that with enough time the multifidelity method will find a high-fidelity optimum. Non-heuristic multifidelity methods may converge slower than single-fidelity methods, but nonetheless they are guaranteed to work

eventually. Our discussion of MFO methods focuses on both heuristic and non-heuristic methods that are broadly applicable and likely to find a high-fidelity optimum for general problems.

In MFO there has largely been two types of approaches, global and local. Global methods search the entire feasible domain for the best design, whereas local methods attempt to find the nearest design that has better performance than all other designs in that neighborhood. Global methods have the benefit that they typically do not require estimates of the high-fidelity function's gradient. This is important because frequently a high-fidelity function's gradient is unavailable and cannot be estimated accurately. For example when the high-fidelity function (i) is a simulation with a finite convergence tolerance, (ii) is a black-box, (iii) contains random noise in the output, (iv) may occasionally fail to provide a solution, or (v) is a physical experiment, then a high-fidelity gradient will not be available and estimations of it may contain significant error. A detriment to using global optimization methods is that they typically require considerably more high-fidelity evaluations than a local method. So, there is clearly a need for both types. This section first presents global MFO methods and then local MFO methods. The section highlights both when a method is guaranteed to find a high-fidelity optimum and when the method requires high-fidelity gradient information.

1.2.1 Global MFO

Common gradient-free global optimization frameworks are based on interpolating the high-fidelity objective function, for example by estimating the location of high-fidelity optima using linear segments [51] or Gaussian process regression (equivalently Kriging) [52]. The latter technique, known as Efficient Global Optimization (EGO), offers many advantages such as an uncertainty estimate. The mean of the Gaussian process interpolates the value of the high-fidelity function, while the mean square error of the Gaussian process models the uncertainty in the high-fidelity function value. This error estimate is zero at all locations where the value of the high-fidelity function is known and increases with distance away from sample points. Optimization is then performed on the Gaussian process model, and the high-fidelity function is sampled at locations likely to reduce to the value of the function

over the current observed minimum. This technique works well in practice, can be made globally convergent [50], and does not require a high-fidelity derivative estimate. However, the method may be globally biased and attempt to explore the entire design space as opposed to simply reducing the objective function. In addition, the method has been shown to be sensitive to both the initial high-fidelity samples [50] and to the exact metric of selecting points likely to improve the high-fidelity function value [101]. Other methods related to EGO include multiobjective methods treating both improvement and risk as objectives when looking for performance improving points [91, 92], and a provably convergent formulation by Regis *et al.* that considers both the value of the Gaussian process and the distance to previous interpolation points [94].

Kennedy and O'Hagan enabled EGO to be used with multiple-fidelity models by proposing Gaussian process models as a way of calibrating lower-fidelity models to best predict the output of high-fidelity models [55]. They also performed a complete Bayesian uncertainty analysis of calibrating computer models to true physical processes, not just higher-fidelity models, and determined where the uncertainties arise [56]. This work is significant in a multifidelity context because calibration-based approaches likely predict high-fidelity behavior better than interpolation-based approaches. The rationale is that an interpolation-based approach can be seen as calibrating a low-fidelity model that is zero everywhere, however, by calibrating a low-fidelity model that captures some high-fidelity behavior only errant portions of the design space need to be corrected. Further calibration-based work by Leary *et al.* suggests that the error between the high- and low-fidelity models can be either additive as posed by Kennedy and O'Hagan or multiplicative [62]. Moreover, Huang *et al.* proposed an extension of these techniques that allows for heuristic combination of multiple lower-fidelity models [48].

When the high-fidelity gradient is available, in lieu of Kriging/Gaussian process regression, Cokriging surrogate models are often used. Cokriging surrogates are similar to Kriging models except they interpolate a function with both the correct function value and with the correct gradient. For the same number of interpolation points, a Cokriging model will typically capture the behavior of a function better than an ordinary Kriging model due to

the additional information the gradient provides [23, 24]. The typical implementation of Cokriging methods in optimization is to create a global surrogate model of the expensive function. Iteratively updating these global Cokriging models during an optimization often results in finding a high-fidelity optimum faster than with other calibration methods [63]; this was even found to be true when the number of Cokriging calibration points was limited to reduce computational cost [42]. However, with these global Cokriging techniques, convergence to a high-fidelity optimum is not guaranteed. Moreover, without controlling the location of calibration points and the optimization steps taken, local convergence of a Cokriging-based optimization method is also not ensured.

The challenge with global optimization is that to prove an algorithm finds a global optimum within a domain requires demonstrating that the sequence of trial points of the algorithm becomes dense, or that function has been evaluated in all arbitrarily small intervals within the domain [112]. Unless there exists some known, or *a priori*, bound on the smoothness of the objective function and the goal of the method is to only get "close" to the optimal design, as in [50, 51], or [94], then this requires sampling infinitely many points. With the limited exception of some cases where there are known bounds on smoothness, global optimization is likely out of reach for optimizing high-fidelity simulations due to the immense number of evaluations required. However, local optimization techniques can offer a guarantee of providing the best design in the neighborhood of a staring point, often quickly and with few evaluations. This is a common goal in engineering design, where an initial concept or layout is available and the design that will be constructed will likely not deviate significantly from that initial concept. In these cases, global exploration may be wasteful and local optimization can provide a significant performance improvement. Another advantage the local methods have is the handling of constraints. For global approaches, incorporating general non-box constraints is a challenge that is often handled heuristically. Some heuristic approaches based on EGO either estimate the probability that a point is both a minimum and that it is feasible or add a smooth penalty to the surrogate model prior to using optimization to select new high-fidelity sample locations [50, 91, 101]. In contrast, local optimization methods can simply evaluate the constraint value at the current location and can use or

approximate sensitivity information from the constraint and restrict movement to feasible directions.

1.2.2 Local MFO

One choice for local multifidelity optimization is the use of trust regions. Trust regions have a considerable history of use in optimization and are provably convergent. A thorough discussion on trust region theory and implementation is available in the book by Conn *et al.* [31]. A classic example of trust region-based optimization, which may be viewed as multifidelity optimization, is to optimize a function using its gradient and Hessian at the center of a trust region as a quadratic surrogate model of the function. The trust region then determines the allowable step length that may be taken based on how well the function can be represented by a quadratic surrogate. Figure 1-1 shows a few iterations of a trust-region optimization on a simple analytical function with quadratic approximations based at the center of the trust region. An important extension of trust regions to a general multifidelity optimization setting, one where any continuous and smooth function can be used as a low-fidelity model, is the development of trust region model management by Alexandrov *et al.* [2, 3, 5]. They showed that general low-fidelity models can be adjusted to be a surrogate model upon which optimization is performed in a manner provably convergent to an optimum of a higher-fidelity function. The requirement for the surrogate is that at the center of each trust region the surrogate model and the high-fidelity function must have both equal value and equal gradient. We refer to this required condition of equal value and equal gradient as *first-order consistency.* Alexandrov *et al.* proposed both a multiplicative and an additive correction to adjust an arbitrary low-fidelity function so that a lower-fidelity surrogate satisfying the first order consistency requirement can be created [2]. These surrogate models are rebuilt after every successful trust region iteration, which is every time the trust region iteration succeeds in reducing the high-fidelity function value. The techniques of Alexandrov *et al.* have been applied to a collection of low-fidelity models, such as reduced order models, data-fits, and space-mapped low-fidelity functions [37, 95–97]. Special attention should be brought to the first-order consistency requirement when

used with these data-fits or interpolations of the high-fidelity function. It is shown when using data-fits the first-order consistency requirement can be detrimental to the fit when enforced after the fit is created [96, 97]. However, frameworks to ensure good fits that satisfy first-order consistency are known [44].

Whenever a high-fidelity function's gradient is unavailable and cannot be estimated accurately, the first-order consistency criterion cannot be satisfied. As mentioned previously, this occurs frequently in practice. When this occurs, methods such as pattern-searches have been used for multifidelity optimization. For instance, Booker *et al.* [16] demonstrate that additional search directions generated from a surrogate model of the high-fidelity function can be added to the search lattice in a general pattern-search algorithm without violating the convergence proof of the underlying pattern-search algorithm. Within this framework, adding directions found by optimizing a surrogate model of the high-fidelity function to a pattern-search lattice may speed convergence of the pattern-search algorithm, but there is no formal proof that it will. An extension of this work is to augment the search lattice with a direction generated by minimizing a surrogate model calibrated to the high-fidelity function at the points used in the search lattice [20]. One calibration method, known as conformal space mapping, maps the design space of a low-fidelity function to the design space of the high-fidelity function by minimizing the square error between the mapped low-fidelity function and the high-fidelity function at each point in the search lattice. This technique enables the generation of calibrated models using only the high-fidelity function evaluations that are already required for provable convergence in the underlying pattern search.

Interpolation-based surrogate models are also a common choice for gradient-free optimization. Examples include that by Conn *et al.* [31, 32], Marazzi *et al.* [70], or Powell [90], and are typically based on polynomial interpolants and trust regions. These gradient-free methods do not require first-order consistency, they ensure a quality model through geometric criteria on the interpolation set which are specific to the polynomials used. Polynomial surrogates capture local function behavior sufficiently well to prove convergence; however, a more general surrogate model may also capture some global function behavior to speed convergence. Some generalizations to gradient-free surrogate-based optimization include a

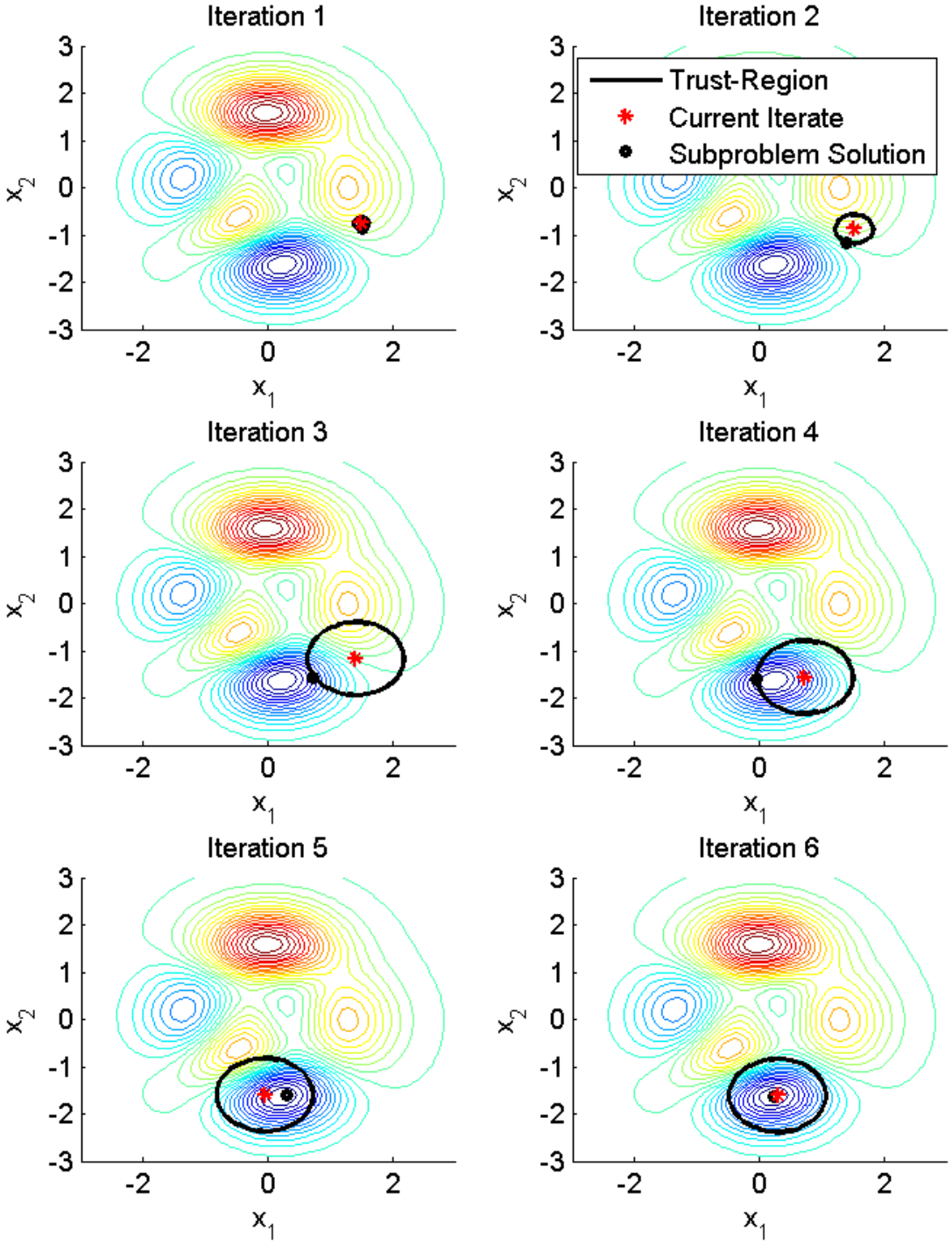


Figure 1-1: A sample of six trust-region iterations showing the current design (center of the trust region), the domain of the trust-region subproblem, and the solution of the trust-region subproblem.

proof that a trust-region algorithm is convergent provided either the error between the gradient of the function and the gradient of surrogate model is bounded by a constant times the gradient of the function [19] or provided the accuracy of the surrogate can be improved dynamically within the trust-region framework [31, Section 10.6]. Oeuvray showed that a radial basis function interpolation satisfies these criteria, provided the interpolation points satisfy certain conditions [84]. Further work by Conn *et al.* on interpolation sets [33], as well as on derivative-free trust-region methods [27, 28] led to a derivative-free trust-region method that also does not require first-order consistency. Wild *et al.* then used this work to develop a derivative-free optimization algorithm that is provably convergent to a high-fidelity optimum also using a radial basis function interpolant [116–118]. This method enables optimization on a surrogate model that can be calibrated in a region of the design space as opposed to only locally and still be provably convergent to a high-fidelity optimum.

Conn *et al.* then showed that both the error between a function and a smooth interpolation model as well as the error between the function's derivative and the interpolation model's derivative can be bounded by appropriately selecting interpolation points[33]. Conn *et al.* also proved that any interpolation model that can locally be made fully linear (defined in the next section) can be used in a provably convergent derivative-free trust-region framework [27]. Wild *et al.* then developed an algorithm to produce fully linear radial basis function interpolation models and showed that his method could be used within Conn's provably convergent optimization framework [116–118].

The gradient-based local multifidelity methods can all handle constraints with typical means: Newton methods, penalty functions, Lagrangians, etc. [5]. However, for the gradient-free methods, constraint handling can be a significant challenge. Some approaches combine either an augmented Lagrangian [58, 59, 64], exact penalty method [68], constraint filtering [7], or barrier [8] with either a pattern-search or a simplex method. Other methods use linear interpolation of both the objective function and the constraint [88, 89]. In the multifidelity setting we have a smooth surrogate model for a high-fidelity function so we are able to generate approximate sensitivity information for both the objective and the constraints to speed convergence, this is potentially a significant advantage over using pattern-search and

simplex methods.

1.2.3 Open Issues in Multifidelity Optimization

This thesis only considers local optimization and does not attempt to find the globally optimal design due to the challenges addressed. For local multifidelity optimization four major challenge themes have emerged, (i) how to rigorously and effectively combine multiple low-fidelity models to best predict the high-fidelity function behavior, (ii) how to find a high-fidelity optimum when gradient information is not available, (iii) how to find a high-fidelity optimum in the presence of potentially computationally expensive constraints, and (iv) how best to use gradient information such that surrogate models use as much information as is known about the high-fidelity function. These challenges are also significant in multidisciplinary system design, as they may emerge within the design of each subsystem and at the integrated-system level.

1.3 Multidisciplinary Optimization

Multidisciplinary Design Optimization (MDO) approaches are tasked with optimizing a complex system that requires evaluating the interactions of multiple disciplines or equivalently multiple subsystems. To explain some of the difficulties in MDO, Figure 1-2 presents a design structure matrix (DSM) which demonstrates feed-forward and feedback between disciplines in a multidisciplinary system. All disciplines are placed on a diagonal, feed-forward is represented as connections above the diagonal, and feedback is indicated by connections below the diagonal. The feedback from Discipline 2 to Discipline 1 means that there must be some iterative solution between Disciplines 1 and 2 to find the resulting output that Discipline 3 requires, and this iterative solution must be done for each set of design variables tested during optimization. This can be further complicated when both Disciplines 1 and 2 have their own constraints that must be satisfied for their individual feasibility. The seminal publication [34], *Problem Formulation for Multidisciplinary Optimization*, defined multidisciplinary feasibility as when (1) all single disciplines are individual feasible and (2)

the input to each discipline corresponds to the output of the other via the interdisciplinary mapping. With respect to Figure 1-2, this means that all the constraints within Disciplines 1, 2 and 3 are satisfied, and that when an output of one discipline is an input to another discipline, the value output and the value input must be equal. This corresponds to the iteration between Discipline 1 and Discipline 2 having been solved. Solving for all of the interacting variables between disciplines such that the system is multidisciplinary feasible is a called a multidisciplinary analysis (MDA).

The remainder of this section presents common MDO formulations. The discussion highlights the degree to which the formulations enable multifidelity methods, are parallelizable, and whether or not the methods have a mathematical proof of convergence to an optimal system design. In terms of parallelization of the system optimization there are three ways in which an optimization method can be parallelized, (i) by evaluating the function in parallel, e.g., evaluating each required function evaluation for a finite difference calculation at the same time, (ii) performing the linear algebra and optimization calculations in parallel, and (iii) decomposing the optimization into smaller subproblems that can each be performed in parallel [102]. This thesis will only consider parallelizability of the optimization itself and not the linear algebra, i.e., only techniques (i) and (iii) are considered.

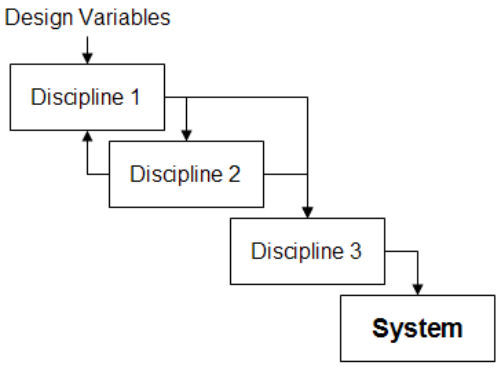


Figure 1-2: Block diagram for a system with feed-forward from discipline 1 to 2 and 3, and feedback from discipline 2 to 1. The system objective is only a function of discipline 3.

1.3.1 Fundamental MDO Methods

Another major contribution from the paper *Problem Formulation for Multidisciplinary Optimization* is to define three fundamental multidisciplinary optimization approaches which allow for the use of nonlinear programming techniques to solve system optimization problems. The first formulation is called the Multidisciplinary Feasible (MDF) formulation, and

in this formulation evaluating the system objective function value for any given set of design variables requires a full MDA. This formulation can be computationally intensive as calculations of the objective function used for sensitivity information also require solving a complete MDA. The second approach is called the All-At-Once (AAO) formulation, and with this approach no feasibility is ensured until convergence. All of the analysis variables and coupling variables are treated as additional design variables, and all of the analyses and disciplinary couplings are treated as constraints. In this formulation the optimization routine is required to solve the nonlinear systems of constraints in order to find a feasible solution; however, the full MDA is not required in a large portion of the design space when feasibility is impossible. The final approach presented is the Individual Discipline Feasible (IDF) formulation. In the IDF approach individual discipline feasibility is ensured; however, the multidisciplinary feasibility is left to the optimizer as a constraint. The IDF approach is flexible and numerous formulations are posed [34].

Using the MDF approach, multifidelity optimization and parallelization are only possible at the system level. Surrogate models of the system performance can be created from full MDAs at many designs. In addition, each MDA may be carried out simultaneously, but the iterative solution of finding feasible designs is only parallelizable if a system exhibits atypical subsystem couplings. Convergence of an MDF optimization only relies on the capabilities of the iterative technique used in the MDA and on the nonlinear programming (NLP) solver used in the optimization, therefore this technique can typically be mathematically guaranteed to find an optimal system given that the assumptions of the iterative solution method and NLP solver are met. The essential drawback of this method is when each subsystem contains a high-fidelity simulation; in this case, the iterative solve in the MDA can be prohibitively expensive and preclude the use of the MDF formulation.

The IDF approach offers a potential benefit in that the subsystems only communicate through the system-level optimization problem. This enables parallelizing the overall system-level optimization problem into separately finding feasible subsystem designs, and also allows the possibility to evaluate the full system performance in parallel. There is also the potential to use multifidelity methods for the system-level and subsystem-level analyses. The conver-

gence of this technique only relies on convergence properties of the NLP solver used for the optimization, so in many cases the approach can be proven to converge.

The AAO approach, also known as Simultaneous Analysis and Design (SAND), has the issue that all discipline analyses need to be converted to residual form, which is atypical of customary disciplinary analyses. However, it is theoretically possible to evaluate all disciplines and the system objective in parallel and to use multifidelity methods to solve the equality constrained optimization problem. The convergence of this method is only dependent on the convergence properties of the NLP solver. In fact, Alexandrov *et al.* developed provably convergent trust-region methods to solve the AAO formulation as a multilevel problem [4]. Their algorithm separates the constraints into blocks and finds solutions that are in the null space of the constraint block Jacobians, and then updates the trust-region according to a merit function that includes constraint violations. The advantage of the AAO formulation is that the problem can be solved without using a constraint following algorithm which given the large number of constraints in an MDO problem, should prove fruitful.

1.3.2 Open Issues in Multidisciplinary Optimization

Multidisciplinary system design optimization is a challenge due to the nonlinear couplings between subsystems and the expense of estimating the performance of subsystems. These challenges suggest that a parallelizable framework for MDO is a necessity, and that rigorous multifidelity optimization techniques should be possible within that framework. Moreover, it is common that the sensitivity of a subsystem's performance to its design variables will be available for some but not for all subsystems. Therefore MDO frameworks should also allow for gradient-free optimization, but must be able to exploit gradient information when it is available in order to rapidly find good system designs.

When considering the single-level MDO frameworks, MDF, IDF, and AAO, the underlying NLP solver can provide a guarantee of finding an optimal design. As NLP solvers with a convergence guarantee are available for both gradient-free and gradient-exploiting optimization problems, this guarantee applies in both cases. However, the ability to mix gradient-free and gradient-exploiting handling for individual disciplines and the paralleliz-

ability of these basic frameworks is limited. The multi-level MDO methods, Collaborative Optimization [18, 60], Bilevel Integrated System Synthesis [107–109], Concurrent Subspace Optimization [106, 119], and Analytical Target Cascading [78, 79, 113], are more parallelizable, but fast convergence or even just convergence for the case of a non-convex optimization problem is not guaranteed. One benefit of these multilevel methods is that the formulations based on response surfaces offer a chance to use either gradient-free or gradient-exploiting handling for the subsystems as appropriate. For example, BLISS [108] requires global sensitivity information for each subproblem optimization, but BLISS2000 [109] approximates this sensitivity information with quadratic response surfaces. Therefore novel MDO methods that enable a mixture of gradient-free and gradient-exploiting optimization methods and are parallelizable are a clear need. Multifidelity optimization techniques offer a possible solution to this challenge. Surrogate-based multifidelity methods typically create smooth surrogate models for each discipline's performance with known bounds for the error in surrogate-based sensitivity information. This surrogate sensitivity information can exploit high-fidelity gradients when available, but does not require that information. In addition, these surrogate models can be used to optimize the system and likely will save a significant number of discipline- and system-level performance evaluations.

1.4 Research Objectives

As mentioned, the field of MDO will benefit from system design frameworks that exploit low-resource low-fidelity simulations, are parallelizable, do not require sensitivity information, but can exploit sensitivity information if it is available. Multifidelity optimization nested within a system design framework offers these possibilities. Therefore the objectives of this thesis are:

1. To propose and demonstrate a multifidelity optimization approach that does not require high-fidelity derivative information and is applicable to nonlinear constrained design optimization problems.

2. To propose and demonstrate a multifidelity optimization approach that can exploit

high-fidelity derivative information when available.

3. To develop a framework for distributed optimization of a system with multiple fidelity levels across multiple disciplines that does not require gradient information, but that can exploit it when available.

4. To apply the multifidelity optimization approaches and decomposition framework to a multidisciplinary aerostructural aircraft design problem.

1.5 Thesis Outline

To accomplish the research objectives, this thesis presents an unconstrained multifidelity optimization algorithm that is provably convergent to a high-fidelity optimum and does not require high-fidelity gradient estimates in Chapter 2. This chapter also addresses combining multiple low-fidelity models into one best prediction of the high-fidelity function. Chapter 3 extends these multifidelity techniques to constrained optimization, and develops a gradient-free constrained multifidelity optimization algorithm. Chapter 4 addresses how to best use gradients in multifidelity optimization and presents a model calibration method, similar to the gradient-free methods, but that exploits high-fidelity gradient information. Chapter 5 then presents two MDO methods that enable both multifidelity optimization and parallelization of a system design problem. Chapter 6 then demonstrates the MDO methods on an aerostructural multidisciplinary system design problem, and addresses optimization when function evaluation failures are prevalent. Conclusions and recommendations for future research are given in Chapter 7. A summary of all algorithms developed in this thesis is presented as Table 1.1.

Disciplines	Algorithm	Section	Constraints	Gradients Req'd	Multifidelity	Failures
One	2.1	Section 2.3	None	None	Obj.	Yes
	3.1	Section 3.1	Yes	Constraint	Obj.	Yes
	3.2	Section 3.2	Yes	None	Obj. & Con.	Yes
	4.1	Section 4.1	Yes	All	Obj. & Con.	Yes*
Multiple	5.1	Section 5.2	Yes	None	Obj. & Con.	Yes
	5.2	Section 5.3.2	Equality	None	Obj. & Con.	Yes
	5.3	Section 5.3.4	Equality	All	Obj. & Con.	Yes*

Table 1.1: A summary of the algorithms developed in this thesis and whether or not they are multidisciplinary, constrained, gradient-based, multifidelity, or support function evaluation failures. An asterisk indicates that the capability is not demonstrated in this thesis.

Chapter 2

Unconstrained Gradient-free Multifidelity Optimization

This chapter presents a provably convergent multifidelity optimization algorithm for unconstrained problems that does not require high-fidelity gradients. The techniques presented in this chapter form the basis for Chapters 3 and 5. The method uses a radial basis function interpolation to capture the error between a high-fidelity function and a low-fidelity function. The error interpolation is added to the low-fidelity function to create a surrogate model of the high-fidelity function in the neighborhood of a trust region. When appropriately distributed spatial calibration points are used, the low-fidelity function and radial basis function interpolation generate a fully linear model. This condition is sufficient to prove convergence in a trust-region framework. In the case when there are multiple lower-fidelity models, the predictions of all calibrated lower-fidelity models can be combined with a maximum likelihood estimator constructed using Kriging variance estimates from the radial basis function models. This procedure allows for flexibility in sampling lower-fidelity functions, does not alter the convergence proof of the optimization algorithm, and is shown to be robust to poor low-fidelity information. The algorithm is compared with a single-fidelity quasi-Newton algorithm and two first-order consistent multifidelity trust-region algorithms on an analytical test problem and a more complex supersonic airfoil design problem.

This chapter first presents an overview of a derivative-free trust-region algorithm using

fully linear models in Section 2.1. It then discusses an approach to build fully linear models using radial basis functions (RBFs), and presents our extension to the case of multifidelity model calibration in Section 2.2. Section 2.3 provides an overview of the computational implementation of the method and suggests a way to incorporate the method of generating fully linear models with flexible Bayesian model calibration techniques. Section 2.4 demonstrates the multifidelity optimization algorithm on an analytical example and a supersonic airfoil design problem. Section 2.5 then develops the extension of our approach to the case when there are multiple lower-fidelity models. Finally, Section 2.6 summarizes the findings and contributions in this chapter.

2.1 Trust-Region-Based Multifidelity Optimization

We consider a setting where we have two (or more) models that represent the physical system of interest: a high-fidelity function that accurately estimates system metrics of interest but is expensive to evaluate, and a low-fidelity function with lower accuracy but cheaper evaluation cost. We define our high-fidelity function as $f_{\text{high}}(\mathbf{x})$ and our low-fidelity function as $f_{\text{low}}(\mathbf{x})$, where $\mathbf{x} \in \mathbb{R}^n$ is the vector of n design variables. Our goal is to solve the unconstrained optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} f_{\text{high}}(\mathbf{x}), \tag{2.1}$$

using information from evaluations of $f_{\text{low}}(\mathbf{x})$ to reduce the required number of evaluations of $f_{\text{high}}(\mathbf{x})$.

We use the derivative-free trust-region algorithm of Conn *et al.* [27] to solve (2.1). From an initial design vector $\mathbf{x}_0$, the trust-region method generates a sequence of design vectors that each reduce the high-fidelity function value, where we denote $\mathbf{x}_k$ to be this design vector on the kth trust-region iteration. Following the general Bayesian calibration approach in [55], we define $e_k(\mathbf{x})$ to be a model of the error between the high- and low-fidelity functions on

the kth trust-region iteration, and we construct a surrogate model $m_k(\mathbf{x})$ for $f_{\text{high}}(\mathbf{x})$ as

$$m_k(\mathbf{x}) = f_{\text{low}}(\mathbf{x}) + e_k(\mathbf{x}) \approx f_{\text{high}}(\mathbf{x}). \tag{2.2}$$

We define the trust region at iteration k, $\mathcal{B}_k$, to be the region centered at $\mathbf{x}_k$ with size Δ_k,

$$\mathcal{B}_k = \{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_k\| \leq \Delta_k\}, \tag{2.3}$$

where any norm can be used, provided there exist constants k_1 and k_2 such that $\|\cdot\|_2 \leq k_1\|\cdot\|$ and $\|\cdot\| \leq k_2\|\cdot\|_2$. The trust-region is a portion of the design space where the surrogate model is considered an accurate representation of the high-fidelity function. The accuracy is ensured with a local calibration, in this case updating the error model, $e_k(\mathbf{x})$. To improve the high-fidelity function value, the surrogate model is minimized, or approximately minimized, within the trust region. The size of the trust region is updated dynamically based on the quality of the surrogate model estimate of the high-fidelity function value; good estimates cause the size of the trust region to increase and poor estimates cause the size of the trust region to decrease. Therefore, the trust region serves as both a means to control the size of the allowable optimization step and as a means to control the accuracy of the surrogate model predictions. For a detailed discussion of trust-region algorithms see [31] or [83].

If the high-fidelity function $f_{\text{high}}(\mathbf{x})$ and the surrogate models $m_k(\mathbf{x})$ satisfy certain conditions, this framework provides a guarantee of convergence to a stationary point of the high-fidelity function $f_{\text{high}}(\mathbf{x})$. Specifically, the convergence proof requires that the high-fidelity function $f_{\text{high}}(\mathbf{x})$ be (i) continuously differentiable, (ii) have a Lipschitz continuous derivative, and (iii) be bounded from below within a region of a relaxed level-set, $\mathcal{L}(\mathbf{x}_0)$, defined as

$$L(\mathbf{x}_0) = \{\mathbf{x} \in \mathbb{R}^n : f_{\text{high}}(\mathbf{x}) \leq f_{\text{high}}(\mathbf{x}_0)\} \tag{2.4}$$

$$B(\mathbf{x}_k) = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x} - \mathbf{x}_k\| \leq \Delta_{\max}\} \tag{2.5}$$

$$\mathcal{L}(\mathbf{x}_0) = L(\mathbf{x}_0) \bigcup_{\mathbf{x_k} \in L(\mathbf{x}_0)} B(\mathbf{x}_k), \tag{2.6}$$

where $\Delta_{\max}$ is the maximum allowable trust-region size. The relaxed level-set is required because the trust-region algorithm may attempt to evaluate the high-fidelity function at points outside of the level set at $\mathbf{x}_0$. The convergence proof further requires that the surrogate models $m_k(\mathbf{x})$ are *fully linear*, where the following definition of a fully linear model is from Conn *et al.* [27]:

Definition 2.1 *Let a function $f_{high}(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ that satisfies the conditions (i)–(iii) above, be given. A set of model functions $\mathcal{M} = \{m : \mathbb{R}^n \to \mathbb{R}, m \in C^1\}$ is called a fully linear class of models if the following occur:*

There exist positive constants κ_f, κ_g and κ_{blg} such that for any $\mathbf{x} \in L(\mathbf{x}_0)$ and $\Delta_k \in (0, \Delta_{\max}]$ there exists a model function $m_k(\mathbf{x})$ in $\mathcal{M}$ with Lipschitz continuous gradient and corresponding Lipschitz constant bounded by k_{blg}, and such that the error between the gradient of the model and the gradient of the function satisfies

$$\|\nabla f_{high}(\mathbf{x}) - \nabla m_k(\mathbf{x})\| \leq \kappa_g \Delta_k \quad \forall \mathbf{x} \in \mathcal{B}_k \tag{2.7}$$

and the error between the model and the function satisfies

$$|f_{high}(\mathbf{x}) - m_k(\mathbf{x})| \leq \kappa_f \Delta_k^2 \quad \forall \mathbf{x} \in \mathcal{B}_k. \tag{2.8}$$

Such a model $m_k(\mathbf{x})$ is called fully linear on $\mathcal{B}_k$[27].

At iteration k, the trust-region algorithm solves the subproblem

$$\begin{aligned} \min_{\mathbf{s}_k} \quad & m_k(\mathbf{x}_k + \mathbf{s}_k) \\ \text{s.t.} \quad & \|\mathbf{s}_k\| \leq \Delta_k \end{aligned} \tag{2.9}$$

to determine the trust-region step $\mathbf{s}_k$. If the subproblem is not solved exactly, the minimum requirement is that the steps found in the trust-region subproblem must satisfy a sufficient decrease condition. At iteration k, we require that the model $m_k(\mathbf{x})$ have a finite upper bound on the norm of its Hessian matrix evaluated at $\mathbf{x}_k$: $\|H_k(\mathbf{x}_k)\| \leq \kappa_{bhm} < \infty$. This bound on

the Hessian may be viewed as a bound on the Lipschitz constant of the gradient of $m_k(\mathbf{x}_k)$ [27]. The sufficient decrease condition requires the step to satisfy the fraction of Cauchy decrease. As given in [27] and [116], this requires that for some constant, $\kappa_{FCD} \in (0,1)$, the step $\mathbf{s}_k$ satisfies

$$m_k(\mathbf{x}_k) - m_k(\mathbf{x}_k + \mathbf{s}_k) \geq \frac{\kappa_{FCD}}{2} \|\nabla m_k(\mathbf{x}_k)\| \min\left[\frac{\|\nabla m_k(\mathbf{x}_k)\|}{\kappa_{bhm}}, \Delta_k\right]. \tag{2.10}$$

The high-fidelity function f_{high} is then evaluated at the new point, $\mathbf{x}_k + \mathbf{s}_k$. We compare the actual improvement in the function value with the improvement predicted by the model by defining

$$\rho_k = \frac{f_{\text{high}}(\mathbf{x}_k) - f_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k)}{m_k(\mathbf{x}_k) - m_k(\mathbf{x}_k + \mathbf{s}_k)}. \tag{2.11}$$

The trial point is accepted or rejected according to

$$\mathbf{x}_{k+1} = \begin{cases} \mathbf{x}_k + \mathbf{s}_k & \text{if } \rho_k > 0 \\ \mathbf{x}_k & \text{otherwise.} \end{cases} \tag{2.12}$$

If the step is accepted, then the trust region is updated to be centered on the new iterate $\mathbf{x}_{k+1}$. The size of the trust region, Δ_k, must now be updated based on the quality of the surrogate model prediction. The size of the trust region is increased if the surrogate model predicts the change in the function value well and the trust region is contracted if the model predicts the function change poorly. Specifically, we update the trust region size using

$$\Delta_{k+1} = \begin{cases} \min\{\gamma_1 \Delta_k, \Delta_{\max}\} & \text{if } \rho_k \geq \eta \\ \gamma_0 \Delta_k & \text{if } \rho_k < \eta, \end{cases} \tag{2.13}$$

where $0 < \eta < 1$, $0 < \gamma_0 < 1$, and $\gamma_1 > 1$.

A new fully linear model, $m_{k+1}(\mathbf{x})$, is then built using the radial basis function interpolation approach described in the next section. That surrogate model will be fully linear on

a region $\mathcal{B}_{k+1}$ having center $\mathbf{x}_{k+1}$ and size Δ_{k+1}.

To check for algorithm termination, the gradient of the model is computed at $\mathbf{x}_{k+1}$. If $\|\nabla m_{k+1}(\mathbf{x}_{k+1})\| > \epsilon$ for a small ϵ, the trust-region algorithm will continue to iterate, solving the next subproblem on the new trust region, $\mathcal{B}_{k+1}$, with the updated model, $m_{k+1}(\mathbf{x})$. However, if $\|\nabla m_{k+1}(\mathbf{x}_{k+1})\| \leq \epsilon$, we need to confirm that the algorithm has reached a stationary point of $f_{\text{high}}(\mathbf{x})$. If gradients of the high-fidelity function are available, one could evaluate if $\|\nabla f_{\text{high}}(\mathbf{x}_{k+1})\| \leq \epsilon$ directly. In the general derivative-free case, we use the condition in Eq. 2.7, and show that if $\Delta_{k+1} \to 0$ then $\|\nabla f_{\text{high}}(\mathbf{x}_{k+1}) - \nabla m_{k+1}(\mathbf{x}_{k+1})\| \to 0$. In practice we achieve this by updating the model to be fully linear on a trust region with size some fraction, $0 < \alpha < 1$, of Δ_{k+1}. This process continues until either $\|\nabla m_{k+1}(\mathbf{x}_{k+1})\| > \epsilon$, in which case the trust-region algorithm will continue with the updated model and updated Δ_{k+1}, or $\Delta_{k+1} \leq \epsilon_2$, for a small ϵ_2, which terminates the algorithm. This process of checking for convergence is referred to as the criticality check in Conn *et al.* [27].

2.2 Interpolation-Based Multifidelity Models

In this section we discuss a method of creating surrogate models that satisfy the conditions for provable convergence presented in Section 2.1. This section first presents an overview of the radial basis function (RBF) interpolation approach of Wild *et al.* [117], where the interpolation points are chosen so that the resulting model is fully linear. Next, we present an extension of this approach to the case of multifidelity models.

Define $\mathbf{d}_j$ to be the jth point in the set of designs at which the high-fidelity and low-fidelity functions have been sampled. Define $\mathbf{y}_i$ to be the vector from the current iterate (i.e., center of the current trust region), $\mathbf{x}_k$, to any sample point inside or within the vicinity of the current trust region, $\mathbf{d}_i$, that is selected to be an interpolation point. Also define $\mathcal{Y}$ to be the set of the zero vector (i.e., $\mathbf{y} = \mathbf{0}$ corresponds to $\mathbf{x}_k$) and all of the vectors $\mathbf{y}_i$. This notation is shown graphically in Figure 2-1.

The RBF interpolation is defined so that by construction the surrogate model is equal to the high-fidelity function at all interpolation points. That is, the error between the high-

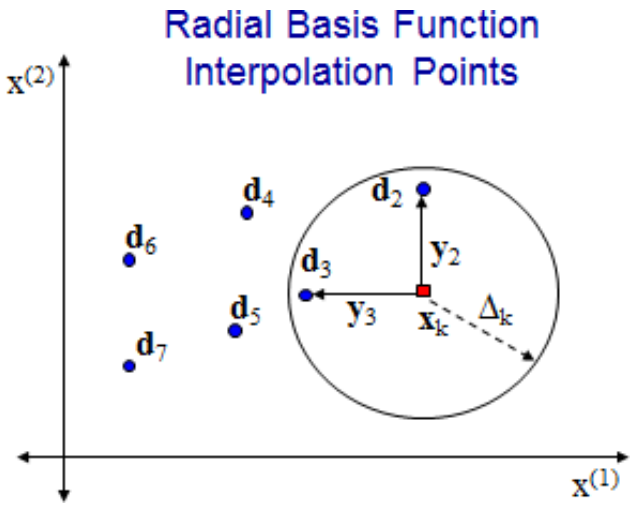


Figure 2-1: Graphical representation of the notation used to define points and vectors in and around the trust region.

and low-fidelity functions is interpolated exactly for all points defined by the vectors within $\mathcal{Y}$,

$$e_k(\mathbf{x}_k + \mathbf{y}_i) = f_{\text{high}}(\mathbf{x}_k + \mathbf{y}_i) - f_{\text{low}}(\mathbf{x}_k + \mathbf{y}_i) \quad \forall \mathbf{y}_i \in \mathcal{Y}. \tag{2.14}$$

The RBF interpolation has the form

$$e_k(\mathbf{x}) = \sum_{i=1}^{|\mathcal{Y}|} \omega^{(i)} \phi\left(\|\mathbf{x} - \mathbf{x}_k - \mathbf{y}_i\|\right) + \sum_{i=1}^{n+1} \nu^{(i)} \pi^{(i)}(\mathbf{x} - \mathbf{x}_k), \tag{2.15}$$

where ϕ is any positive definite, twice continuously differentiable RBF with $\phi'(0) = 0$, and the second term in (2.15) represents a linear tail, where π_i denotes the i^{th} component of the vector $\Pi(\mathbf{x} - \mathbf{x}_k) = [1 \ (\mathbf{x} - \mathbf{x}_k)]^\top$. The coefficients $\omega^{(i)}$ and $\nu^{(i)}$ represent the RBF interpolation, and are found by the QR-factorization technique of Wild *et al.* [117]. In order for the model to be fully linear, the RBF coefficients $\omega^{(i)}$ and $\nu^{(i)}$ must be bounded in magnitude. This is achieved by using the interpolation point selection method in Wild *et al.* [117]. The process can be summarized as follows. First, the existing high-fidelity sample points, $\mathbf{d}_j$, in the vicinity of the trust region are tested to see if there are $n + 1$ affinely independent vectors. (This test is carried out using the singular value decomposition of a

matrix containing as columns the vectors $\mathbf{y}_j$ and is computationally inexpensive compared to the typical cost of a high-fidelity solve.) If fewer than $n+1$ affinely independent points are found, additional high-fidelity function evaluations are required to generate additional interpolation points. Second, we test all other points $\mathbf{d}_j$ at which the high-fidelity function value is known, by measuring the impact of their addition as interpolation points on the RBF coefficients $\omega^{(i)}$ and $\nu^{(i)}$. Those points that ensure the RBF coefficients remain bounded are used as additional interpolation points to update the model. Wild proved that this RBF interpolation model construction algorithm produces a fully linear model for a function satisfying conditions (i) and (ii) above [116]. An illustration of the calibrated models resulting from this process is presented as Figure 2-2.

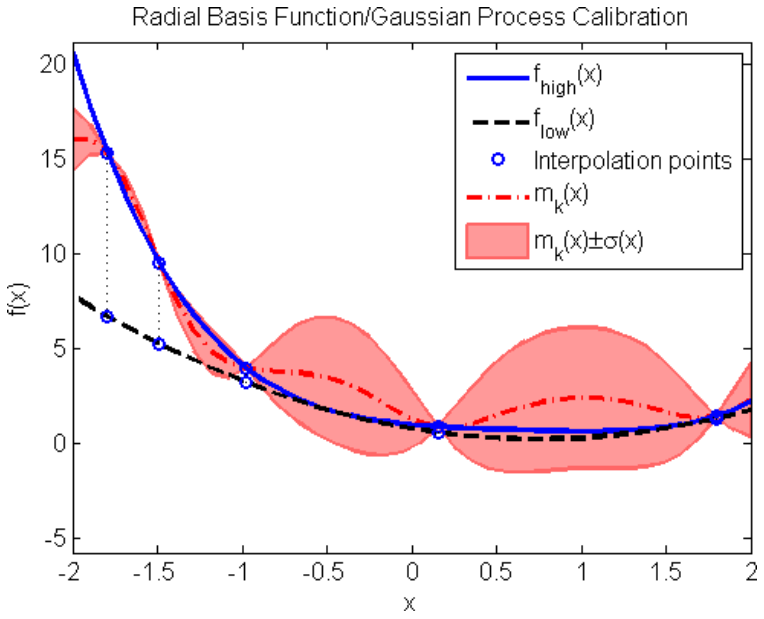


Figure 2-2: An illustration of the surrogate model of a high-fidelity function created using a radial basis function interpolation of the error between a high- and low-fidelity function. The high-fidelity function is in blue, the low-fidelity function is in black, the fully linear surrogate model is in red, the uncertainty estimate of the surrogate model is in pink, and the calibration points are circled.

In order for Wild's interpolation approach to be applicable in our Bayesian calibration setting, we require that the error function defined by $f_{\text{high}}(\mathbf{x}) - f_{\text{low}}(\mathbf{x})$ satisfies conditions (i) and (ii) above. Condition (i), that the function is continuously differentiable, is satisfied if both $f_{\text{high}}(\mathbf{x})$ and $f_{\text{low}}(\mathbf{x})$ are continuously differentiable. To establish condition (ii), that

the derivative of $f_{\text{high}}(\mathbf{x}) - f_{\text{low}}(\mathbf{x})$ is Lipschitz continuous, we require that both $\nabla f_{\text{high}}(\mathbf{x})$ and $\nabla f_{\text{low}}(\mathbf{x})$ be Lipschitz continuous in the relaxed level set defined in Eq. (2.6). For the high-fidelity function we require

$$\frac{\|\nabla f_{\text{high}}(\mathbf{x}_1) - \nabla f_{\text{high}}(\mathbf{x}_2)\|}{\|\mathbf{x}_1 - \mathbf{x}_2\|} \leq \kappa_{\text{high}} \quad \forall \mathbf{x}_1, \mathbf{x}_2 \in \mathcal{L}(\mathbf{x}_0), \tag{2.16}$$

and for the low-fidelity function,

$$\frac{\|\nabla f_{\text{low}}(\mathbf{x}_1) - \nabla f_{\text{low}}(\mathbf{x}_2)\|}{\|\mathbf{x}_1 - \mathbf{x}_2\|} \leq \kappa_{\text{low}} \quad \forall \mathbf{x}_1, \mathbf{x}_2 \in \mathcal{L}(\mathbf{x}_0), \tag{2.17}$$

with Lipschitz constants κ_{high} and κ_{low}, respectively. Therefore by the triangle inequality, we obtain

$$\frac{\|[\nabla f_{\text{high}}(\mathbf{x}_1) - \nabla f_{\text{low}}(\mathbf{x}_1)] - [\nabla f_{\text{high}}(\mathbf{x}_2) - \nabla f_{\text{low}}(\mathbf{x}_2)]\|}{\|\mathbf{x}_1 - \mathbf{x}_2\|} \leq \kappa_{\text{high}} + \kappa_{\text{low}} \quad \forall \mathbf{x}_1, \mathbf{x}_2 \in \mathcal{L}(\mathbf{x}_0), \tag{2.18}$$

where the Lipschitz constant of the difference is bounded by $\kappa_{\text{high}} + \kappa_{\text{low}}$. Accordingly, the convergence proof for the trust-region algorithm used in Conn *et al.* [27] holds.

2.3 Numerical Implementation of Algorithms

This section presents an overview of the numerical implementation of the multifidelity optimization algorithm and suggests a manner in which the method of Wild *et al.* [118] to generate fully linear models can be used in a flexible Bayesian calibration setting. The first subsection, Section 2.3.1, implements the the trust region based optimization algorithm presented in Section 2.1. Whenever creation of a new fully linear model is needed, the method discussed in Section 2.2 is implemented using the algorithm presented in Section 2.3.2.

2.3.1 Trust Region Implementation

Algorithm 2.1 provides an overview of the numerical implementation of the trust-region optimization method presented in Section 2.1. For each trust-region iteration, the algorithm guarantees that a step is found that satisfies the fraction of Cauchy decrease, Eq. 2.10. The algorithm only samples the high-fidelity function when necessary for convergence, and it stores all high-fidelity function evaluations in a database so that design points are never re-evaluated. Whenever an updated surrogate model is needed, the model generation method described in the following subsection creates a surrogate model using this database of high-fidelity function evaluations together with new high-fidelity evaluations when necessary. The parameters of the trust-region optimization algorithm were defined in Section 2.1, while recommended values and sensitivity of results to those values will be presented in Section 2.4.

2.3.2 Fully Linear Bayesian Calibration Models

Algorithm 2.2 presents the numerical implementation of the method to generate fully linear surrogate models, allowing for a Bayesian maximum likelihood estimate of the RBF correlation length. The RBF models used in Bayesian model calibration have a length scale parameter that provides flexibility. For instance, in the Gaussian RBF model, $\phi(r) = e^{-r^2/\xi^2}$, the parameter ξ is a variable length scale that can alter the shape of the correlation structure. If the interpolation errors are assumed to have a Gaussian distribution, then a maximum likelihood estimate can be used to estimate the value of ξ that best represents the data [69, 93]. Therefore, our process to generate a fully linear surrogate model uses the method of Wild *et al.* [117] on a set of candidate length scales, $\xi_i \in \{\xi_1, \ldots, \xi_n\}$. A fully linear model is constructed for each candidate length scale, and the likelihood of each length scale is computed. The trust region algorithm then uses the surrogate model constructed with ξ^*, where ξ^* is chosen as the value of ξ corresponding to the maximum likelihood. This maximum likelihood approach can improve the model calibration, and also provides flexibility in selecting sample points in the extension to the case when there are multiple lower-fidelity models (as will be

Algorithm 2.1: Trust-Region Algorithm for Iteration k

1: Compute a step, $\mathbf{s}_k$, that satisfies the fraction of Cauchy decrease requirement, Eq. 2.10, for the trust-region subproblem, by solving

$$\begin{aligned} \min_{\mathbf{s}_k} \quad & m_k(\mathbf{x}_k + \mathbf{s}_k) \\ \text{s.t.} \quad & \|\mathbf{s}_k\| \leq \Delta_k. \end{aligned}$$

2: If $f_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k)$ has not been evaluated previously, evaluate the high-fidelity function at that point.

2a: Store $f_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k)$ in database.

3: Compute the ratio of actual improvement to predicted improvement,

$$\rho_k = \frac{f_{\text{high}}(\mathbf{x}_k) - f_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k)}{m_k(\mathbf{x}_k) - m_k(\mathbf{x}_k + \mathbf{s}_k)}.$$

4: Accept or reject the trial point according to ρ_k,

$$\mathbf{x}_{k+1} = \begin{cases} \mathbf{x}_k + \mathbf{s}_k & \text{if } \rho_k > 0 \\ \mathbf{x}_k & \text{otherwise.} \end{cases}$$

5: Update the trust region size according to ρ_k,

$$\Delta_{k+1} \quad = \quad \begin{cases} \min\{\gamma_1 \Delta_k, \Delta_{\max}\} & \text{if } \rho_k \geq \eta \\ \gamma_0 \Delta_k & \text{if } \rho_k < \eta. \end{cases}$$

6: Create a new model $m_{k+1}(\mathbf{x})$ that is fully linear on $\{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_{k+1}\| \leq \Delta_{k+1}\}$ using Algorithm 2.2.

7: Check for convergence: if $\|\nabla m_{k+1}(\mathbf{x}_{k+1})\| > \epsilon$, algorithm has not converged—go to step 1. Otherwise,

7a: While $\|\nabla m_{k+1}(\mathbf{x}_{k+1})\| \leq \epsilon$ and $\Delta_{k+1} > \epsilon_2$,

7b: Reduce the trust region size, $\alpha \Delta_{k+1} \to \Delta_{k+1}$.

7c: Update model $m_{k+1}(\mathbf{x})$ to be fully linear on $\{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_{k+1}\| \leq \Delta_{k+1}\}$ using Algorithm 2.2.

discussed in Section 2.5).

Algorithm 2.2: Create Fully Linear Models Allowing Maximum Likelihood Correlation Lengths

1: Compute the likelihood for all RBF correlation lengths, $\xi_i \in \{\xi_1, \ldots, \xi_n\}$ with steps 2-5.
2: Generate a set of $n+1$ affinely independent points in the vicinity of the trust region:
  2a: Set $\mathbf{y}_1 = \mathbf{0}$, and add $\mathbf{y}_1$ to the set of calibration vectors $\mathcal{Y}$.
  2b: Randomly select any high-fidelity sample point, $\mathbf{d}_2$, within the current trust region and add the vector $\mathbf{y}_2 = \mathbf{d}_2 - \mathbf{x}_k$ to $\mathcal{Y}$.
  2c: For all unused high-fidelity sample points within the current trust region, add the vector $\mathbf{y} = \mathbf{d}_j - \mathbf{x}_k$ to $\mathcal{Y}$ if the projection of $\mathbf{y}$ onto the nullspace of the span of the vectors in the current $\mathcal{Y}$ is greater than $\theta_1 \Delta_k$, $0 < \theta_1 < 1$.
  2d: If fewer than $n+1$ vectors are in the calibration set, repeat step 2c allowing a larger search region of size $\theta_3 \Delta_k$, $\theta_3 > 1$.
  2e: While fewer than $n+1$ vectors are in $\mathcal{Y}$,
  2f: Evaluate the high-fidelity function at a point within the nullspace of the span of the vectors in $\mathcal{Y}$ and add $\mathbf{y} = \mathbf{d} - \mathbf{x}_k$ to $\mathcal{Y}$.
  2g: Store the results of all high-fidelity function evaluations in the database.
3: Consider the remaining unused high-fidelity sample points within a region centered at the current iterate with size $\theta_4 \Delta_k$, $\theta_4 > 1$. Add points so that the total number of interpolation points does not exceed p_{max}, the RBF coefficients remain bounded, and the surrogate model is fully linear (using, for example, the AddPoints algorithm of Wild *et al.* [118]).
4: Compute the RBF coefficients using the QR factorization technique of Wild *et al.* [117].
5: If only $n+1$ vectors are in the calibration set, $\mathcal{Y}$, assign the likelihood of the current correlation length, ξ_i, to $-\infty$. Otherwise compute the likelihood of the RBF interpolation using standard methods [69, 93].
6: Select the ξ_i with the maximum likelihood.
  6a: If the maximum likelihood is $-\infty$ choose the largest ξ_i. This model typically occurs if exactly $n+1$ points are in the neighborhood of the trust region and corresponds to a linear regression of the high-fidelity function at the calibration points included in $\mathcal{Y}$, but it still satisfies conditions for convergence.
7: Return the set of calibration vectors $\mathcal{Y}$, RBF coefficients, and updated database of high-fidelity function evaluations.

2.4 Multifidelity Optimization Examples

This section demonstrates the multifidelity optimization scheme for two examples. The first is an analytical example considering the Rosenbrock function and the second is a supersonic

airfoil design problem.

2.4.1 Rosenbrock Function

The first example multifidelity optimization example is the Rosenbrock function,

$$\min_{\mathbf{x}\in\mathbb{R}^2} f_{\text{high}}(\mathbf{x}) = \left(x_2 - x_1^2\right)^2 + \left(1 - x_1\right)^2. \tag{2.19}$$

The minimum of the Rosenbrock function is at $x^* = (1, 1)$ and $f(x^*) = 0$. For this simple analytical function with only two design variables, we do not expect the multifidelity method to significantly outperform a quasi-Newton method, however, the example is useful to illustrate the multifidelity approach and to demonstrate effects of algorithm parameters. Table 2.1 presents the number of high-fidelity function evaluations required to optimize the Rosenbrock function using a variety of low-fidelity functions. All of the low-fidelity functions have a different minimum than the Rosenbrock function, with the exception of the case when the low-fidelity function is set equal to the Rosenbrock function, corresponding to a perfect low-fidelity function. For all of the examples in this section the optimization parameters used are given in Table 2.2 and are discussed in the remainder of this subsection.

We use a Gaussian RBF, $\phi(r) = e^{-r^2/\xi^2}$, to build the RBF error interpolation and two methods of selecting the spatial correlation length, ξ. The first method is to fix a value of ξ, and the second approach is based on Kriging methods, which assume interpolation errors are normally distributed and maximize the likelihood that the RBF surface predicts the function [69, 93]. To save computation time, the maximum likelihood correlation length is estimated by examining 10 correlation lengths between 0.1 and 5.1, and the correlation length that has the maximum likelihood is chosen. If all correlation lengths have the same likelihood, the maximum correlation length is used. The results in Table 2.1 show that the correlation length has a moderate impact on the convergence rate of the method. For this problem, using either $\xi = 2$ or ξ^*, the correlation length that maximizes the likelihood at each trust-region iteration, leads to the best result.

Table 2.1 demonstrates that the quality of the low-fidelity function can significantly

	Gradient-Free (Calibration)						Gradient-Based	
Low-Fidelity Function	$\xi = 1$	$\xi = 2$	$\xi = 3$	$\xi = 5$	ξ^*	EGO	Add-Corr.	Mult.-Corr.
$f_{\text{low}}(\mathbf{x}) = 0$	148	107	177	223	178	27	503	289†
$f_{\text{low}}(\mathbf{x}) = x_1^2 + x_2^2$	129	77	106	203	76	28	401	312
$f_{\text{low}}(\mathbf{x}) = x_1^4 + x_2^2$	74	74	73	87	65	27	289	171
$f_{\text{low}}(\mathbf{x}) = f_{\text{high}}(\mathbf{x})$	5	5	5	5	7	fail	6	7
$f_{\text{low}}(\mathbf{x}) = -x_1^2 - x_2^2$	195	130	132	250	100	27	fail	352

Table 2.1: Table of average number of function evaluations required to minimize the Rosenbrock function, Eq. 2.19, from a random initial point on $x_1, x_2 \in [-5, 5]$. Results for a selection of Gaussian radial bases function spatial parameters, ξ, are shown. ξ^* corresponds to optimizing the spatial parameter according to a maximum likelihood criteria [69]. Also included are the number of function evaluations required using Efficient Global Optimization (EGO) and first-order consistent trust region methods with a multiplicative correction and an additive correction. The gradient-free calibration uses only additive corrections. For a standard quasi-Newton method the average number of function evaluations is 69 and using DIRECT requires 565 function evaluations. The solutions of all methods have similar accuracy; the errors in the optimal objective function values are all $\mathcal{O}(10^{-6})$. †indicates $f_{\text{low}}(\mathbf{x}) = 1$ had to be used.

impact the number of required high-fidelity function evaluations. As a baseline, the average number of function calls for a quasi-Newton method [76] directly optimizing the Rosenbrock function is 69 and for the global optimization method, DIRECT [51], is 565. The Bayesian calibration approach uses between 5 and 250 high-fidelity function evaluations depending on the quality of the low-fidelity model. The worst case, 250 high-fidelity function evaluations, corresponds to having a poor low-fidelity model, one with opposite trends in most of the design space. The best case, 5 high-fidelity evaluations, corresponds to the case when the low-fidelity function exactly models the high-fidelity function. With a rather good low-fidelity function, for example a 4^{th} degree polynomial, the multifidelity method performs similarly to the quasi-Newton method. Clearly the performance of this method compared to conventional optimization methods depends considerably on the quality of the low-fidelity function used. Results for two first-order consistent multifidelity trust-region methods and multifidelity EGO are presented in Table 2.1 along with the results of the Bayesian calibration method. The multifidelity EGO method creates a Kriging model of the error between the high- and low-fidelity functions and maximizes the expected improvement [52] of the low-fidelity function plus the Kriging model to select additional high-fidelity evaluations and

to update the Kriging model. The Kriging model is initialized using a Latin hypercube sampling of the design space with four function evaluations. The first of the gradient-based multifidelity methods creates a trust-region surrogate by multiplying the low-fidelity function by a correction such that the surrogate matches both the high-fidelity function value and gradient at the current iterate [5]. The second creates the trust-region surrogate by adding a linear function with the error between the high- and low-fidelity function value and gradient at the current iterate to the low-fidelity function. Though an analytical gradient is available for this simple case, when comparing gradient-based multifidelity methods to gradient-free multifidelity methods, we estimate the high-fidelity gradient at each $\mathbf{x}_k$ using a finite difference approximation (requiring n additional high-fidelity function evaluations). This gradient estimate is inexact; however, it is necessary when using a gradient-based optimization method for a function that does not have an available derivative. The general result for this test problem is that the Bayesian calibration approach uses fewer high-fidelity function evaluations than the first-order consistent trust-region approaches of Alexandrov *et al.*[5] but more than multifidelity EGO.

For this simple high-fidelity function, the first-order consistent trust-region methods and the quasi-Newton method require less than half the wall-clock time that the Bayesian calibration method requires (EGO takes roughly twice as long as the Bayesian calibration method). Building the RBF models requires multiple matrix inversions, each of which requires $\mathcal{O}(p_{\max}(p_{\max}+n+1)^3)$ operations, where n is the number of design variables and $p_{\max}$ is the user-set maximum number of calibration points allowed in a model. Accordingly, the Bayesian calibration method is only recommended for high-fidelity functions that are expensive compared to the cost of repeatedly solving for RBF coefficients, which is the case of interest in this chapter.

As with any optimization algorithm, tuning parameters can affect performance significantly; however, the best choices for these tuning parameters can be highly problem dependent. A sensitivity study measured the impact of algorithm parameters on the number of high-fidelity function evaluations for the Rosenbrock example using $f_{\text{low}}(\mathbf{x}) = x_1^2 + x_2^2$ as the low-fidelity function. For all of these tests, one parameter is varied and the remainder are

all set to the values in Table 2.2. The conclusions drawn are based on the average of at least ten runs with random initial conditions on the interval $x_1, x_2 \in [-5, 5]$. While these conclusions may provide general useful guidance for setting algorithm parameters, similar sensitivity studies are recommended for application to other problems.

Parameter	Description	value
$\phi(r)$	RBF Correlation	e^{-r^2/ξ^2}
ξ	RBF spatial correlation length	See Table 2.1
Δ_0	Initial trust region size	$\max[10, \|\mathbf{x}_0\|_\infty]$
$\Delta_{\max}$	Maximum trust region size	$10^3\Delta_0$
ϵ, ϵ_2	Termination tolerances	5×10^{-4}
γ_0	Trust region contraction ratio	0.5
γ_1	Trust region expansion ratio	2
η	Trust region expansion criterion	0.2
α	Trust region contraction ratio used in convergence check	0.9
κ_{FCD}	Fraction of Cauchy decrease requirement	10^{-4}
$p_{\max}$	Maximum number of calibration points	50
θ_1	Minimum projection into nullspace of calibration vectors	10^{-3}
θ_2	RBF coefficient conditioning parameter	10^{-4}
θ_3	Expanded trust-region size to find basis, $\theta_3\Delta_k$	10
θ_4	Maximum calibration region size, $\theta_4\Delta_k$	10
δ_x	Finite difference step size	10^{-6}

Table 2.2: Optimization parameters used in the Rosenbrock function demonstration.

The parameter η is the trust region expansion criterion, where the trust region expands if $\rho_k \geq \eta$ and contracts otherwise. The sensitivity results show that lower values of η have the fewest high-fidelity function calls, and any value $0 \leq \eta \leq 0.2$ performs well. For the trust region expansion ratio, γ_1, the best results are at $\gamma_1 \approx 2$, and high-fidelity function evaluations increase substantially for other values. Similarly, for the contraction ratio, γ_0, the best results are observed at $\gamma_0 \approx 0.5$, with a large increase in high-fidelity function evaluations otherwise. For the fraction of Cauchy decrease, κ_{FCD}, the results show the number of high-fidelity evaluations is fairly insensitive to any value $0 < \kappa_{FCD} < 10^{-2}$. Similarly, for the trust-region contraction ratio used in the algorithm convergence check, α, the number of high-fidelity function evaluations is insensitive to any value $0.5 < \alpha < 0.95$.

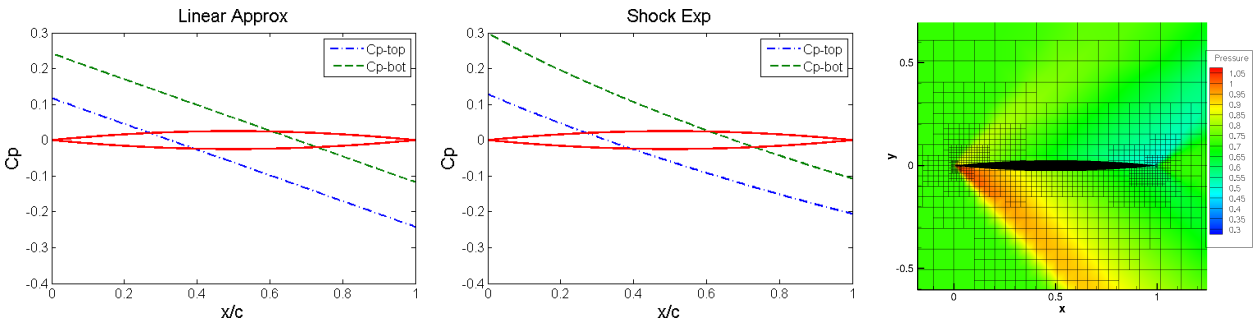
The method of Wild *et al.* to generate fully linear models requires four tuning parameters, θ_1, θ_2, θ_3, and θ_4 [117, 118]. The parameter θ_1 $(0 < \theta_1 < 1)$ determines the acceptable points

when finding the affinely independent basis in the vicinity of the trust region in Algorithm 2.2. As θ_1 increases, the calibration points added to the basis must have a larger projection onto the nullspace of the current basis, and therefore fewer points are admitted to the basis. We find for the Rosenbrock example that the fewest function evaluations occurs with $\theta_1 \approx 10^{-3}$; however, for any value of θ_1 within two orders of magnitude of this value, the number of function evaluations increases by less than 50%. The second parameter, θ_2 ($0 < \theta_2 < 1$), is used in the AddPoints algorithm of Wild *et al.* [118] to ensure that the RBF coefficients remain bounded when adding additional calibration points. The number of allowable calibration points increases as θ_2 decreases to zero; however, the matrix used to compute the RBF coefficients also becomes more ill-conditioned. For our problem, we find that $\theta_2 \approx 10^{-4}$ enables a large number of calibration points while providing acceptable matrix conditioning. The two other parameters, θ_3 and θ_4, used in the calibration point selection algorithm, are significant to the algorithm's performance. The parameter θ_3 ($\theta_3 > 1$) is used if $n+1$ affinely independent previous high-fidelity sample points do not exist within the current trust region. If fewer than $n+1$ points are found, the calibration algorithm allows a search region of increased size $\{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_k\| \leq \theta_3 \Delta_k\}$ in order to find $n+1$ affinely independent points prior to evaluating the high-fidelity function in additional locations. The results show that the number of function calls is insensitive to θ_3 for $1 < \theta_3 \leq 10$, with $\theta_3 \approx 3$ yielding the best results. The parameter θ_4 ($\theta_4 > 1$) represents the balance between global and local model calibration, as it determines how far points can be from the current iterate, $\mathbf{x}_k$, and still be included in the RBF interpolation. Points that lie within a region $\{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_k\| \leq \theta_4 \Delta_k\}$ are all candidates to be added to the interpolation. Calibration points outside of the trust region will affect the shape of the model within the trust region, but the solution to the subproblem must lie within the current trust region. The results of our analysis show that $\theta_4 \approx 10$ offers the best performance, with the number of high-fidelity function calls increasing substantially if $\theta_4 < 5$ or $\theta_4 > 15$. We also note that the parameter values suggested in this section are similar to the values recommended by Wild *et al.* [117].

2.4.2 Supersonic Airfoil Optimization

As an engineering example, a supersonic airfoil is optimized for minimum drag at Mach 1.5. Three analysis tools are available: a supersonic linear panel method, a shock-expansion theory panel method, and a computational fluid dynamics model Cart3D [1]. The linear panel method and shock-expansion theory method only compute flow quantities on the surface of the airfoil. The panel method assumes that all changes in geometry are small, so that the local surface pressure varies linearly with the slope changes in the airfoil surface. Shock-expansion theory computes the change in the local surface pressure by solving nonlinear equations associated with compression and expansion waves related to the surface slope changes. Cart3D uses a finite-volume method to approximate the Euler equations and solve for the flow field in a domain around the airfoil. Cart3D uses an adjoint-based mesh refinement approach. The refinement is managed so that the error in the flow solution due to spatial approximation on the computational mesh is less than a set tolerance. (The smallest tolerance we were able to consistently achieve was about $\mathcal{O}(10^{-5})$ for Drag.)

Figure 2-3 shows the approximate level of detail used in the models, and Table 2.3 compares the lift and drag estimates from each of the models for a 5% thick biconvex airfoil at Mach 1.5 and 2° angle of attack. For this particular test case shock-expansion theory is exact, but in general Cart3D is more broadly applicable (including mixed subsonic/supersonic flows and 3D problems) so it is treated as the highest fidelity. The linear panel method and shock-expansion theory both require sharp leading and trailing edges on the airfoil, so the airfoils



(a) Panel method. (b) Shock-expansion method. (c) Cart3D.

Figure 2-3: Supersonic airfoil model comparisons at Mach 1.5 and 2° angle of attack.

are parameterized by a set of spline points on the upper and lower surfaces and the angle of attack. The leading and trailing edge points of both surfaces are constrained to be coincident to maintain the sharp leading and trailing edges. Accordingly, an airfoil with eleven variables is parameterized by the angle of attack, and has seven spline points on the upper and lower surfaces, but only five points on each surface can be varied.

To demonstrate the RBF calibration approach to optimization, the linear supersonic panel method is used as the low-fidelity function and shock-expansion theory is used as the high-fidelity function. For supersonic flow, a zero thickness airfoil will have the minimum drag, so the airfoil must be constrained to have a thickness to chord ratio greater than 5%. This is accomplished by adding a penalty function, so that if the maximum thickness of the airfoil is less than 5%, the penalty term $1000(t/c - 0.05)^2$ is added to the drag. A similar penalty is added if the thickness anywhere on the airfoil is less than zero.

	Panel	Shock-Expansion	Cart3D
Time	0.001s	0.5s	3-5m
C_L	0.1244	0.1278	0.1250
C_D	0.0164	0.0167	0.01666

Table 2.3: 5% thick biconvex airfoil results comparison at Mach 1.5 and 2° angle of attack. These results are typical for "well-designed" airfoils, however, in other parts of the feasible design space much larger discrepancies are observed.

The optimization parameters used by this method are the same as in Table 2.2, with the exception that the RBF correlation length is either $\xi = 2$ or optimized at each iteration. A consecutive step size of less than 5×10^{-6} is an additional termination criteria for all of the multifidelity methods compared. The number of high-fidelity function evaluations required to optimize the airfoil for each of the methods using a different number of design variables is presented in Figure 2-4. The airfoil optimization shows that both the first-order consistent methods and the RBF calibration method perform significantly better than the quasi-Newton method. This is largely because the multifidelity methods have a significant advantage over the single-fidelity methods in that the physics-based low-fidelity model is a reasonable representation of the high-fidelity model. However, the RBF calibration approach uses less

than half the number of function evaluations as the multiplicative-correction approach. In addition, the additive correction outperforms the multiplicative correction for this problem, but the RBF calibration outperformed both. The method of maximizing the likelihood of the RBF calibration performs slightly better than just using a fixed correlation length. The results in Figure 2-4 are also better than two global optimization methods, DIRECT [51] and a multifidelity formulation of EGO [52]. DIRECT requires significantly more high-fidelity evaluations than the quasi-Newton method and all of the trust-region approaches. For example to get within 1% of the optimal function value with 11 design variables, DIRECT requires over 13,000 evaluations. We are unable to get the multifidelity formulation of EGO to find an airfoil design having a drag coefficient within 20% of the optimal value for a variety of low-fidelity functions and a budget of 1,500 high-fidelity evaluations. Many EGO implementations have been attempted, including initializing the Kriging model with a Latin hypercube sample using $3n$, $4n$, and $5n$ high-fidelity evaluations and maximizing the expected improvement using DIRECT and a genetic algorithm.

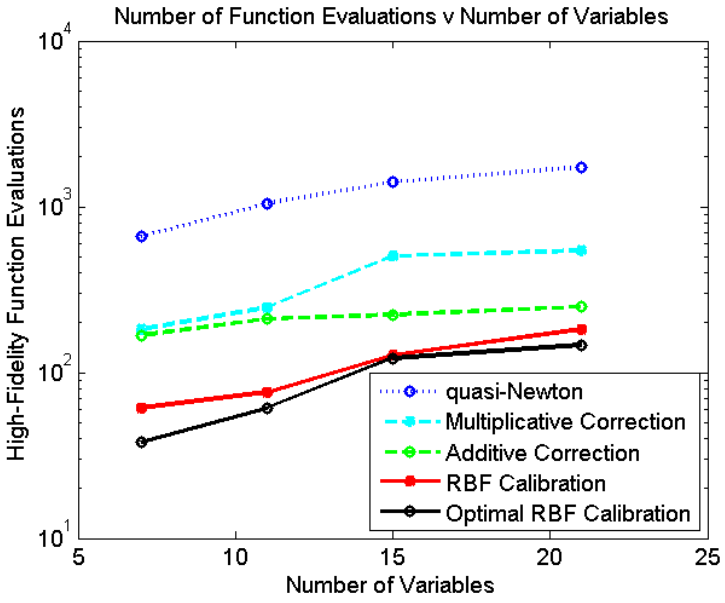


Figure 2-4: Number of shock-expansion theory evaluations required to minimize the drag of a supersonic airfoil verse the number of parameters. The low-fidelity model is the supersonic panel method. The errors in the optimal objective function values are $\mathcal{O}(10^{-6})$ for all methods.

As a second test case, the panel method was used as a low-fidelity function to minimize the drag of an airfoil with Cart3D as the high-fidelity function. Cart3D has an adjoint-based mesh refinement, which ensures the error caused by the discretization is less than a tolerance. Accordingly, the drag computed by Cart3D is only to within a tolerance. The drag is therefore not Lipschitz continuous due to the finite precision. In the execution of our multifidelity optimization algorithm, the gradient of the surrogate model does not go to zero. However, no progress is made and the trust region radius converges to zero. This forces the algorithm to take small steps and the combination of a small step size and small trust region is a supplemental termination criteria. On average, the airfoil parameterized with 11 variables requires 88 high-fidelity (Cart3D) function evaluations. A comparison of the minimum drag airfoils from the panel method, shock-expansion theory, and Cart3D is presented in Figure 2-5. The airfoils all resemble the expected diamond shape.

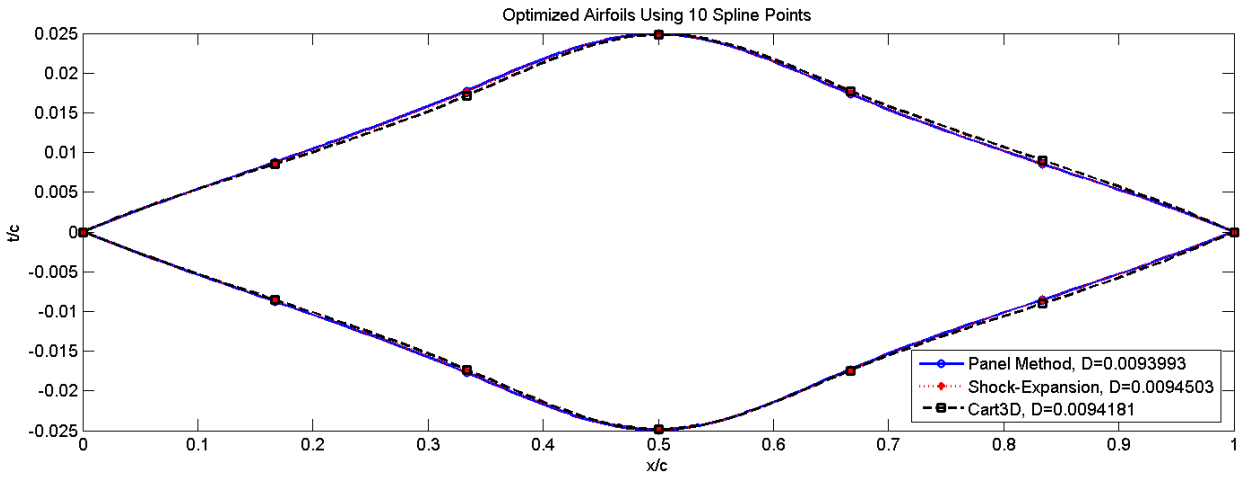


Figure 2-5: Minimum drag airfoils from each of the three analysis models. The panel method airfoil is generated by solving a single-fidelity optimization problem with a quasi-Newton method. The shock-expansion method and Cart3D airfoils are generated with this multifidelity approach and spatial correlation length $\xi = 2$ with the panel method as a low-fidelity model.

To compare this gradient-free approach on a problem with multiple local minima, we change the objective to find an airfoil with a lift coefficient of 0.3 and a drag coefficient of less than 0.05, minimizing $(C_L - 0.3)^2 + \max\{C_D - 0.05, 0\}^2$, and retain the same penalty function to ensure positive thickness and at least 5% thickness to chord ratio. There are

numerous airfoil designs in the design space that have the optimal function value, zero (approximately, due to the penalty function). Considering shock-expansion theory as the high-fidelity analysis and the panel method as the low-fidelity analysis, Table 2.4 compares the average number of high-fidelity evaluations from random initial airfoils parametrized by the angle of attack and ten surface spline points. Two low-fidelity functions are used, $(C_L - 0.3)^2 + \max\{C_D - 0.05, 0\}^2$, which is the same as the objective function except computed with the panel method instead of shock-expansion theory, and $(C_L - 0.4)^2 + \max\{C_D - 0.05, 0\}^2$ also computed by the panel method. Both of these low-fidelity objectives are augmented with the penalty function. The first-low fidelity objective has minima near the high-fidelity minima; however, the second low-fidelity objective has multiple local minima that are all separate from the high-fidelity minima. For comparison, from random initial airfoil designs a quasi-Newton method [76] requires 643 shock-expansion evaluations, and the global optimization method, DIRECT, requires over 1,000 evaluations to find an objective value of 1×10^{-5}, and over 8,000 evaluations for 1×10^{-8}. The results show that on average with a "good" low-fidelity model the RBF calibration approach uses the fewest high-fidelity evaluations to find a locally optimal solution. For this problem, with a "poor" low-fidelity model the RBF calibration approach finished second to the first-order consistent trust-region approach using an additive correction. We also observe that the RBF calibration approach is not significantly affected by either choosing a correlation length, $\xi = 2$, or by using the maximum likelihood correlation length, $\xi = \xi^*$. For the EGO implementation, $4n$ high-fidelity evaluations are used for the initial Latin hypercube sample and a genetic algorithm is used to maximize the expected improvement.

2.5 Combining Multiple Fidelity Levels

This section addresses how the radial basis function interpolation technique can be extended to optimize a function when there are multiple lower-fidelity functions. For instance, consider the case when our goal is to find the $\mathbf{x}^*$ that minimizes $f_{\text{high}}(\mathbf{x})$, and there exists two or more lower-fidelity functions, an intermediate-fidelity, $f_{\text{med}}(\mathbf{x})$, and a low-fidelity, $f_{\text{low}}(\mathbf{x})$.

	Gradient-Free (Calibration)			Gradient-Based	
Low-Fidelity Function	$\xi = 2$	$\xi = \xi^*$	EGO	Add-Corr.	Mult.-Corr.
$(C_L - 0.3)^2 + \max\{C_D - 0.05, 0\}^2$	78	54	118	80	$123^\dagger$
$(C_L - 0.4)^2 + \max\{C_D - 0.05, 0\}^2$	152	147	285	83	$642^\dagger$

Table 2.4: Average number of shock-expansion theory evaluations required to find an airfoil (angle of attack and ten surface spline points, 11 design parameters) with a target lift coefficient of 0.3 and maximum drag coefficient of 0.05. The low-fidelity analysis is the panel method, and two low-fidelity objective functions are used to demonstrate the influence of low-fidelity model quality. For comparison, a quasi-Newton method requires 643 evaluations and the global optimization method, DIRECT, requires 1,031 evaluations to get within 1×10^{-5} of the optimal objective, and 8,215 evaluations to get within 1×10^{-8} of the optimal objective. The errors in the optimal objective function values are $\mathcal{O}(10^{-8})$ for all methods except DIRECT. †indicates 1×10^{-4} was added to the low-fidelity objective function.

The typical approach to solve this problem is to nest the lower-fidelity function; that is, to use the intermediate-fidelity function as the low-fidelity model of the high-fidelity function, and to use the lowest-fidelity function as the low-fidelity model of the intermediate-fidelity function. To do this, two calibration models are needed,

$$f_{\text{high}}(\mathbf{x}) \approx f_{\text{med}}(\mathbf{x}) + e_{\text{med}}(\mathbf{x}) \tag{2.20}$$

$$f_{\text{med}}(\mathbf{x}) \approx f_{\text{low}}(\mathbf{x}) + e_{\text{low}}(\mathbf{x}). \tag{2.21}$$

In the nested approach, the high-fidelity optimization is performed on the approximate high-fidelity function, which is the medium-fidelity function plus the calibration model e_{med}. However, to determine the steps in that optimization, another optimization is performed on a lower-fidelity model. This low-fidelity optimization is performed on the model

$$m(\mathbf{x}) \approx f_{\text{low}}(\mathbf{x}) + e_{\text{low}}(\mathbf{x}) + e_{\text{med}}(\mathbf{x}), \tag{2.22}$$

but only the low-fidelity calibration model e_{low} is adjusted. The nested approach can quickly become computationally inefficient. In order to take one step in the high-fidelity space, an optimization is required on the medium-fidelity function. However, for each step in medium-fidelity space, an optimization is required on the lower-fidelity function. Even if the medium and low-fidelity models are cheap to evaluate compared to the high-fidelity model,

the potential exponential scaling in the number of lower-fidelity function evaluations required between fidelity levels can counter computational gains.

An alternative to nesting multiple lower-fidelity functions is to use a maximum likelihood estimator to estimate the high-fidelity function. Since the multifidelity optimization method proposed in this chapter uses radial basis function interpolants, a variance estimate of the interpolation error can be created using standard Gaussian process techniques [69, 93]. This variance estimate reflects the overall level of variability in the observed data (roughly the average error between models) and the observed correlations (how smoothly the error changes in the design space). It is therefore zero at all sampled points and grows with distance away from sampled points [52]. Figure 2-2 presents an illustration of this. In the case of multiple fidelity levels, the lower fidelity estimates of $f_{\text{high}}(\mathbf{x})$, for example $f_{\text{low}}(\mathbf{x})+e_k(\mathbf{x})$, are assumed to have uncertainty that is normally distributed with zero mean and variance $\sigma_k^2(\mathbf{x})$, denoted $\mathcal{N}\left(0, \sigma_k^2(\mathbf{x})\right)$. In the two-fidelity optimization, only the value of the surrogate model is used, but in the following, both the value of the surrogate model and the uncertainty estimate are used.

For two lower-fidelity models, the estimates of the high-fidelity function are

$$f_{\text{high}}(\mathbf{x}) \approx f_{\text{med}}(\mathbf{x}) + e_{\text{med},k}(\mathbf{x}) + \mathcal{N}\left(0, \sigma^2_{\text{med},k}(\mathbf{x})\right) \tag{2.23}$$

$$f_{\text{high}}(\mathbf{x}) \approx f_{\text{low}}(\mathbf{x}) + e_{\text{low},k}(\mathbf{x}) + \mathcal{N}\left(0, \sigma^2_{\text{low},k}(\mathbf{x})\right). \tag{2.24}$$

From these two or more models, a maximum likelihood estimate of the high-fidelity function weights each prediction according to a function of the variance estimates. The high-fidelity maximum likelihood estimate has a mean $f_{\text{est},k}$, given by

$$\begin{aligned} f_{\text{est},k}(\mathbf{x}) = \left(f_{\text{med}}(\mathbf{x}) + e_{\text{med},k}(\mathbf{x})\right) &\left[\frac{\sigma^2_{\text{low},k}(\mathbf{x})}{\sigma^2_{\text{low},k}(\mathbf{x}) + \sigma^2_{\text{med},k}(\mathbf{x})}\right] \\ + \left(f_{\text{low}}(\mathbf{x}) + e_{\text{low},k}(\mathbf{x})\right) &\left[\frac{\sigma^2_{\text{med},k}(\mathbf{x})}{\sigma^2_{\text{low},k}(\mathbf{x}) + \sigma^2_{\text{med},k}(\mathbf{x})}\right]. \end{aligned} \tag{2.25}$$

The estimate of the high-fidelity function also has a variance, $\sigma^2_{\text{est},k}$, which is less than either

of the variances of the lower-fidelity models since

$$\frac{1}{\sigma^2_{\text{est},k}(\mathbf{x})} = \frac{1}{\sigma^2_{\text{low},k}(\mathbf{x})} + \frac{1}{\sigma^2_{\text{med},k}(\mathbf{x})}. \tag{2.26}$$

A thorough discussion of using a maximum likelihood estimator to combine two or more estimates with normally distributed uncertainties is available in [77, Chapter 1]. We note that naming the two lower-fidelity estimates $f_{\text{low}}(\mathbf{x})$ and $f_{\text{med}}(\mathbf{x})$ may be misleading, since the maximum likelihood estimator makes no hierarchical distinction between models. In fact, our approach applies naturally to the case where different models have varying relative levels of fidelity over different regions of the design space.

We present a schematic of the behavior of this maximum likelihood estimator in Figure 2-6. In the first case with two similar models, the combined estimate has a similar mean with a reduced variance. In the second case with two dissimilar estimates, the combined estimate has the average mean of the two models again with lower variance. In the third case when one model has a considerably smaller variance than the other model, the combined estimate has a similar mean and slightly reduced variance than the model with the lower variance. Accordingly, the maximum likelihood estimate is the best probabilistic guess of the high-fidelity function at a non-calibrated point.

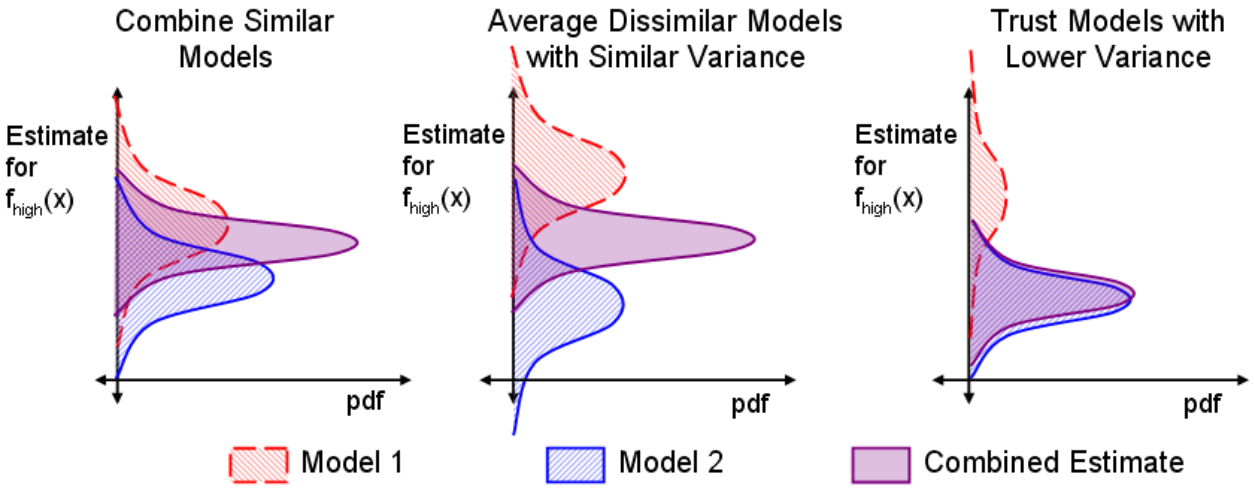


Figure 2-6: Behavior of the combined maximum likelihood estimate given the behavior of the individual estimates.

This method provides flexibility while still being provably convergent to a high-fidelity optimum using our multifidelity optimization approach. The requirements for convergence are that the surrogate model upon which the optimization is performed be smooth and exactly interpolate the function at the necessary calibration points. Using this maximum likelihood estimator, only one of the lower-fidelity functions needs to be sampled at the calibration points because at a calibration point an individual Gaussian process model has zero variance. Accordingly, at that calibration point the model is known to be correct and the other lower-fidelity information is not used. Therefore, the user has flexibility in selecting which of the lower-fidelity models are calibrated at a calibration point. For example, the calibration procedure could choose a ratio, such as one intermediate-fidelity update for every three low-fidelity updates, or simply update both the intermediate-fidelity and low-fidelity models each time a new calibration point is needed.

The changes to Algorithm 2.1 necessary to use multiple low-fidelity models are minimal. In steps 1, 3, and 7, $f_{\text{est},k}(\mathbf{x})$ should be used in lieu of $m_k(\mathbf{x})$, and in step 6 at least one of the error models, $e_{\text{med},k}(\mathbf{x})$ or $e_{\text{low},k}(\mathbf{x})$, needs to be updated using Algorithm 2.2 to make $f_{\text{est},k+1}(\mathbf{x})$ fully linear on the updated trust region.

Optimization results show that the nesting approach suffers from poor scaling between fidelity levels and that the maximum likelihood approach speeds convergence of our multifidelity optimization method even if the lowest-fidelity function is a poor representation of the high-fidelity function. In all examples presented, the calibration strategy employed for the maximum likelihood method is to update all lower-fidelity models whenever the optimization method requires a new calibration point.

The first example is an optimization of the Rosenbrock function with two parabolic lower-fidelity functions. The number of required function evaluations for each fidelity level is presented in Table 2.5. Using the maximum likelihood approach, the number of high-fidelity function evaluations has been reduced by 34%, and the number of combined lower-fidelity evaluations has been reduced by 27%. However, combining the multiple lower-fidelity functions through nesting leads to a large increase in the number of function evaluations at each level.

Method	$(x_2 - x_1^2)^2 + (1 - x_1)^2$	$(x_1 - 1)^2 + x_2^2$	$x_1^2 + x_2^2$
Two-Fidelities	87	0	6975
Max. Likelihood	57	2533	2533
Nested	137	4880	50455

Table 2.5: Number of function calls required to optimize the Rosenbrock function using multiple lower-fidelity functions. The maximum likelihood approach requires the least high-fidelity function evaluations to converge and the nested approach the most.

The second example is to optimize a supersonic airfoil for minimum drag with respect to an Euler code, Cart3D. Two lower-fidelity methods are used: shock-expansion theory and a panel method. These results, presented in Table 2.6, also show that the maximum likelihood approach converges faster and with fewer calibration points than the original multifidelity method using only the panel method. The nesting approach failed to converge as the step size required in the intermediate-fidelity optimization became too small. The likely cause of this is that the adjoint-based mesh refinement used in Cart3D allows numerical oscillations in the output functional at a level that is still significant in the optimization, and this makes the necessary calibration surface non-smooth. The lack of smoothness violates the convergence criteria of this method.

Method	Cart3D	Shock-expansion	Panel Method
Two-Fidelities	88	0	47679
Max. Likelihood	66	23297	23297
Nested	66*	7920*	167644*

Table 2.6: Number of function calls required to optimize an airfoil for minimum drag using the Euler equations (Cart3D) with multiple lower-fidelity models. An asterisk indicates that solution was not converged due to numerical limitations.

The final example demonstrates that the maximum likelihood approach can still benefit from a poor low-fidelity model. The results in Table 2.7 are for minimizing the drag of a supersonic airfoil using shock-expansion theory, with the panel method as an intermediate-fidelity function; however, unlike the preceding example, the lowest-fidelity model is quite poor and uses the panel method only on the camberline of the airfoil. Using this method, any symmetric airfoil at zero angle of attack has no drag and many of the predicted trends are incorrect compared to the panel method or shock-expansion theory. The optimization results

show an important benefit of this maximum likelihood approach: even adding this additional bad information, the number of high-fidelity function calls has been reduced by 33%, and the number of intermediate-fidelity function calls has decreased by 31%. An additional point of note is the magnitude to which the nested approach suffers by adding poor low-fidelity information. In most test problems, the nested optimization was terminated due to an exceptionally large number of function evaluations. The results presented are the minimum number of function evaluations the nested approach required to converge.

Method	Shock-expansion	Panel Method	Camberline
Two-Fidelities	126	43665	0
Max. Likelihood	84	30057	30057
Nested	212*	59217*	342916*

Table 2.7: Number of function calls required to optimize an airfoil for minimum drag using shock-expansion theory with multiple lower-fidelity models. An asterisk indicates a minimum number of function evaluations as opposed to an average value from random starting points.

From this observed behavior, we recommend the use of the two-fidelity optimization approach demonstrated in Section 2.4 for the optimization of any computationally expensive function for which accurate gradient information is not available. When considering the addition of a third or fourth fidelity level, we note there are diminishing returns for each additional fidelity level. In the three-fidelity examples presented, the third fidelity level reduced the number of high-fidelity evaluations in the two-fidelity case by about 30%, which is about an 8% additional reduction over a single-fidelity method. We therefore expect the greatest multifidelity benefit when going from a single fidelity level to two fidelity levels, and diminishing benefit for each additional level. However, if high-fidelity function evaluations are extremely costly (e.g., requiring hours or days on a supercomputer), then the additional reduction may certainly be worthwhile. Since the reduction in high-fidelity evaluations using gradient-free methods is similar to what has been observed using gradient-based multifidelity optimization methods [5], we expect similar returns for additional fidelity levels in the gradient-based case.

2.6 Summary

This chapter has presented a provably convergent multifidelity optimization method that does not require computation of derivatives of the high-fidelity function. The optimization results show that this method reduces the number of high-fidelity function calls required to find a local minimum compared with other state-of-the-art methods. The method creates surrogate models that retain accurate local behavior while also capturing some global behavior of the high-fidelity function. However, some downfalls of the method are that its performance is sensitive to the quality of the low-fidelity model and that its overhead increases dramatically with the number of design variables and the number of calibration points used to build the radial basis function model. Accordingly, this approach is only recommended for high-fidelity functions that require a considerable wall-clock time. Furthermore, in this unconstrained optimization algorithm engineering design constraints are handled with a penalty method, but using surrogate models for the constraints and objective may be desirable. While trust-region methods exist for constrained optimization problems, incorporating multifidelity models in both the objective function and the constraints presents several challenges which will be discussed in Chapters 3 and 5.

This chapter has also shown that a multifidelity optimization method based on a maximum likelihood estimator is an effective way of combining many fidelity levels to optimize a high-fidelity function. The maximum likelihood estimator permits flexible sampling strategies among the low-fidelity models and is robust with respect to poor low-fidelity estimates. In addition, the estimator offers a natural and automated way of selecting among different models that are known to be accurate in different parts of the design space, which is frequently the case in engineering design.

Chapter 3

Constrained Gradient-Free Multifidelity Optimization

This chapter extends the Bayesian model calibration technique and unconstrained multifidelity optimization method presented in Chapter 2 to constrained optimization problems. The algorithm minimizes a high-fidelity objective function subject to a high-fidelity constraint and other simple constraints. The algorithm never computes the gradient of a high-fidelity function; however, it achieves first-order optimality using sensitivity information from the calibrated low-fidelity models, which are constructed to have negligible error in a neighborhood around the solution. The method is demonstrated for aerodynamic shape optimization and is compared with other single-fidelity derivative-free and sequential quadratic programming methods.

Section 3.1 of this chapter presents the derivative-free method to optimize a high-fidelity objective function subject to constraints with available derivatives. Fully linear surrogate models of the objective function are minimized within a trust-region setting until no further progress is possible or when convergence to a high-fidelity optimum is achieved. Section 3.2 presents a technique for minimizing a high-fidelity objective function subject to both constraints with available derivatives and computationally expensive constraints with unavailable derivatives. The constraints without available derivatives are approximated with multifidelity methods, whereas the other constraints are handled either implicitly with a

penalty method or explicitly. Section 3.3 presents an aerodynamic shape optimization problem to demonstrate the proposed multifidelity optimization techniques and compares the results with other single-fidelity methods and approximation model management using finite difference gradient estimates. Finally, Section 3.4 summarizes the chapter and discusses extensions of the method to the case when constraints are hard (when the objective function fails to exist if the constraints are violated).

3.1 Constrained Optimization of a Multifidelity Objective Function

This section considers the constrained optimization of a high-fidelity function, $f_{\text{high}}(\mathbf{x})$, that accurately estimates system metrics of interest but for which accurate gradient estimates are unavailable. We first present a formalized problem statement and some qualifying assumptions. We then present a trust-region framework, the surrogate-based optimization problems performed within the trust region, and the trust region updating scheme. We follow this with an algorithmic implementation of the method, and with a brief discussion of algorithmic limitations and theoretical considerations needed for robustness.

3.1.1 Problem Setup and Assumptions

We seek the vector $\mathbf{x} \in \mathbb{R}^n$ of n design variables that minimizes the value of the high-fidelity objective function subject to equality constraints, $\mathbf{h}(\mathbf{x})$, and inequality constraints $\mathbf{g}(\mathbf{x})$,

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & f_{\text{high}}(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}) = 0 \\ & \mathbf{g}(\mathbf{x}) \leq 0, \end{aligned} \tag{3.1}$$

where we assume gradients of $\mathbf{h}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$ with respect to $\mathbf{x}$ are available or can be estimated accurately. To reduce the number of evaluations of $f_{\text{high}}(\mathbf{x})$ we use a low-fidelity

function, $f_{\text{low}}(\mathbf{x})$, that estimates the same metric as $f_{\text{high}}(\mathbf{x})$ but with cheaper evaluation cost and lower accuracy. We seek to find the solution to (3.1) without estimating gradients of $f_{\text{high}}(\mathbf{x})$, by calibrating $f_{\text{low}}(\mathbf{x})$ to $f_{\text{high}}(\mathbf{x})$ and using sensitivity information from the calibrated surrogate model. The calibration strategy employed may break down should either $f_{\text{high}}(\mathbf{x})$ or $f_{\text{low}}(\mathbf{x})$ not be twice continuously differentiable or not have a Lipschitz continuous first derivative, although in many such cases the algorithm may still perform well.

3.1.2 Trust-region Model Management

From an initial design vector $\mathbf{x}_0$, the trust-region method generates a sequence of design vectors that each reduce a merit function consisting of the high-fidelity function value and penalized constraint violation, where we denote $\mathbf{x}_k$ to be this design vector on the kth trust-region iteration. We follow the general Bayesian calibration approach in [55], and construct a surrogate model $m_k(\mathbf{x})$ for $f_{\text{high}}(\mathbf{x})$, defined in (2.2), within the trust region, $\mathcal{B}_k$, defined in (2.3).

To solve the constrained optimization problem presented in (3.1) we define a merit function, $\Upsilon(\mathbf{x}_k, \mu_k)$, where μ_k is a parameter that must go to infinity as the iteration number k goes to infinity and serves to increase the penalty placed on the constraint violation. To prevent divergence of this algorithm, we need the penalty function to satisfy some basic properties. First, the merit function with the initial penalty, μ_0, must be bounded from below within a relaxed level-set, $\mathcal{L}(\mathbf{x}_0, \mu_0)$, defined as

$$L(\mathbf{x}_0, \mu_0) = \{\mathbf{x} \in \mathbb{R}^n : \Upsilon(\mathbf{x}, \mu_0) \leq \Upsilon(\mathbf{x}_0, \mu_0)\} \tag{3.2}$$

$$B(\mathbf{x}_k) = \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x} - \mathbf{x}_k\| \leq \Delta_{\max}\} \tag{3.3}$$

$$\mathcal{L}(\mathbf{x}_0, \mu_0) = L(\mathbf{x}_0, \mu_0) \bigcup_{\mathbf{x_k} \in L(\mathbf{x}_0, \mu_0)} B(\mathbf{x}_k), \tag{3.4}$$

where $\Delta_{\max}$ is the maximum allowable trust-region size and the relaxed level-set is required because the trust-region algorithm may attempt to evaluate the high-fidelity function at points outside of the level set at $\mathbf{x}_0$. Second, the level sets of $\Upsilon(\mathbf{x}_k, \mu_k > \mu_0)$ must be

contained within $L(\mathbf{x}_0, \mu_0)$, and third, $L(\mathbf{x}_0, \mu_0)$ must be a compact set. These properties ensure that all design iterates, $\mathbf{x}_k$, remain within $L(\mathbf{x}_0, \mu_0)$.

Although other merit functions, such as augmented Lagrangians, are possible, we restrict our attention to merit functions based on quadratic penalty functions because it is trivial to show that they are bounded from below if the objective function obtains a finite global minimum and there is no need to consider arbitrarily bad Lagrange multiplier estimates. The merit function used in this method is the objective function plus the scaled sum-squares of the constraint violation, where $\mathbf{g}^+(\mathbf{x})$ denotes the values of the nonnegative inequality constraints,

$$\Upsilon(\mathbf{x}, \mu_k) = f_{\text{high}}(\mathbf{x}) + \frac{\mu_k}{2}\mathbf{h}(\mathbf{x})^\top \mathbf{h}(\mathbf{x}) + \frac{\mu_k}{2}\mathbf{g}^+(\mathbf{x})^\top \mathbf{g}^+(\mathbf{x}). \tag{3.5}$$

The parameter μ_k is a penalty weight, which must go to $+\infty$ as the iteration k goes to $+\infty$. Note that when using a quadratic penalty function for constrained optimization, under suitable hypotheses on the optimization algorithm and penalty function, the sequence of iterates generated, $\{\mathbf{x}_k\}$, can either terminate at a feasible regular point at which the Karush-Kuhn-Tucker (KKT) conditions are satisfied, or at a point that minimizes the squared norm of the constraint violation, $\mathbf{h}(\mathbf{x})^\top \mathbf{h}(\mathbf{x}) + \mathbf{g}^+(\mathbf{x})^\top \mathbf{g}^+(\mathbf{x})$ [13, 83].

We now define a surrogate merit function, $\hat{\Upsilon}(\mathbf{x}, \mu_k)$, which replaces $f_{\text{high}}(\mathbf{x})$ with its surrogate model $m_k(\mathbf{x})$,

$$\hat{\Upsilon}(\mathbf{x}, \mu_k) = m_k(\mathbf{x}) + \frac{\mu_k}{2}\mathbf{h}(\mathbf{x})^\top \mathbf{h}(\mathbf{x}) + \frac{\mu_k}{2}\mathbf{g}^+(\mathbf{x})^\top \mathbf{g}^+(\mathbf{x}). \tag{3.6}$$

Optimization is performed on this function, and updates to the trust-region are based on how changes in this surrogate merit function compare with changes in the original merit function, $\Upsilon(\mathbf{x}, \mu_k)$.

Our calibration strategy is to make the surrogate models $m_k(\mathbf{x})$ *fully linear*, Definition 2.1 but with the level-set $L(\mathbf{x}_0, \mu_0)$ as opposed to $L(\mathbf{x}_0)$.

3.1.3 Trust-region Subproblem

At each trust-region iteration a point likely to decrease the merit function is found by solving one of two minimization problems on the fully linear model for a step $\mathbf{s}_k$, on a trust region of size Δ_k:

$$\begin{aligned} \min_{\mathbf{s}_k \in \mathbb{R}^n} \quad & m_k(\mathbf{x}_k + \mathbf{s}_k) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_k + \mathbf{s}_k) = 0 \\ & \mathbf{g}(\mathbf{x}_k + \mathbf{s}_k) \leq 0 \\ & \|\mathbf{s}_k\| \leq \Delta_k, \end{aligned} \tag{3.7}$$

or

$$\begin{aligned} \min_{\mathbf{s}_k \in \mathbb{R}^n} \quad & \hat{\Upsilon}_k(\mathbf{x}_k + \mathbf{s}_k, \mu_k) \\ \text{s.t.} \quad & \|\mathbf{s}_k\| \leq \Delta_k. \end{aligned} \tag{3.8}$$

The subproblem in (3.8) is used initially to reduce constraint infeasibility. However, there is a limitation with this subproblem that the norm of the objective function Hessian grows without bound due to the penalty parameter increasing to infinity. Therefore to both speed convergence and prevent Hessian conditioning issues, the subproblem in (3.7) with explicit constraint handling is used as soon as a point that satisfies $\mathbf{h}(\mathbf{x}) = 0$ and $\mathbf{g}(\mathbf{x}) \leq 0$ exists within the current trust region. This is estimated by a linear approximation to the constraints, however, if the linearized estimate falsely suggests (3.7) has a feasible solution then we take recourse to (3.8).[1]

For both trust-region subproblems, (3.7) and (3.8), the subproblem must be solved such that the 2-norm of the first-order optimality conditions is less than a constant τ_k. This requirement is stated as $\|\nabla_x \mathfrak{L}_k\| \leq \tau_k$, where $\mathfrak{L}_k$ is the Lagrangian for the trust-region

[1]Note that an initial feasible point, $\mathbf{x}_0$ could be found directly using the gradients of $\mathbf{h}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$; however, since the penalty method includes the descent direction of the objective function it may better guide the optimization process in the case of multiple feasible regions. Should the initial iterate be feasible, the deficiencies of a quadratic penalty function are not an issue.

subproblem used. There are two requirements for τ_k. First $\tau_k < \epsilon$, where ϵ is the desired termination tolerance for the optimization problem in (3.1). Second, τ_k must decrease to zero as the number of iterations goes to infinity. Accordingly, we define

$$\tau_k = \min\left[\beta\epsilon, \alpha\Delta_k\right], \tag{3.9}$$

with a constant $\beta \in (0,1)$ to satisfy the overall tolerance criteria, and a constant $\alpha \in (a,1)$ multiplying Δ_k to ensure that τ_k goes to zero. The constant a will be defined as part of a sufficient decrease condition that forces the size of the trust-region to decrease to zero in the next subsection.

3.1.4 Trust-region Updating

Without using the high-fidelity function gradient, the trust-region update scheme must ensure the size of the trust-region decreases to zero to establish convergence. To do this, a requirement similar to the fraction of Cauchy decrease requirement in an the unconstrained trust-region formulation is used (see, for example, [27]). We require that the improvement in our merit function is at least a small constant $a \in (0, \epsilon]$, multiplying Δ_k,

$$\hat{\Upsilon}(\mathbf{x}_k, \mu_k) - \hat{\Upsilon}(\mathbf{x}_k + \mathbf{s}_k, \mu_k) \geq a\Delta_k. \tag{3.10}$$

The sufficient decrease condition is enforced through the trust region update parameter, ρ_k. The update parameter is the ratio of the actual reduction in the merit function to the predicted reduction in the merit function unless the sufficient decrease condition is not met,

$$\rho_k = \begin{cases} 0 & \hat{\Upsilon}(\mathbf{x}_k, \mu_k) - \hat{\Upsilon}(\mathbf{x}_k + \mathbf{s}_k, \mu_k) < a\Delta_k \\ \frac{\Upsilon(\mathbf{x}_k, \mu_k) - \Upsilon(\mathbf{x}_k + \mathbf{s}_k, \mu_k)}{\hat{\Upsilon}(\mathbf{x}_k, \mu_k) - \hat{\Upsilon}(\mathbf{x}_k + \mathbf{s}_k, \mu_k)} & \text{otherwise.} \end{cases} \tag{3.11}$$

The size of the trust region, Δ_k, must now be updated based on the quality of the surrogate model prediction. The size of the trust region is increased if the surrogate model predicts the change in the function value well, kept constant if the prediction is fair, and the trust

region is contracted if the model predicts the change poorly. Specifically, we update the trust region size using

$$\Delta_{k+1} = \begin{cases} \min\{\gamma_1 \Delta_k, \Delta_{\max}\} & \text{if } \eta_1 \leq \rho_k \leq \eta_2, \\ \gamma_0 \Delta_k & \text{if } \rho_k \leq \eta_0, \\ \Delta_k & \text{otherwise,} \end{cases} \tag{3.12}$$

where $0 < \eta_0 < \eta_1 < 1 < \eta_2$, $0 < \gamma_0 < 1$, and $\gamma_1 > 1$. Regardless of whether or not a sufficient decrease has been found, the trust-region center will be updated if the trial point has decreased the value of the merit function,

$$\mathbf{x}_{k+1} = \begin{cases} \mathbf{x}_k + \mathbf{s}_k & \text{if } \Upsilon(\mathbf{x}_k, \mu_k) > \Upsilon(\mathbf{x}_k + \mathbf{s}_k, \mu_k) \\ \mathbf{x}_k & \text{otherwise.} \end{cases} \tag{3.13}$$

A new surrogate model, $m_{k+1}(\mathbf{x})$, is then built such that it is fully linear on a region $\mathcal{B}_{k+1}$ having center $\mathbf{x}_{k+1}$ and size Δ_{k+1}. The new fully linear model is constructed using Algorithm 2.2.

3.1.5 Termination

For termination, we must establish that the approximate first-order KKT conditions,

$$\|\nabla f_{\text{high}}(\mathbf{x}_k) + A(\mathbf{x}_k)^\top \lambda(\mathbf{x}_k)\| \leq \epsilon, \tag{3.14}$$

$$\| \left[\mathbf{h}(\mathbf{x}_k)^\top, \mathbf{g}^+(\mathbf{x}_k)^\top\right] \| \leq \epsilon \tag{3.15}$$

are satisfied at $\mathbf{x}_k$, where $A(\mathbf{x}_k)$ is defined to be the Jacobian of all active or violated constraints at $\mathbf{x}_k$,

$$A(\mathbf{x}_k) = \left[\nabla \mathbf{h}(\mathbf{x}_k)^\top, \nabla \mathbf{g}^+(\mathbf{x}_k)^\top\right]^\top, \tag{3.16}$$

and $\lambda(\mathbf{x}_k)$ are Lagrange multipliers. The additional complementarity and sign conditions are assumed to be satisfied from the solution of (3.7). The constraint violation criteria, (3.15), can be evaluated directly. However, the first-order condition, (3.14), cannot be verified directly in the derivative-free case because the gradient, $\nabla f_{\text{high}}(\mathbf{x}_k)$, is unknown. Therefore, for first-order optimality we require two conditions: first-order optimality with the surrogate model, and a sufficiently small trust-region. The first-order optimality condition using the surrogate model is

$$\|\nabla m_k(\mathbf{x}_k) + A(\mathbf{x}_k)^\top \hat{\lambda}(\mathbf{x}_k)\| \leq \max\left[\beta\epsilon, \alpha\Delta_k, a\right] \leq \epsilon, \tag{3.17}$$

where $\hat{\lambda}$ are the Lagrange multipliers computed using the surrogate model and active constraint set estimated from the surrogate model instead of the high-fidelity function. This approximate stationarity condition is similar to what would be obtained using a finite-difference gradient estimate with a fixed step size where the truncation error in the approximate derivative eventually dominates the stationarity measure, but in our case the sufficient decrease modification of the update parameter eventually dominates the stationarity measure [15]. For $\Delta_k \to 0$, we have from (2.7) that $\|\nabla f_{\text{high}}(\mathbf{x}) - \nabla m_k(\mathbf{x})\| \to 0$, and also $\|\hat{\lambda}(\mathbf{x}_k) - \lambda(\mathbf{x}_k)\| \to 0$. Therefore, we have first-order optimality as given in (3.14). In practice, the algorithm is terminated when the constraint violation is small, (3.15), first-order optimality is satisfied on the model, (3.17), and the trust region is small, say $\Delta_k < \epsilon_2$ for a small ϵ_2.

3.1.6 Implementation

The numerical implementation of the multifidelity optimization algorithm, which does not compute the gradient of the high-fidelity objective function, is presented as Algorithm 3.1. A set of possible parameters that may be used in this algorithm is listed in Table 3.1 in Section 3.3. A key element of this algorithm is the logic to switch from the penalty function trust-region subproblem, (3.8), to the subproblem that uses the constraints explicitly, (3.7). Handling the constraints explicitly will generally lead to faster convergence and fewer function evaluations; however, a feasible solution to this subproblem likely does not exist at early

iterations. If either the constraint violation is sufficiently small, $\| \left[\mathbf{h}(\mathbf{x}_k)^\top,\ \mathbf{g}^+(\mathbf{x}_k)^\top\right] \| \leq \epsilon$, or the linearized steps, δ_h, satisfying $\mathbf{h}(\mathbf{x}) + \nabla\mathbf{h}(\mathbf{x})^\top\delta_h = 0$ for all equality and inequality constraints are all smaller than the size of the trust region, then the subproblem with the explicit constraints is attempted. If the optimization fails, then the penalty function subproblem is solved.

This method may be accelerated with the use of multiple lower-fidelity models. The multifidelity filtering technique presented in Chapter 2 to combine estimates from multiple low-fidelity functions into a single maximum likelihood estimate of the high-fidelity function value will work unmodified within this multifidelity optimization framework.

Algorithm 3.1: Multifidelity Objective Trust-Region Algorithm

0: Set initial parameters, a, α, β, ϵ, ϵ_2, η_0, η_1, η_2, γ_0, γ_1, Δ_0, $\Delta_{\max}$, and μ_0. (Recommended values are given in Table 3.1.) Choose initial starting point $\mathbf{x}_0$, and build initial surrogate model $m_0(\mathbf{x})$ fully linear on $\{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_0\| \leq \Delta_0\}$. Set $k = 0$.

1: Update tolerance, $\tau_k = \min[\beta\epsilon, \alpha\Delta_k]$.

2: Choose and solve a trust-region subproblem:

2a: If the constraint violation is small, $\| \left[\mathbf{h}(\mathbf{x}_k)^\top \ \mathbf{g}^+(\mathbf{x}_k)^\top\right] \| \leq \epsilon$, or the maximum linearized step to constraint feasibility for all active and violated constraints is smaller than the current trust region size, Δ_k, then solve:

$$\begin{aligned} \min_{\mathbf{s}_k \in \mathbb{R}^n} \quad & m_k(\mathbf{x}_k + \mathbf{s}_k) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_k + \mathbf{s}_k) = 0 \\ & \mathbf{g}(\mathbf{x}_k + \mathbf{s}_k) \leq 0 \\ & \|\mathbf{s}_k\| \leq \Delta_k, \end{aligned}$$

to convergence tolerance τ_k.

2b: If 2a is not used or fails to converge to the required tolerance, solve the trust-region subproblem:

$$\begin{aligned} \min_{\mathbf{s}_k \in \mathbb{R}^n} \quad & \hat{\Upsilon}_k(\mathbf{x}_k + \mathbf{s}_k, \mu_k) \\ \text{s.t.} \quad & \|\mathbf{s}_k\| \leq \Delta_k. \end{aligned}$$

to convergence tolerance τ_k.

3: If $f_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k)$ has not been evaluated previously, evaluate the high-fidelity function at that point.

3a: Store $f_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k)$ in database.

4: Compute the merit function $\Upsilon(\mathbf{x}_k, \mu_k), \Upsilon(\mathbf{x}_k + \mathbf{s}_k, \mu_k)$, and the surrogate merit function, $\hat{\Upsilon}(\mathbf{x}_k + \mathbf{s}_k, \mu_k)$.

5: Compute the ratio of actual improvement to predicted improvement,

$$\rho_k = \begin{cases} 0 & \hat{\Upsilon}(\mathbf{x}_k, \mu_k) - \hat{\Upsilon}(\mathbf{x}_k + \mathbf{s}_k, \mu_k) < a\Delta_k \\ \frac{\Upsilon(\mathbf{x}_k, \mu_k) - \Upsilon(\mathbf{x}_k + \mathbf{s}_k, \mu_k)}{\hat{\Upsilon}(\mathbf{x}_k, \mu_k) - \hat{\Upsilon}(\mathbf{x}_k + \mathbf{s}_k, \mu_k)} & \text{otherwise.} \end{cases}$$

6: Update the trust region size according to ρ_k,

$$\Delta_{k+1} \quad = \quad \begin{cases} \min\{\gamma_1\Delta_k, \Delta_{\max}\} & \text{if } \eta_1 \leq \rho_k \leq \eta_2 \\ \gamma_0\Delta_k & \text{if } \rho_k \leq \eta_0, \\ \Delta_k & \text{otherwise,} \end{cases}$$

7: Accept or reject the trial point according to improvement in the merit function,

$$\mathbf{x}_{k+1} = \begin{cases} \mathbf{x}_k + \mathbf{s}_k & \Upsilon(\mathbf{x}_k, \mu_k) - \Upsilon(\mathbf{x}_k + \mathbf{s}_k, \mu_k) > 0 \\ \mathbf{x}_k & \text{otherwise.} \end{cases}$$

8: Create new model $m_{k+1}(\mathbf{x})$ fully linear on $\{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_{k+1}\| \leq \Delta_{k+1}\}$.

9: Increment the penalty, $\mu_{k+1} = \max\left[e^{k+1/10}, 1/\Delta_{k+1}^{1.1}\right]$. Increment k.

10: Check for convergence: if $\|\nabla m_k(\mathbf{x}_k) + A(\mathbf{x}_k)^\top\hat{\lambda}\| \leq \epsilon$, $\| \left[\mathbf{h}(\mathbf{x}_k)^\top, \mathbf{g}^+(\mathbf{x}_k)^\top\right] \| \leq \epsilon$, and $\Delta_k \leq \epsilon_2$ the algorithm is converged, otherwise go to step 1.

3.1.7 Theoretical Considerations

This subsection discusses some performance limitations of the proposed algorithm as well as theoretical considerations needed for robustness.

This chapter has not presented a formal convergence theory; at best, such a theory will apply only under many restrictive assumptions. In addition, we can at best guarantee convergence to a near-optimal solution (i.e., optimality to an *a priori* tolerance level and not to stationarity). Forcing the trust region size to decrease every time the sufficient decrease condition is not satisfied means that if the projection of the gradient onto the feasible domain is less than the parameter a, then the algorithm can fail to make progress. This limitation is presented in (3.17); however, it is not seen as severe because a can be set at any value arbitrarily close to (but strictly greater than) zero.

A second limitation is that the model calibration strategy, generating a fully linear model, is theoretically only guaranteed to be possible for functions that are twice continuously differentiable and have Lipschitz-continuous first derivatives. Though this assumption may seem to limit some desired opportunities for derivative-free optimization of functions with noise, noise with certain characteristics like that discussed in [28, Section 9.3], or the case of models with dynamic accuracy as in [31, Section 10.6] can be accommodated in this framework. Approaches such as those in [80] may be used to characterize the noise in a specific problem. However, our algorithm applies even in the case of general noise, where no guarantees can be made on the quality of the surrogate models. As will be shown in the test problem in Section 3.3, in such a case our approach exhibits robustness and significant advantages over gradient-based methods that are susceptible to poor gradient estimates.

Algorithm 3.1 is more complicated than conventional gradient-based trust-region algorithms using quadratic penalty functions, such as the algorithm in [31, Algorithm 14.3.1]. There are three sources of added complexity. First, our algorithm increases the penalty parameter after each trust-region subproblem as opposed to after a completed minimization of the penalty function. This aspect can significantly reduce the number of required high-fidelity evaluations, but adds complexity to the algorithm. Second, our algorithm switches between two trust region subproblems to avoid numerical issues associated with the condi-

tioning of the quadratic penalty function Hessian. The third reason for added complexity is that in derivative-free optimization the size of the trust-region must decrease to zero in order to demonstrate optimality. This aspect serves to add unwanted coupling between the penalty parameter and the size of the trust region, which must be handled appropriately. We now discuss how these two aspects of the algorithm place important constraints on the penalty parameter μ_k.

The requirements for the penalty parameter, μ_k, are that (i) $\lim_{k\to\infty} \mu_k \Delta_k = \infty$, (ii) $\lim_{k\to\infty} \mu_k \Delta_k^2 = 0$, and (iii) $\sum_{k=0}^{\infty} 1/\mu_k$ is finite. The lower bound for the growth of μ_k, that (i) $\lim_{k\to\infty} \mu_k \Delta_k = \infty$, comes from the properties of the minima of quadratic penalty functions presented in [31, Theorem 14.3.1], properties of a fully linear model, and (3.9). If $\mathbf{x}_k$ is at an approximate minimizer of the surrogate quadratic penalty function, (3.6), then a bound on the constraint violation is

$$\| \left[\mathbf{h}(\mathbf{x}_k)^\top \ \mathbf{g}^+(\mathbf{x}_k)^\top \right] \| \leq \frac{\kappa_1(\max\{\alpha\Delta_k, a\} + \kappa_g\Delta_k) + \|\lambda(\mathbf{x}^*)\| + \kappa_2\|\mathbf{x}_k - \mathbf{x}^*\|}{\mu_k}, \tag{3.18}$$

where $\mathbf{x}^*$ is a KKT point of 3.1, and κ_1, κ_2 are finite positive constants. If $\{\mu_k\Delta_k\}$ diverges then an iteration exists where both the bound, (3.18), holds (the trust region size must be large enough that a feasible point exists in the interior) and the constraint violation is less than the given tolerance ϵ. This enables the switching between the two trust region subproblems, (3.8) and (3.7). The upper bound for the growth of μ_k, that
(ii) $\lim_{k\to\infty} \mu_k \Delta_k^2 = 0$, comes from the smoothness of the quadratic penalty function. To establish an upper bound for the Hessian 2-norm for the subproblem in (3.8) we compare the value of the merit function at a point $\Upsilon(\mathbf{x}_k + \mathbf{p}, \mu_k)$ with its linearized prediction based on $\Upsilon(\mathbf{x}_k, \mu_k)$, $\tilde{\Upsilon}(\mathbf{x}_k + \mathbf{p}, \mu_k)$. If κ_{fg} is the Lipschitz constant for $\nabla f_{\text{high}}(\mathbf{x})$, κ_{cm} is the maximum Lipschitz constant for the constraints, and κ_{cgm} is the maximum Lipschitz constant for a

constraint gradient, we can show that

$$\|\Upsilon(\mathbf{x}_k + \mathbf{p}, \mu_k) - \tilde{\Upsilon}(\mathbf{x}_k + \mathbf{p}, \mu_k)\| \leq \tag{3.19}$$
$$\left[\kappa_{fg} + \mu_k \left(\kappa_{cm} \| \left[\mathbf{h}(\mathbf{x}_k)^\top \ \mathbf{g}^+(\mathbf{x}_k)^\top\right] \| + \kappa_{cgm} \|A(\mathbf{x}_k)\| \|\mathbf{p}\| + \kappa_{cm}\kappa_{cgm} \|\mathbf{p}\|^2\right)\right] \|\mathbf{p}\|^2.$$

We have a similar result for $\hat{\Upsilon}(\mathbf{x}, \mu_k)$ by replacing κ_{fg} with the sum $\kappa_{fg} + \kappa_g$, using the definition of a fully linear model, and by bounding $\|\mathbf{p}\|$ by Δ_k. Therefore, we may show the error in a linearized prediction of the surrogate model will go to zero and that Lipschitz-type smoothness is ensured provided that the sequence $\{\mu_k \Delta_k^2\}$ converges to zero regardless of the constraint violation.

The final requirement, that (iii) $\sum_{k=0}^{\infty} 1/\mu_k$ is finite, comes from the need for the size of the trust region to go to zero in gradient-free optimization. The sufficient decrease condition, that $\rho_k = 0$ unless $\hat{\Upsilon}(\mathbf{x}_k, \mu_k) - \hat{\Upsilon}(\mathbf{x}_k + \mathbf{s}_k, \mu_k) \geq a\Delta_k$ ensures that the trust region size decreases unless the change in the merit function $\Upsilon(\mathbf{x}_k, \mu_k) - \Upsilon(\mathbf{x}_k + \mathbf{s}_k, \mu_k) \geq \eta_0 a \Delta_k$. This provides an upper bound on the total number of times in which the size of the trust region is kept constant or increased. We have assumed that the merit function is bounded from below and also that the trust-region iterates remain within a level-set $L(\mathbf{x}_0, \mu_0)$ as defined by (3.2). We now consider the merit function written in an alternate form,

$$\Upsilon(\mathbf{x}_k, \mu_k) = f_{\text{high}}(\mathbf{x}_k) + \frac{\mu_k}{2} \| \left[\mathbf{h}(\mathbf{x}_k)^\top, \mathbf{g}^+(\mathbf{x}_k)^\top\right] \|^2. \tag{3.20}$$

From (3.18), if μ_k is large enough such that the bound on the constraint violation is less than unity, we may use the bound on the constraint violation in (3.18) to show that an upper bound on the total remaining change in the merit function is

$$f_{\text{high}}(\mathbf{x}_k) - \min_{\mathbf{x} \in L(\mathbf{x}_0, \mu_0)} f_{\text{high}}(\mathbf{x}) + \tag{3.21}$$
$$\frac{\left[\kappa_1(\max\{\alpha\Delta_k, a\} + \kappa_g \Delta_k) + \|\lambda(\mathbf{x}^*)\| + \kappa_2 \|\mathbf{x}_k - \mathbf{x}^*\|\right]^2}{\mu_k}.$$

Each term in the numerator is bounded from above because Δ_k is always bounded from

above by $\Delta_{\max}$, $\|\lambda(\mathbf{x}^*)\|$ is bounded from above because $\mathbf{x}^*$ is a regular point, and $\|\mathbf{x}_k - \mathbf{x}^*\|$ is bounded from above because $L(\mathbf{x}_0, \mu_0)$ is a compact set. Therefore, if the series $\{1/\mu_k\}$ has a finite sum, then the total remaining improvement in the merit function is finite. Accordingly, the sum of the series $\{\Delta_k\}$ must be finite, and $\Delta_k \to 0$ as $k \to \infty$. The prescribed sequence for $\{\mu_k\}$ in our algorithm satisfies these requirements for a broad range of problems.

3.2 Multifidelity Objective and Constraint Optimization

This section considers a more general constrained optimization problem with a computationally expensive objective function and computationally expensive constraints. The specific problem considered is where the gradients for both the expensive objective and expensive constraints are either unavailable, unreliable or expensive to estimate. Accordingly, the multifidelity optimization problem in (3.1) is augmented with the high-fidelity constraint, $c_{\text{high}}(\mathbf{x}) \leq 0$. In addition, we have a low-fidelity estimate of this constraint function, $c_{\text{low}}(\mathbf{x})$, which estimates the same metric as $c_{\text{high}}(\mathbf{x})$, but with unknown error. Therefore, our goal is to find the vector $\mathbf{x} \in \mathbb{R}^n$ of n design variables that solves the nonlinear constrained optimization problem,

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & f_{\text{high}}(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}) = 0 \\ & \mathbf{g}(\mathbf{x}) \leq 0 \\ & c_{\text{high}}(\mathbf{x}) \leq 0, \end{aligned} \tag{3.22}$$

where $\mathbf{h}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$ represent vectors of inexpensive equality and inequality constraints with derivatives that are either known or may be estimated cheaply. The same assumptions for the expensive objective function formulation given in Section 3.1.1 are made for the functions presented in this formulation. It is also necessary to make an assumption similar to that in Section 3.1.2, that a quadratic penalty function with the new high-fidelity constraint is

bounded from below within an initial expanded level-set. A point of note is that multiple high-fidelity constraints can be used if an initial point $\mathbf{x}_0$ is given that is feasible with respect to all constraints; however, due to the effort required to construct approximations of the multiple high-fidelity constraints, it is recommended that all of the high-fidelity constraints be combined into a single high-fidelity constraint through, as an example, a discriminant function [85, 98].

The optimization problem in (3.22) is solved in two phases. First, the multifidelity optimization method presented in Section 3.1 is used to find a feasible point, and then an interior point formulation is used to find a minimum of the optimization problem in (3.22). The interior point formulation is presented in Section 3.2.2 and the numerical implementation is presented in Section 3.2.3.

3.2.1 Finding a Feasible Point

This algorithm begins by finding a point that is feasible with respect to all of the constraints by applying Algorithm 1 to the optimization problem

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & c_{\text{high}}(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}) = 0 \\ & \mathbf{g}(\mathbf{x}) \leq 0, \end{aligned} \tag{3.23}$$

until a point that is feasible with respect to the constraints in (3.22) is found. If this optimization problem is unconstrained (i.e., there are no constraints $\mathbf{h}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$) then the trust-region algorithm of [27] is used with the multifidelity calibration method of [71]. The optimization problem in (3.23) may violate one of the assumptions for Algorithm 3.1 in that $c_{\text{high}}(\mathbf{x})$ may not be bounded from below. This issue will be addressed in the numerical implementation of the method in Section 3.2.3.

3.2.2 Interior Point Trust-region Method

Once a feasible point is found we minimize the high-fidelity objective function ensuring that we never again violate the constraints, that is we solve (3.22). This is accomplished in two steps, first by solving trust region subproblems that use fully linear surrogate models for both the high-fidelity objective function and the high-fidelity constraint. Second, the trust region step is evaluated for feasibility and any infeasible step is rejected. The surrogate model for the objective function is $m_k(\mathbf{x})$ as defined in (2.2). For the constraint, the surrogate model, $\bar{m}_k(\mathbf{x})$, is defined as

$$\bar{m}_k(\mathbf{x}) = c_{\text{low}}(\mathbf{x}) + \bar{e}_k(\mathbf{x}). \tag{3.24}$$

From the definition of a fully linear model, (2.7) and (2.8), $\bar{m}_k(\mathbf{x})$ satisfies

$$\|\nabla c_{\text{high}}(\mathbf{x}) - \nabla \bar{m}_k(\mathbf{x})\| \leq \kappa_{cg}\Delta_k \quad \forall \mathbf{x} \in \mathcal{B}_k, \tag{3.25}$$

$$|c_{\text{high}}(\mathbf{x}) - \bar{m}_k(\mathbf{x})| \leq \kappa_c \Delta_k^2 \quad \forall \mathbf{x} \in \mathcal{B}_k. \tag{3.26}$$

In addition, we require that our procedure to construct fully linear models ensures that at the current design iterate, the fully linear models exactly interpolate the function they are modeling,

$$m_k(\mathbf{x}_k) = f_{\text{high}}(\mathbf{x}_k), \tag{3.27}$$

$$\bar{m}_k(\mathbf{x}_k) = c_{\text{high}}(\mathbf{x}_k). \tag{3.28}$$

This is required so that every trust-region subproblem is feasible at its initial point $\mathbf{x}_k$.

The trust-region subproblem is

$$\begin{aligned} \min_{\mathbf{s}_k \in \mathbb{R}^n} \quad & m_k(\mathbf{x}_k + \mathbf{s}_k) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_k + \mathbf{s}_k) = 0 \\ & \mathbf{g}(\mathbf{x}_k + \mathbf{s}_k) \leq 0 \\ & \bar{m}_k(\mathbf{x}_k + \mathbf{s}_k) \leq \max\{c_{\text{high}}(\mathbf{x}_k), -v\Delta_k\} \\ & \|\mathbf{s}_k\| \leq \Delta_k. \end{aligned} \tag{3.29}$$

The surrogate model constraint does not have zero as a right hand side to account for the fact the algorithm is looking for interior points. The right hand side, $\max\{c_{\text{high}}(\mathbf{x}_k), -v\Delta_k\}$, ensures that the constraint is initially feasible and that protection of constraint violation decreases to zero as the number of iterations increase to infinity. The constant v must be greater than α, which is defined as part of the termination tolerance τ_k in (3.9). The trust-region subproblem is solved to the same termination tolerance as the multifidelity objective function formulation, $\|\nabla_x \mathfrak{L}_k\| \leq \tau_k$, where $\mathfrak{L}_k$ is the Lagrangian of (3.29).

The center of the trust region is updated if a decrease in the objective function is found at a feasible point,

$$\mathbf{x}_{k+1} = \begin{cases} \mathbf{x}_k + \mathbf{s}_k & \text{if } f_{\text{high}}(\mathbf{x}_k) > f_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k) \\ & \quad \text{and } c_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k) \leq 0 \\ \mathbf{x}_k & \text{otherwise,} \end{cases} \tag{3.30}$$

with $\mathbf{h}(\mathbf{x}_k + \mathbf{s}_k) = 0$ and $\mathbf{g}(\mathbf{x}_k + \mathbf{s}_k) \leq 0$ already satisfied in (3.29). The trust-region size update must ensure that the predictions of the surrogate models are accurate and that the size of the trust region goes to zero in the limit as the number of iterations goes to infinity. Therefore, we again impose a sufficient decrease condition, that the change in the objective

function is at least a constant, a, multiplying Δ_k,

$$\Delta_{k+1} = \begin{cases} \min\{\gamma_1 \Delta_k, \Delta_{\max}\} & \text{if } f_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k) - f_{\text{high}}(\mathbf{x}_k) \geq a\Delta_k \\ & \quad \text{and } c_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k) \leq 0 \\ \gamma_0 \Delta_k & \text{otherwise.} \end{cases} \tag{3.31}$$

New surrogate models, $m_{k+1}(\mathbf{x})$ and $\bar{m}_{k+1}(\mathbf{x})$, are then built such that they are fully linear on a region $\mathcal{B}_{k+1}$ having center $\mathbf{x}_{k+1}$ and size Δ_{k+1}. The new fully linear models are constructed using the procedure of [117] with the calibration technique of [71].

3.2.3 Multifidelity Objective and Constraint Implementation

The numerical implementation of this multifidelity optimization algorithm is presented as Algorithm 3.2. A set of possible parameters that may be used in this algorithm is listed in Table 3.1 in Section 3.3. An important implementation issue with this algorithm is finding the initial feasible point. Algorithm 3.1 is used to minimize the high-fidelity constraint value subject to the constraints with available derivatives in order to find a point that is feasible. However, Algorithm 3.1 uses a quadratic penalty function to handle the constraints with available derivatives if the constraints are violated. The convergence of a penalty function requires that the objective function is bounded from below, therefore a more general problem than (3.23) to find an initial feasible point is to use,

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} \quad & \left[\max\{c_{\text{high}}(\mathbf{x}) + d, 0\}\right]^2 \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}) = 0 \\ & \mathbf{g}(\mathbf{x}) \leq 0. \end{aligned} \tag{3.32}$$

The maximization in the objective prevents the need for the high-fidelity constraint to be bounded from below, and the constraint violation is squared to ensure the gradient of the objective is continuous. The constant d is used to account for the fact that the surrogate model will have some error in its prediction of $c_{\text{high}}(\mathbf{x})$, so looking for a slightly negative

value of the constraint may save iterations as compared to seeking a value that is exactly zero. For example, if $d \geq \kappa_c \Delta_k^2$ and $\bar{m}_k(\mathbf{x}) + d = 0$ then (3.26) guarantees that $c_{\text{high}}(\mathbf{x}) \leq 0$.

There is a similar consideration in the solution of (3.29), where a slightly negative value of the surrogate constraint is desired. If this subproblem is solved with an interior point algorithm this should be satisfied automatically; however, if a sequential quadratic programming method is used the constraint violation will have a numerical tolerance that is either slightly negative or slightly positive. It may be necessary to bias the subproblem to look for a value of the constraint that is more negative than the optimizer constraint violation tolerance to ensure the solution is an interior point. This feature avoids difficulties with the convergence of the trust-region iterates.

A final implementation note is that if a high-fidelity constraint has numerical noise or steep gradients it may be wise to shrink the trust region at a slower rate, increasing γ_0. This will help to ensure that the trust-region does not decrease to zero at a suboptimal point.

3.3 Supersonic Airfoil Design Test Problem

This section presents results of the two multifidelity optimization algorithms on a supersonic airfoil design problem.

3.3.1 Problem Setup

The supersonic airfoil design problem has 11 parameters: the angle of attack, 5 spline points on the upper surface and 5 spline points on the lower surface. However, there is an actual total of 7 spline points on both the upper and lower surfaces because the leading and trailing edges must be sharp for the low-fidelity methods used. The airfoils are constrained such that the minimum thickness-to-chord ratio is 0.05 and that the thickness everywhere on the airfoil must be positive. In addition, there are lower and upper bounds for all spline points.

As presented in Chapter 2, three supersonic airfoil analysis models are available: a linearized panel method, a nonlinear shock-expansion theory method, and Cart3D, an Euler CFD solver [1]. Note, that Cart3D has a finite convergence tolerance so there is some nu-

Algorithm 3.2: Multifidelity Objective and Constraint Trust-Region Algorithm

-1: Find a feasible design vector using Algorithm 3.1 to iterate on:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \quad & \left[\max\{c_{\text{high}}(\mathbf{x}) + d, 0\}\right]^2 \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}) = 0 \\ & \mathbf{g}(\mathbf{x}) \leq 0, \end{aligned}$$

-1a: Store all evaluations of $c_{\text{high}}(\mathbf{x})$.

0: Set initial parameters, a, α, β, d, υ, ϵ, ϵ_2, γ_0, γ_1, Δ_0, $\Delta_{\max}$, and μ_0. (Recommended values are given in Table 3.1.) Build initial surrogate models $m_0(\mathbf{x})$, $\bar{m}_0(\mathbf{x})$ fully linear on $\{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_0\| \leq \Delta_0\}$, where $\mathbf{x}_0$ is the terminal point of step -1. Set $k = 0$.

1: Update tolerance, $\tau_k = \min\left[\beta\epsilon, \alpha\Delta_k\right]$.

2: Solve the trust-region subproblem:

$$\begin{aligned} \min_{\mathbf{s}_k\in\mathbb{R}^n} \quad & m_k(\mathbf{x}_k + \mathbf{s}_k) \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}_k + \mathbf{s}_k) = 0 \\ & \mathbf{g}(\mathbf{x}_k + \mathbf{s}_k) \leq 0 \\ & \bar{m}_k(\mathbf{x}_k + \mathbf{s}_k) \leq \min\{c_{\text{high}}(\mathbf{x}_k), -\upsilon\Delta_k\} \\ & \|\mathbf{s}_k\| \leq \Delta_k. \end{aligned}$$

3: If $f_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k)$ or $c_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k)$ have not been evaluated previously, evaluate the high-fidelity functions at that point.

3a: Store $f_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k)$ and $c_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k)$ in a database.

4: Accept or reject the trial point according to:

$$\mathbf{x}_{k+1} = \begin{cases} \mathbf{x}_k + \mathbf{s}_k & \text{if } f_{\text{high}}(\mathbf{x}_k) > f_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k) \text{ and } c_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k) \leq 0 \\ \mathbf{x}_k & \text{otherwise.} \end{cases}$$

5: Update the trust region size according to:

$$\Delta_{k+1} = \begin{cases} \min\{\gamma_1\Delta_k, \Delta_{\max}\} & \text{if } f_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k) - f_{\text{high}}(\mathbf{x}_k) \geq a\Delta_k \text{ and } c_{\text{high}}(\mathbf{x}_k + \mathbf{s}_k) \leq 0 \\ \gamma_0\Delta_k & \text{otherwise.} \end{cases}$$

6: Create new models $m_{k+1}(\mathbf{x})$ and $\bar{m}_{k+1}(\mathbf{x})$ fully linear on $\{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_{k+1}\| \leq \Delta_{k+1}\}$. Increment k.

7: Check for convergence, if the trust region constraint is inactive, $\|\nabla m(\mathbf{x}_k) + A(\mathbf{x}_k)^\top \hat{\lambda}(\mathbf{x}_k) + \nabla\bar{m}(\mathbf{x}_k)\lambda_{\bar{m}}\| \leq \epsilon$, and $\Delta_k \leq \epsilon_2$, the algorithm is converged, otherwise go to step 1.

merical noise in the lift and drag predictions. In addition, because random airfoils are used as initial conditions, Cart3D may fail to converge, in which case the results of the panel method are used. Figure 2-3 shows computed pressure distributions for each of the models for a 5% thick biconvex airfoil at Mach 1.5 and 2° angle of attack. Table 2.3 provides the estimated lift and drag coefficients for the same airfoil and indicates the approximate level of accuracy of the codes with respect to each other.

We first present single-fidelity results using state-of-the-art derivative-free methods. Then the following sections present results for three optimization examples each using this airfoil problem to demonstrate the capabilities of the optimization algorithms presented. In the first example, Section 3.3.3, the airfoil drag is minimized using the constrained multifidelity objective function formulation presented in Section 3.1 with only the simple geometric constraints. In the second example, Section 3.3.4, the airfoil lift-to-drag ratio is maximized subject to a constraint that the drag coefficient is less than 0.01, where the constraint is handled with the multifidelity framework presented in Section 3.2. In the final example, Section 3.3.5, the airfoil lift-to-drag ratio is maximized subject to the constrained drag coefficient and both the objective function and the constraints are handled with the multifidelity framework presented in Section 3.2. The initial airfoils for all problems are randomly generated and likely will not satisfy the constraints.

The three multifidelity airfoil problems are solved with four alternative optimization algorithms: Sequential Quadratic Programming (SQP) [76]; a first-order consistent multifidelity trust-region algorithm that uses a SQP formulation and an additive correction [5]; the high-fidelity-gradient-free approach presented in this chapter using a Gaussian radial basis function and a fixed spatial correlation parameter, $\xi = 2$; and the approach presented in this chapter using a maximum likelihood estimate to find an improved correlation length, $\xi = \xi^*$. The Gaussian correlation functions used in these results are all isotropic. An anisotropic correlation function (i.e., choosing a correlation length for each direction in the design space) may speed convergence of this algorithm and reduce sensitivity to Hessian conditioning. The parameters used for the optimization algorithm are presented in Table 3.1. The fully linear models are constructed using Algorithm 2.2 and the parameters stated Table 2.2.

Constant	Description	Value
a	Sufficient decrease constant	1×10^{-4}
α	Convergence tolerance multiplier	1×10^{-2}
β	Convergence tolerance multiplier	1×10^{-2}
d	Artificial lower bound for constraint value	1
v	Constraint violation conservatism factor	0.1
ϵ, ϵ_2	Termination Tolerance	5×10^{-4}
γ_0	Trust region contraction ratio	0.5
γ_1	Trust region expansion ratio	2
η_0	Trust region contraction criterion	0.25
η_1, η_2	Trust region expansion criterion	$0.75, 2.0$
Δ_0	Initial trust region radius	1
Δ_{max}	Maximum trust region size	20
μ_k	Penalty parameter	$\max\left[e^{k/10}, 1/\Delta_k^{1.1}\right]$
δ_x	Finite difference step	1×10^{-5}

Table 3.1: List of constants used in the algorithm. All parameters used in constructing the radial basis function error model are listed in Table 2.2. These parameter values are based on recommendations for unconstrained trust-region algorithms and through numerical testing appear to have good performance for an assortment of problems.

3.3.2 Single-fidelity Derivative-free Optimization

For benchmark purposes, we first solve the airfoil optimization problem using three single-fidelity gradient-free optimization methods: the Nelder-Mead simplex algorithm [81], the global optimization method DIRECT [51], and the constrained gradient-free optimizer, COBYLA [88, 89]. The test case is to minimize the drag of supersonic airfoil estimated with a panel method, subject to the airfoil having positive thickness everywhere and at least 5% thickness to chord ratio. This test case is a lower-fidelity version of the example in Section 3.3.3. Nelder-Mead simplex and DIRECT are unconstrained optimizers that use a quadratic penalty function known to perform well on this problem [71], while COBYLA handles the constraints explicitly. On this problem, starting from ten random initial airfoils the best observed results were 5,170 function evaluations for the Nelder-Mead simplex algorithm, over 11,000 evaluations for DIRECT, and 6,284 evaluations for COBYLA. Such high numbers of function evaluations mean that these single-fidelity gradient-free algorithms are too expensive for use with an expensive forward solver, such as Cart3D.

3.3.3 Multifidelity Objective Function Results

This section presents optimization results in terms of the number of high-fidelity function evaluations required to find the minimum drag for a supersonic airfoil at Mach 1.5 with only geometric constraints on the design. Two cases are tested: the first uses the shock-expansion method as the high-fidelity function and the panel method as the low-fidelity function; the second uses Cart3D as the high-fidelity function and the panel method as the low-fidelity function. These problems are solved using the multifidelity optimization algorithm for a computationally expensive objective function and constraints with available derivatives presented in Section 3.1.

The average numbers of high-fidelity function evaluations required to find a locally optimal design starting from random initial airfoils are presented in Table 3.2. The results show that our approach uses approximately 78% fewer high-fidelity function evaluations than SQP and approximately 30% fewer function evaluations than the first-order consistent trust-region method using finite difference gradient estimates. In addition, Figure 3-1 compares the objective function and constraint violation histories verse the number of high-fidelity evaluations for these methods as well as DIRECT and Nelder-Mead simplex from a representative random initial airfoil for the case with the shock-expansion method as the high-fidelity function. For the single-fidelity optimization using Cart3D, the convergence results were highly sensitive to the finite difference step length. The step size required tuning, and the step with the highest success rate was used. A reason for this is that the initial airfoils were randomly generated, and the convergence tolerance of Cart3D for airfoils with sharp peaks and negative thickness was large compared with the airfoils near the optimal design. This sensitivity of gradient-based optimizers to the finite difference step length highlights the benefit of gradient-free approaches, especially when constraint gradient estimates become poor.

High-Fidelity	Low-Fidelity	SQP	First-Order TR	RBF, $\xi = 2$	RBF, $\xi = \xi^*$
Shock-Expansion	Panel Method	314 (-)	110 (-65%)	73 (-77%)	68 (-78%)
Cart3D	Panel Method	359*(-)	109 (-70%)	80 (-78%)	79 (-78%)

Table 3.2: The average number of high-fidelity function evaluations to minimize the drag of a supersonic airfoil with only geometric constraints. The asterisk for the Cart3D results means a significant fraction of the optimizations failed and the average is taken over fewer samples. The numbers in parentheses indicate the percentage reduction in high-fidelity function evaluations relative to SQP.

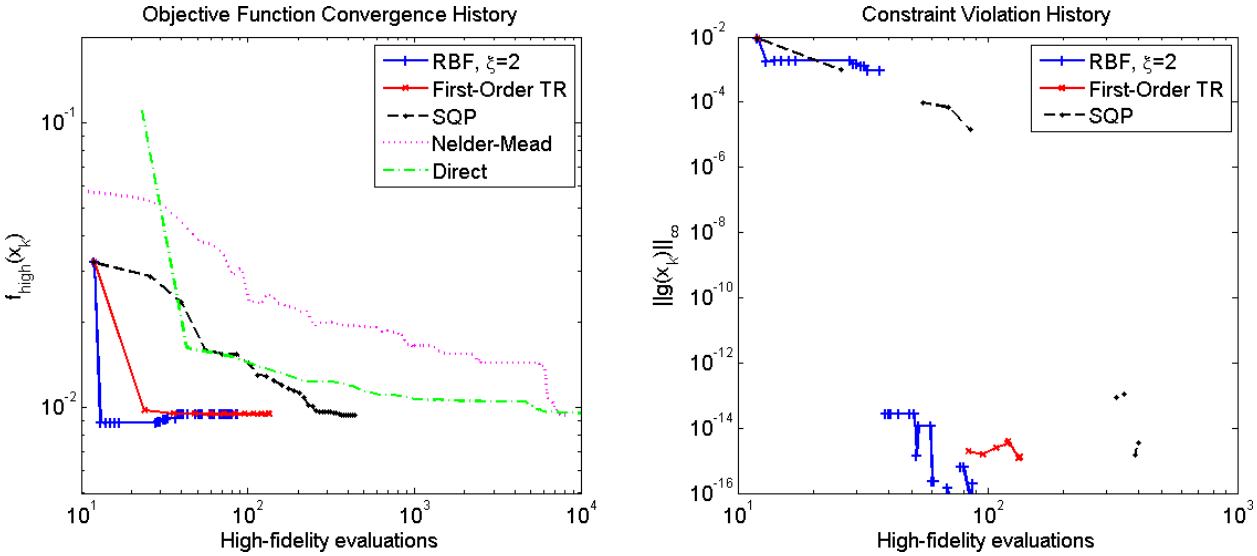


Figure 3-1: Convergence history for minimizing the drag of an airfoil using the shock-expansion theory method as the high-fidelity function subject to only geometric constraints. The methods presented are our calibration approach, a first-order consistent multifidelity trust-region algorithm, sequential quadratic programming, DIRECT, and Nelder-Mead simplex. Both DIRECT and Nelder-Mead simplex use a fixed penalty function to handle the constraints, so only an objective function value is shown. COBYLA and BOBYQA were attempted, but failed to find the known solution to this problem. On the constraint violation plot, missing points denote a feasible iterate, and the sudden decrease in the constraint violation for the RBF calibration approach at 37 high-fidelity evaluations (13^{th} iteration, $\mu_k = 9.13 \times 10^5$) is when the algorithm switches from solving (3.8) to solving (3.7).

High-Fidelity	Low-Fidelity	SQP	First-Order TR	RBF, $\xi = 2$	RBF, $\xi = \xi^*$
Shock-Expansion	Panel Method	827 (-)	104 (-87%)	104 (-87%)	115 (-86%)
Cart3D	Panel Method	909*(-)	100 (-89%)	103 (-89%)	105 (-88%)

Table 3.3: The average number of high-fidelity constraint evaluations required to maximize the lift-to-drag ratio of a supersonic airfoil estimated with a panel method subject to a multifidelity constraint. The asterisk for the Cart3D results means a significant fraction of the optimizations failed and the average is taken over fewer samples. The numbers in parentheses indicate the percentage reduction in high-fidelity function evaluations relative to SQP.

3.3.4 Multifidelity Constraint Results

This section presents optimization results in terms of the number of function evaluations required to find the maximum lift-to-drag ratio for a supersonic airfoil at Mach 1.5 subject to both geometric constraints and the requirement that the drag coefficient is less than 0.01. The lift-to-drag ratio is computed with the panel method; however, the drag coefficient constraint is handled using the multifidelity technique presented in Section 3.2. Two cases are examined: in the first the shock-expansion method models the high-fidelity constraint and the panel method models the low-fidelity constraint; in the second case, Cart3D models the high-fidelity constraint and the panel method models the low-fidelity constraint. Table 3.3 presents the average number of high-fidelity constraint evaluations required to find the optimal design using SQP, a first-order consistent trust-region algorithm and the multifidelity techniques developed in this chapter. A significant decrease (almost 90%) in the number of high-fidelity function evaluations is observed when compared with SQP. Performance is almost the same as the first-order consistent trust-region algorithm.

3.3.5 Multifidelity Objective Function and Constraint Results

This section presents optimization results in terms of the number of function evaluations required to find the maximum lift-to-drag ratio for a supersonic airfoil at Mach 1.5 subject to geometric constraints and the requirement that the drag coefficient is less than 0.01. In this case, both the lift-to-drag ratio and the drag coefficient constraint are handled using the multifidelity technique presented in Section 3.2. In the first case, the shock-expansion

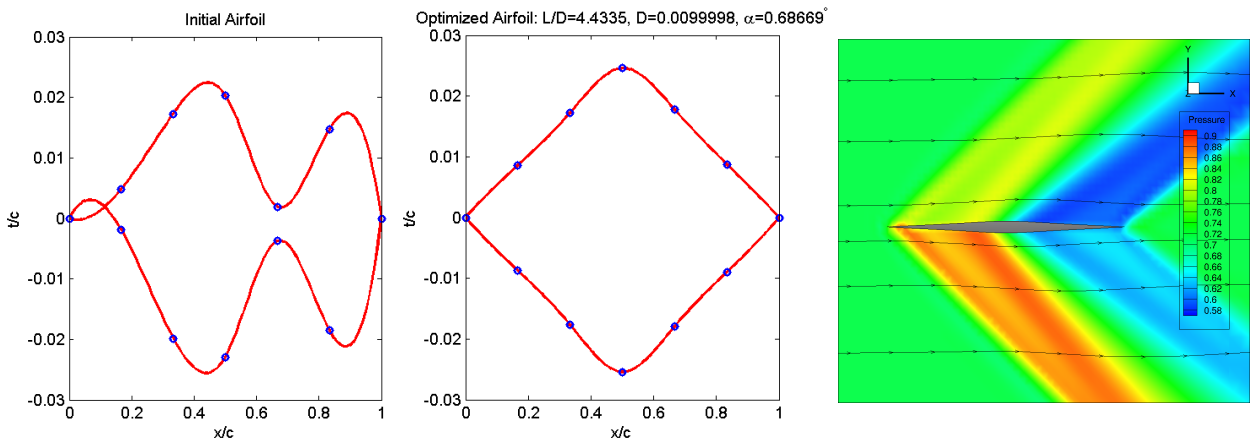
	High-Fidelity	Low-Fidelity	SQP	First-Order TR	RBF, $\xi = 2$	RBF, $\xi = \xi^*$
Objective:	Shock-Expansion	Panel Method	773 (-)	132 (-83%)	93 (-88%)	90 (-88%)
Constraint:	Shock-Expansion	Panel Method	773 (-)	132 (-83%)	97 (-87%)	96 (-88%)
	High-Fidelity	Low-Fidelity	SQP	First-Order TR	RBF, $\xi = 2$	RBF, $\xi = \xi^*$
Objective:	Cart3D	Panel Method	1168*(-)	97 (-92%)	104 (-91%)	112 (-90%)
Constraint:	Cart3D	Panel Method	2335*(-)	97 (-96%)	115 (-95%)	128 (-94%)

Table 3.4: The average number of high-fidelity objective function and high-fidelity constraint evaluations to optimize a supersonic airfoil for a maximum lift-to-drag ratio subject to a maximum drag constraint. The asterisk for the Cart3D results means a significant fraction of the optimizations failed and the average is taken over fewer samples. The numbers in parentheses indicate the percentage reduction in high-fidelity function evaluations relative to SQP.

method is the high-fidelity analysis used to estimate both metrics of interest and the panel method is the low-fidelity analysis. In the second case, Cart3D is the high-fidelity analysis and the panel method is the low-fidelity analysis. The optimal airfoils are shown in Figure 3-2. Table 3.4 presents the number of high-fidelity function evaluations required to find the optimal design using SQP, a first-order consistent trust-region algorithm and the techniques developed in this chapter. Again a significant reduction (about 90%) in the number of high-fidelity function evaluations, both in terms of the constraint and the objective, are observed compared with SQP, and a similar number of high-fidelity function evaluations are observed when compared with the first-order consistent trust region approach using finite differences.

3.4 Summary

This chapter has presented two algorithms for multifidelity constrained optimization of computationally expensive functions when their derivatives are not available. The first method minimizes a high-fidelity objective function without using its derivative while satisfying constraints with available derivatives. The second method minimizes a high-fidelity objective without using its derivative while satisfying both constraints with available derivatives and an additional high-fidelity constraint without an available derivative. Both of these methods support multiple lower-fidelity models through the use of a multifidelity filtering technique without any modifications to the methods. For the supersonic airfoil design example consid-



(a) Initial airfoil. (b) Optimal shock-expansion theory airfoil. (c) Optimal Cart3D flow solution.

Figure 3-2: Initial airfoil and supersonic airfoil with the maximum lift-to-drag ratio having drag less than 0.01 and 5% thickness at Mach 1.5.

ered here, the multifidelity methods resulted in approximately 90% reduction in the number of high-fidelity function evaluations compared to solution with a single-fidelity sequential quadratic programming method. In addition, the multifidelity methods performed similarly to a first-order consistent trust-region algorithm with gradients estimated using finite difference approximations. This shows that derivative-free multifidelity methods provide significant opportunity for optimization of computationally expensive functions without available gradients.

The behavior of the gradient-free algorithms presented is slightly atypical of nonlinear programming methods. For example, the convergence rate of these gradient-free algorithms is rapid initially and then slows when close to an optimal solution. In contrast, convergence for a gradient-based method is often initially slow and then accelerates when close to an optimal solution (e.g., as approximate Hessian information becomes more accurate in a quasi-Newton approach). Also, gradient-based optimizers typically find the local optimal solution nearest the initial design. Although by virtue of the physics involved, the presented examples have unique optimal solutions, in a general problem the local optimum to which these gradient-free algorithms converge may not be the one in the immediate vicinity of the initial iterate. An example of this behavior is the case when the initial iterate is itself a

local optimum, the surrogate model may not capture this fact and the iterate may move to a different point in the design space with a lower function value.

In the case of hard constraints, or when the objective function fails to exist if the constraints are violated, it is still possible to use Algorithm 2.2. After the initial feasible point is found, no design iterate will be accepted if the high-fidelity constraint is violated. Therefore the overall flow of the algorithm is unchanged. What must be changed is the technique to build fully linear models. In order to build a fully linear model, the objective function must be evaluated at a set of $n + 1$ points that span $\mathbb{R}^n$. When the objective function can be evaluated outside the feasible region, the constraints do not influence the construction of the surrogate model. However, when the objective function does not exist where the constraints are violated, then the points used to construct the surrogate model must all be feasible and this restricts the shape of the feasible region. Specifically, this requirement prohibits equality constraints and means that strict linear independent constraint qualification must be satisfied everywhere in the design space (preventing two inequality constraints from mimicking an equality constraint). If these two additional conditions hold, then it will be possible to construct fully linear models everywhere in the feasible design space and use this algorithm to optimize computationally expensive functions with hard constraints.

Lastly, we comment on the applicability of the proposed multifidelity approach. Though this chapter presents no formal convergence theory, and at best that theory will only apply if many restrictive assumptions hold (for example, assumptions on smoothness, constraint qualification, and always using a fully linear surrogate) our numerical experiments indicate the robustness of the approach in application to a broader class of problems. For example, the Cart3D CFD model employed in our case studies does not satisfy the Lipschitz continuity requirements, due to finite convergence tolerances in determining the CFD solution; however, with the aid of the smooth calibrated surrogates combined with the trust-region model management, our multifidelity method is successful in finding locally optimal solutions. Another example is a high-fidelity optimal solution for which constraint qualification conditions are not satisfied. In the algorithms presented, the design vector iterate will approach a local minimum and the sufficient decrease test for the change in the objective function value will

fail. This causes the size of the trust region to decay to zero around the local minimum even though the KKT conditions may not be satisfied.

In summary, this chapter has presented a multifidelity optimization algorithm that does not require estimating gradients of high-fidelity functions, enables the use of multiple low-fidelity models, enables optimization of functions with hard constraints, and exhibits robustness over a broad class of optimization problems, even when non-smoothness is present in the objective function and/or constraints. For airfoil design problems, this approach has been shown to perform similarly in terms of the number of function evaluations to finite-difference-based multifidelity optimization methods. This suggests that the multifidelity derivative-free approach is a promising alternative for the wide range of problems where finite-difference gradient approximations are unreliable.

Chapter 4

Gradient-Exploiting Multifidelity Optimization

Chapters 2 and 3 presented multifidelity optimization methods for the case when no high-fidelity sensitivity information is available. There are however many cases in which the gradients of expensive functions are available, for instance, analytically, through automatic differentiation, adjoint methods, or finite-differences. This chapter develops a multifidelity optimization algorithm based on an existing gradient-based multifidelity optimization framework, but which includes a strategy for Bayesian model calibration that uses both the function value and the gradient of the high-fidelity function. The calibration is accomplished by interpolating the high-fidelity function at points where the function value and gradient are known within the vicinity of a trust-region, this enables the surrogate to be well-calibrated to the high-fidelity function in a region of the design space and to construct the surrogate model with little cost. The technique is compared with a single-fidelity sequential quadratic programming method and a conventional first-order trust-region method on both a two-dimensional structural optimization and an airfoil design problem. In both problems adjoint formulations are used to provide inexpensive sensitivity information.

Section 4.1 of this chapter provides an overview of the trust-region algorithm, the fundamental ideas of Bayesian model calibration, and the technique developed to construct the Cokriging surrogate. Section 4.2 demonstrates this algorithm on a 26-dimensional structural

design problem and Section 4.3 demonstrates this method on an 11-dimensional airfoil design problem.

4.1 Optimization Method

In this chapter we consider optimization problems where the high-fidelity function comes from the solution of a partial differential equation or includes state variables. For example, we may have a problem with a high-fidelity objective, $\mathcal{J}_{\text{high}}(\mathbf{x}, \mathbf{u})$, with a vector $\mathbf{u}$ of l state variables set by l residual or state equations $\mathcal{R}_{\text{high}}(\mathbf{x}, \mathbf{u}) = 0$. In problems of this form we write the optimization formulation as,

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n, \mathbf{u}\in\mathbb{R}^l} \quad & \mathcal{J}_{\text{high}}(\mathbf{x}, \mathbf{u}) \\ \text{s.t. } & \mathcal{R}_{\text{high}}(\mathbf{x}, \mathbf{u}) = 0 \\ & \mathbf{h}(\mathbf{x}) = 0 \\ & \mathbf{g}(\mathbf{x}) \leq 0. \end{aligned} \tag{4.1}$$

When considering problems of the form of Eq. 4.1 we can eliminate the state variables by solving the residual equations. Thus, these systems can be written in the form of Eq. 3.1, however, we now are now a setting in where inexpensive gradient estimates for $f_{\text{high}}(\mathbf{x})$, $h(\mathbf{x})$, and $g(\mathbf{x})$ are available. In the following presentation of the multifidelity optimization method the more general form of Eq. 3.1 is used.

4.1.1 Trust Region Method

To solve Eq. 3.1 when gradient information is available we may use any of the multifidelity trust-region algorithms by Alexandrov [3, 5]. This chapter uses a modified form of sequential quadratic programming (SQP) approximation model management, which generates a sequence of design iterates $\mathbf{x}_k$ that converge to an optimum of the high-fidelity problem Eq. 3.1. At each trust-region iteration we minimize a surrogate model $m_k(\mathbf{x})$ of the high-fidelity function. We define the surrogate model using the additive error model presented in

Eq. 2.2. The requirements for convergence of this algorithm are that, (i) at each trust-region iteration the surrogate model satisfies first-order consistency requirements,

$$m_k(\mathbf{x}_k) = f_{\text{high}}(\mathbf{x}_k) \tag{4.2}$$

$$\nabla m_k(\mathbf{x}_k) = \nabla f_{\text{high}}(\mathbf{x}_k) \tag{4.3}$$

and (ii) that there exists a constant $\kappa_{bhm} < \infty$ that is an upper bound for the 2-norm of the surrogate model Hessian [2, 31].

The trust-region algorithm uses the l_1 penalty function,

$$\Upsilon(\mathbf{x}, \mu_k) = f_{\text{high}}(\mathbf{x}) + \mu_k \|\mathbf{h}(\mathbf{x})\|_1 + \mu_k \|\mathbf{g}^+(\mathbf{x})\|_1, \tag{4.4}$$

where $\mathbf{g}^+(\mathbf{x})$ is the inequality constraint violation and μ_k is a penalty parameter. In addition, a surrogate penalty function, $\hat{\Upsilon}(\mathbf{x}, \mu_k)$, which replaces the high-fidelity function value with the surrogate model value, is needed for trust region updating,

$$\hat{\Upsilon}(\mathbf{x}, \mu_k) = m_k(\mathbf{x}) + \mu_k \|\mathbf{h}(\mathbf{x})\|_1 + \mu_k \|\mathbf{g}^+(\mathbf{x})\|_1. \tag{4.5}$$

The penalty parameter, μ_k, must be larger than the smallest Lagrange multiplier associated with the optimality conditions for Eq. 3.1 which can only be estimated at the beginning of the optimization [5].

From an initial design vector $\mathbf{x}_0$, trust-region size Δ_0, and penalty μ_0, the trust-region iterates are generated by finding a step, $\mathbf{s}_k$, that minimizes the surrogate model subject to linearized design constraints and trust-region constraint,

$$\begin{aligned} \min_{\mathbf{s}_k \in \mathbb{R}^n} \quad & m_k(\mathbf{x}_k + \mathbf{s}_k) \\ \text{s.t.} \quad & \nabla \mathbf{h}(\mathbf{x}_k)^\top \mathbf{s}_k + \mathbf{h}(\mathbf{x}_k) = 0 \\ & \nabla \mathbf{g}(\mathbf{x}_k)^\top \mathbf{s}_k + \mathbf{g}(\mathbf{x}_k) \leq 0 \\ & \|\mathbf{s}_k\|_\infty \leq \Delta_k, \end{aligned} \tag{4.6}$$

or subject to the full design constraints and trust-region constraint,

$$\begin{aligned} \min_{\mathbf{s}_k \in \mathbb{R}^n} \quad & m_k(\mathbf{x}_k + \mathbf{s}_k) \\ \text{s.t. } & \mathbf{h}(\mathbf{x}_k + \mathbf{s}_k) = 0 \\ & \mathbf{g}(\mathbf{x}_k + \mathbf{s}_k) \leq 0 \\ & \|\mathbf{s}_k\|_\infty \leq \Delta_k. \end{aligned} \tag{4.7}$$

Eq. 4.6 is the general subproblem which allows for the current iterate to be infeasible and for first-order consistent surrogate models of the constraints to be used. However, if the constraints are inexpensive compared to the high-fidelity objective function and the current iterate is feasible, Eq. 4.7 can speed finding the optimal high-fidelity design since the constraints are included explicitly in the trust region subproblem. After either subproblem has been solved, the performance of the surrogate model is estimated with the parameter ρ_k, which is the ratio of the actual improvement in the high-fidelity penalty function with the improvement estimated by the surrogate penalty function,

$$\rho_k = \frac{\Upsilon(\mathbf{x}_k, \mu_k) - \Upsilon(\mathbf{x}_k + \mathbf{s}_k, \mu_k)}{\Upsilon(\mathbf{x}_k, \mu_k) - \hat{\Upsilon}(\mathbf{x}_k + \mathbf{s}_k, \mu_k)}. \tag{4.8}$$

The size of the trust region is updated based on the performance of the surrogate model. If the surrogate model predicted the high-fidelity behavior well, the trust region is expanded, if the prediction is poor the trust region is contracted. Specifically, we update the trust region size according to,

$$\Delta_{k+1} = \begin{cases} \min\{\gamma_1 \Delta_k, \Delta_{\max}\} & \text{if } \rho_k \geq \eta_1 \\ \gamma_0 \|\mathbf{s}_k\| & \text{if } \rho_k \leq \eta_2 \\ \Delta_k & \text{otherwise,} \end{cases} \tag{4.9}$$

where $0 < \eta_2 < \eta_1 < 1$, $\gamma_0 < 1$ and $\gamma_1 > 1$. In addition we move the trust region if the step

results in a decrease in the penalty function,

$$\mathbf{x}_{k+1} = \begin{cases} \mathbf{x}_k + \mathbf{s}_k & \text{if } \Upsilon(\mathbf{x}_k + \mathbf{s}_k, \mu_k) < \Upsilon(\mathbf{x}_k, \mu_k) \\ \mathbf{x}_k & \text{otherwise.} \end{cases} \tag{4.10}$$

We then create a new surrogate model $m_{k+1}(\mathbf{x})$ on the new trust region $\{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_{k+1}\|_\infty \leq \Delta_{k+1}\}$. If the optimality conditions are not sufficiently satisfied we repeat the algorithm. In addition, if the size of the trust region becomes sufficiently small we also terminate the algorithm. The trust-region algorithm is summarized as Algorithm 4.1.

4.1.2 Bayesian Model Calibration

In conventional trust-region algorithms the surrogate model for optimization is created by correcting the low-fidelity model such that at the center of the trust-region the first-order consistency requirements, Eqs. 4.2 and 4.3, are satisfied. Using the additive correction model, Eq. 2.2, this would correspond to

$$e_k(\mathbf{x}) = f_{\text{high}}(\mathbf{x}_k) - f_{\text{low}}(\mathbf{x}_k) + \left[\nabla f_{\text{high}}(\mathbf{x}_k) - \nabla f_{\text{low}}(\mathbf{x}_k)\right]^\top (\mathbf{x} - \mathbf{x}_k)\,. \tag{4.11}$$

In the trust-region framework presented in Section 4.1.1, this calibration approach is provably convergent to a high-fidelity optimum. There are two possible drawbacks with this calibration technique. The first drawback is that the calibration may only be first-order, which means in the worst case the convergence rate could be linear and extremely slow, although we note that quasi-second-order trust region calibration approaches have been proposed in the literature [36]. The second drawback is that both the function value and gradient are known at all previous design iterates encountered by the trust region algorithm and this information is typically only used to approximate the Hessian, if at all.

The idea of Bayesian model calibration is to use all available information to estimate unknown high-fidelity information. In the multifidelity setting, this translates to using all previous information about the error between the high- and low-fidelity models to calibrate

Algorithm 4.1: Iteration k of a Trust-Region Algorithm

0: Choose initial design vector $\mathbf{x}_0$, initial trust region size Δ_0, initial penalty μ_0, and initial surrogate model $m_0(\mathbf{x})$ that satisfies Eqs. 4.2 and 4.3 at $\mathbf{x}_0$ (Algorithm 4.2 can be used to create $m_0(\mathbf{x})$).

1: Solve the trust-region subproblem using nonlinear programming techniques to find the step, $\mathbf{s}_k$, that solves,

$$\begin{aligned} &\min_{\mathbf{s}_k \in \mathbb{R}^n} \quad m_k(\mathbf{x}_k + \mathbf{s}_k) \\ &\text{s.t. } \nabla \mathbf{h}(\mathbf{x}_k)^\top \mathbf{s}_k + \mathbf{h}(\mathbf{x}_k) = 0 \\ &\qquad \nabla \mathbf{g}(\mathbf{x}_k)^\top \mathbf{s}_k + \mathbf{g}(\mathbf{x}_k) \leq 0 \\ &\qquad \|\mathbf{s}_k\|_\infty \leq \Delta_k. \end{aligned} \qquad \text{or} \qquad \begin{aligned} &\min_{\mathbf{s}_k \in \mathbb{R}^n} \quad m_k(\mathbf{x}_k + \mathbf{s}_k) \\ &\text{s.t. } \mathbf{h}(\mathbf{x}_k + \mathbf{s}_k) = 0 \\ &\qquad \mathbf{g}(\mathbf{x}_k + \mathbf{s}_k) \leq 0 \\ &\qquad \|\mathbf{s}_k\|_\infty \leq \Delta_k. \end{aligned}$$

2: Evaluate penalty functions $\Upsilon(\mathbf{x}_k + \mathbf{s}_k, \mu_k)$ and $\hat{\Upsilon}(\mathbf{x}_k + \mathbf{s}_k, \mu_k)$.

3: Compute the ratio of actual improvement to predicted improvement,

$$\rho_k = \frac{\Upsilon(\mathbf{x}_k, \mu_k) - \Upsilon(\mathbf{x}_k + \mathbf{s}_k, \mu_k)}{\Upsilon(\mathbf{x}_k, \mu_k) - \hat{\Upsilon}(\mathbf{x}_k + \mathbf{s}_k, \mu_k)}.$$

4: Update the trust region size according to ρ_k,

$$\Delta_{k+1} = \begin{cases} \min\{\gamma_1 \Delta_k, \Delta_{\max}\} & \text{if } \rho_k \geq \eta_1 \\ \gamma_0 \|\mathbf{s}_k\| & \text{if } \rho_k \leq \eta_2 \\ \Delta_k & \text{otherwise.} \end{cases}$$

5: Accept or reject the trial point,

$$\mathbf{x}_{k+1} = \begin{cases} \mathbf{x}_k + \mathbf{s}_k & \text{if } \Upsilon(\mathbf{x}_k + \mathbf{s}_k, \mu_k) < \Upsilon(\mathbf{x}_k, \mu_k) \\ \mathbf{x}_k & \text{otherwise.} \end{cases}$$

6: Create a new surrogate model $m_{k+1}(\mathbf{x})$ that satisfies Eqs. 4.2 and 4.3 using Algorithm 4.2 on the trust-region $\{\mathbf{x} : \|\mathbf{x} - \mathbf{x}_{k+1}\|_\infty \leq \Delta_{k+1}\}$.

7: Check for convergence. If the solution is not optimal and the trust-region size is sufficient, return to step 1.

Algorithm 4.2: Procedure to generate a Cokriging model

0: Select initial calibration point at $\mathbf{x}_k$ and initial correlation vector ξ.
1: Randomly order all high-fidelity sample points within $\|\mathbf{x} - \mathbf{x}_k\| \leq \theta_1 \Delta_k$.
2: Maximize the likelihood function of the Cokriging model using an optimization algorithm that is robust to non-smoothness, such as a pattern-search or simulated annealing.
  2a: For a given ξ, test each high-fidelity sample point in the order chosen in step 1 to see if after adding that point to the Cokriging model the eigenvalues of $\mathbf{R}_C$ with smallest absolute value and the largest absolute value satisfy, $|\Lambda_1| \geq \theta_2$ and $|\Lambda_{q(n+1)}| \leq \theta_3 |\Lambda_1|$. If the conditions are satisfied, add that calibration point (up to user set maximum, $q_{\max}$ calibration points), if the conditions are not satisfied reject that calibration point.
  2b: Compute the likelihood of the Cokriging model, Θ, using Eq. 4.19.
3: If more than one calibration point is in the Cokriging model, compute $\mathbf{R}_C^{-1}$ and ψ_C using the maximum likelihood correlation vector, ξ. Otherwise use the additive correction given in Eq. 4.11.

the low-fidelity model. The calibrated model provides an estimate of the new design that has the largest high-fidelity improvement. However, the amount of high-fidelity information collected during an optimization is likely too much to use when creating a surrogate model that will only be used for one trust-region iteration. Therefore, the next section addresses how high-fidelity information is selected for a calibration procedure that guarantees convergence to a high-fidelity optimum and limits the computational effort required to construct a surrogate model.

4.1.3 Cokriging

The Cokriging Bayesian model calibration technique is demonstrated in Figure 4-1. The figure shows that at all calibration points the surrogate model exactly interpolates the high-fidelity function and it has the same gradient as the high-fidelity function. The Cokriging model also estimates the uncertainty in its prediction using a maximum likelihood variance estimate. It can be observed that the uncertainty in the Cokriging estimates are zero at all calibration points and increase with distance away from the calibration points. There are two Cokriging formulations, direct [24, 42], and indirect [61, 67]. The former augments a Kriging model with analytical gradients and the latter adds additional sample points to a Kriging fit

such that the Cokriging model interpolates points replicating a Taylor series. This chapter uses the direct Cokriging formulation. We first summarize the Cokriging method, then we present the requirements for convergence in a trust-region framework, and we finally present our method to construct Cokriging models that satisfies both the convergence requirements of a trust-region algorithm and limits the computational effort required to construct the surrogate models.

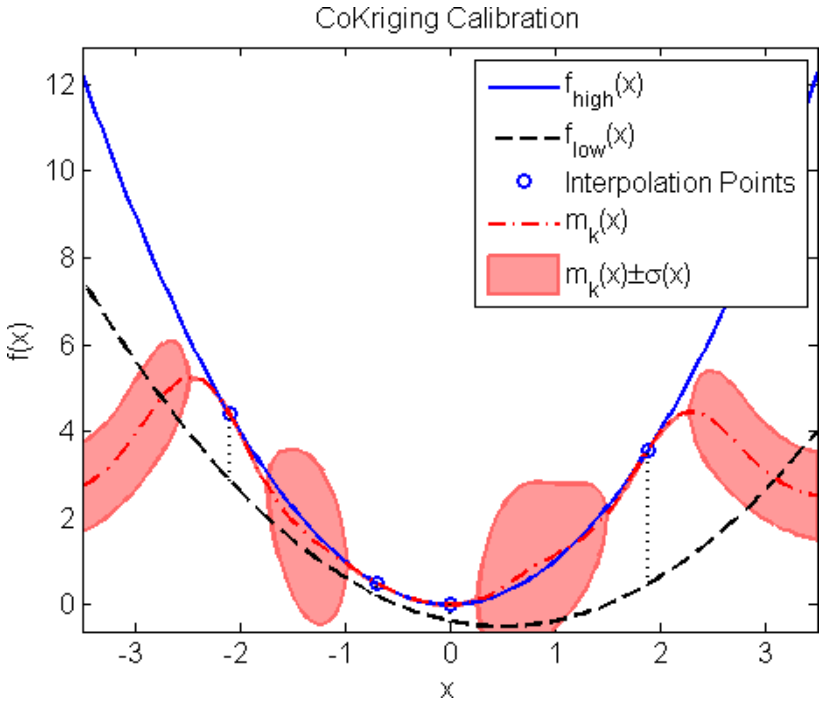


Figure 4-1: Demonstration of a Cokriging surrogate model for the simple univariate function $f_{\text{high}}(x) = x^2$.

Kriging methods come from the field of Geostatistics and have the underlying assumption that function values from nearby samples are correlated [75]. The correlation function we use is an anisotropic Gaussian,

$$\phi(\mathbf{x}_i, \mathbf{x}_j) = \exp\left[-\sum_{p=1}^{n} \xi^{(p)} \left(x_i^{(p)} - x_j^{(p)}\right)^2\right], \tag{4.12}$$

where the correlation between points $\mathbf{x}_i$ and $\mathbf{x}_j$, $\phi(\mathbf{x}_i, \mathbf{x}_j)$, has n spatial tuning parameters, $\xi^{(p)}$ for each design variable $x^{(p)}$. The tuning parameters are estimated using a Bayesian

maximum likelihood estimator.

A Cokriging model uses q calibration points in $\mathbb{R}^n$ to generate a correlation vector with length $q(1+n)$,

$$\Phi(\mathbf{x}) = \Big[\phi(\mathbf{x},\mathbf{x}_1),\ldots,\phi(\mathbf{x},\mathbf{x}_q),\nabla\phi(\mathbf{x},\mathbf{x}_1)^\top,\ldots,\nabla\phi(\mathbf{x},\mathbf{x}_q)^\top\Big]^\top, \tag{4.13}$$

where all derivatives of $\phi(\mathbf{x},\mathbf{x}_j)$ are with respect to $\mathbf{x}$. In addition, we need to define the Cokriging correlation matrix that describes the influence of each calibration point on the other calibration points,

$$\mathbf{R}_C = \left[\begin{array}{c|c} \overbrace{\begin{array}{ccc}\phi(\mathbf{x}_1,\mathbf{x}_1)\ldots\phi(\mathbf{x}_1,\mathbf{x}_q)\\ \vdots\quad\ddots\quad\vdots\\ \phi(\mathbf{x}_q,\mathbf{x}_1)\ldots\phi(\mathbf{x}_q,\mathbf{x}_q)\end{array}}^{q} & \overbrace{\begin{array}{ccc}\nabla\phi(\mathbf{x}_1,\mathbf{x}_1)^\top\ldots\nabla\phi(\mathbf{x}_1,\mathbf{x}_q)^\top\\ \vdots\quad\ddots\quad\vdots\\ \nabla\phi(\mathbf{x}_q,\mathbf{x}_1)^\top\ldots\nabla\phi(\mathbf{x}_q,\mathbf{x}_q)^\top\end{array}}^{n\times q} \\ \hline \begin{array}{ccc}\nabla\phi(\mathbf{x}_1,\mathbf{x}_1)\ldots\nabla\phi(\mathbf{x}_1,\mathbf{x}_q)\\ \vdots\quad\ddots\quad\vdots\\ \nabla\phi(\mathbf{x}_q,\mathbf{x}_1)\ldots\nabla\phi(\mathbf{x}_q,\mathbf{x}_q)\end{array} & \begin{array}{ccc}\nabla^2\phi(\mathbf{x}_1,\mathbf{x}_1)\ldots\nabla^2\phi(\mathbf{x}_1,\mathbf{x}_q)\\ \vdots\quad\ddots\quad\vdots\\ \nabla^2\phi(\mathbf{x}_q,\mathbf{x}_1)\ldots\nabla^2\phi(\mathbf{x}_q,\mathbf{x}_q)\end{array} \end{array}\right] \begin{array}{l} \left.\begin{array}{l}\\\\\\\end{array}\right\} q \\ \\ \left.\begin{array}{l}\\\\\\\end{array}\right\} n\times q \end{array} \tag{4.14}$$

where all first and second derivatives of $\phi(\cdot,\cdot)$ are with respect to the first variable. Using the correlation vector and correlation matrix we can interpolate the values in the vector,

$$\begin{aligned}\mathbf{y}_C = [&f_{\text{high}}(\mathbf{x}_1) - f_{\text{low}}(\mathbf{x}_1),\ldots,f_{\text{high}}(\mathbf{x}_q) - f_{\text{low}}(\mathbf{x}_q),\\ &\nabla f_{\text{high}}(\mathbf{x}_1) - \nabla f_{\text{low}}(\mathbf{x}_1),\ldots,\nabla f_{\text{high}}(\mathbf{x}_q) - \nabla f_{\text{low}}(\mathbf{x}_q)]^\top\end{aligned} \tag{4.15}$$

by defining the vector of ones and zeros,

$$\mathbf{z}_C = \Big[\mathbf{1}^q,\mathbf{0}^{q\cdot n}\Big]^\top \tag{4.16}$$

and a generalized least-squares constant term,

$$\psi_C = \left(\mathbf{z}_C^\top \mathbf{R}_C^{-1} \mathbf{z}_C\right)^{-1} \mathbf{z}_C^\top \mathbf{R}_C^{-1} \mathbf{y}_C. \tag{4.17}$$

This leads to a final Cokriging error model that has the form,

$$e_k(\mathbf{x}) = \psi_C + \Phi(\mathbf{x})^\top \mathbf{R}_C^{-1} \left(\mathbf{y}_C - \mathbf{z}_C \psi_C\right). \tag{4.18}$$

To estimate the spatial correlation parameters ξ we maximize the likelihood function, Θ, which is the probability of observing these data if these data had been generated by a Gaussian process [24, 93]. The likelihood is only a function of the observed data and the correlation parameters ξ. Ignoring the contribution of the gradient to the likelihood, the partial likelihood only depends on the Kriging portion of the Cokriging model, so $\mathbf{R}_c^{(1:q\times 1:q)}$ indicates that only the first q terms of the Cokriging vectors are used,

$$\Theta = -\frac{q \ln \hat{\sigma}^2 + \ln \left|\mathbf{R}_c^{(1:q\times 1:q)}\right|}{2}. \tag{4.19}$$

Where $\hat{\sigma}^2$ maximizes the Kriging portion of the likelihood function, Eq. 4.19, [100]

$$\hat{\sigma}^2 = \frac{\left(\mathbf{y}_C^{(1:q)} - \mathbf{1}^q \psi\right)^\top \left(\mathbf{R}_c^{(1:q\times 1:q)}\right)^{-1} \left(\mathbf{y}_C^{(1:q)} - \mathbf{1}^q \psi\right)}{q}, \tag{4.20}$$

and

$$\psi = \left([\mathbf{1}^q]^\top \left(\mathbf{R}_c^{(1:q\times 1:q)}\right)^{-1} \mathbf{1}^q\right)^{-1} [\mathbf{1}^q]^\top \left(\mathbf{R}_c^{(1:q\times 1:q)}\right)^{-1} \mathbf{y}_C^{(1:q)}. \tag{4.21}$$

To use a Cokriging model within a trust-region algorithm we must ensure the first-order consistency requirements, Eqs. 4.2 and 4.3, are satisfied, and that the norm of the Cokriging Hessian is bounded. The first-order consistency requirements are satisfied provided the current trust-region iterate, $\mathbf{x}_k$, is a calibration point and that $\mathbf{R}_C$ is not ill-conditioned. To satisfy the requirement that the Hessian of the surrogate model has bounded norm, we first assume that both the high- and low-fidelity functions have bounded Hessian norms for all $\mathbf{x}$.

What must follow is that the Hessian of $e_k(\mathbf{x})$ has bounded norm, or equivalently the Hessian of $\Phi(\mathbf{x})^\top \mathbf{R}_C^{-1} (\mathbf{y}_C - \mathbf{z}_C \psi_C)$ has bounded norm. The vector $(\mathbf{y}_C - \mathbf{z}_C \psi_C)$ has bounded norm by virtue of assumptions on the high- and low-fidelity function. Furthermore, it can be shown by differentiating the correlation function in Eq. 4.12 four times that the maximum absolute value of any second derivative term for the first q components of $\Phi(\mathbf{x})^\top$ is less than $\max_p 2\xi^{(p)}$, and the maximum absolute value of any second derivative term for the remaining qn components is less than $\max_p 3.904 \left(\xi^{(p)}\right)^{3/2}$. Therefore, using properties of the 2-norm and ∞-norm, we can establish that the 2-norm of the Hessian of $\Phi(\mathbf{x})^\top \mathbf{1}$ is bounded by $[q(1+n)]^{3/2} \max\left\{\max_p 2\xi^{(p)}, \max_p 3.904 \left(\xi^{(p)}\right)^{3/2}\right\}$. Now, provided $\|\mathbf{R}_C^{-1}\|_2$ is bounded, we have ensured that the 2-norm of the Cokriging model Hessian is bounded. We address both this criterion and the conditioning of $\mathbf{R}_C$ during the construction of the Cokriging model.

To construct the Cokriging model, we want to limit the number of calibration points so the size of $\mathbf{R}_C$ remains tractable to invert repeatedly. Therefore, we set a maximum number of calibration points that we may use, $q_{\max}$. In addition, because the trust-region algorithm only requires accuracy within the trust-region we only want to include points in the calibration that will affect the shape of the surrogate model within the trust region. So we allow a user-set distance parameter, θ_1, that controls the extent to which the Cokriging model calibrates locally as opposed to globally. Specifically, any point at which the high-fidelity function value and gradient are known that is located within $\|\mathbf{x} - \mathbf{x}_k\| \leq \theta_1 \Delta_k$, $\theta_1 > 0$ is a candidate calibration point. The first-step in constructing the Cokriging model is to select an initial vector of correlation parameters ξ_0 and to select $\mathbf{x}_k$ as the initial calibration point. We then randomly order all candidate calibration points.

To find the Cokriging model with the maximum likelihood correlation parameters and that satisfies all trust-region requirements, we use a greedy approach. We add any candidate calibration point to the Cokriging model provided that we remain able to bound the condition number of $\mathbf{R}_C$ and to bound $\|\mathbf{R}_C^{-1}\|_2$. These two criteria are satisfied if the eigenvalue of $\mathbf{R}_C$ with the smallest absolute value, $|\Lambda_1|$, is greater than a constant, $\theta_2 > 0$, and the eigenvalue with the greatest absolute value, $\Lambda_{q(n+1)}$, is less than or equal to $\theta_3 |\Lambda_1|$, for a constant $\theta_3 > 1$. This means the parameter θ_3 represents the maximum allowable condition number

for the correlation matrix. The upper limit for θ_3 depends on the numerical precision and inversion algorithm used. After all the candidate points have been tested or $q_{\max}$ points have been selected, the likelihood of the Cokriging model, Θ is computed. A pattern-search algorithm is used to find the correlation vector, ξ, giving the Cokriging model with the maximum likelihood. An optimization algorithm that is not highly sensitive to non-smooth functions must be used because for each correlation vector considered the calibration points used may vary. This means that the likelihood function will likely have non-smooth features. A summary of the procedure to generate Cokriging models is presented as Algorithm 4.2. Table 4.1 lists the parameter values used for the sample problems presented in the next two sections.

Parameter	Description	value
$q_{\max}$	Maximum calibration points	10
Δ_0	Initial trust region size	$\max\{5, \|\mathbf{x}_0\|_\infty\}$
$\Delta_{\max}$	Maximum trust region size	20
$\Delta_{\min}$	Minimum trust region size	10^{-6}
γ_0	Trust region contraction ratio	0.5
γ_1	Trust region expansion ratio	2
ϵ	Termination tolerances	1×10^{-4}
η_1	Trust-region expansion criteria	0.75
η_2	Trust-region contraction criteria	0.25
θ_1	Trust-region neighborhood size	10^3
θ_2	Min. eigenvalue	10^{-9}
θ_3	Max. condition number	10^8
	Max. pattern-search iterations	$10n$

Table 4.1: Values of the optimization parameters used.

4.2 Structural Design Problem

Typical structural design problems have deflection or stress requirements with low-weight as an objective or additional constraint. As an example problem we minimize the deflection of a two-dimensional hook subjected to a bearing load and maximum weight constraint. This objective function is linear in the state variables, however, a nonlinear objective will not change the derivation. In the finite element formulation used, the state variables are nodal

displacements, $\mathbf{u}$, and our objective function can be written as

$$\mathcal{J}_{\text{high}}(\mathbf{x}, \mathbf{u}) = \mathbf{C}(\mathbf{x})\mathbf{u}, \tag{4.22}$$

where $\mathbf{C}(\mathbf{x})$ is an output matrix that only depends on the twenty-six design variables representing the geometry of the hook, $\mathbf{x}$, and not the state variables. In lieu of computing displacements the output matrix could easily be modified to compute element stresses using Hooke's Law. The equations of state are obtained using a Ritz finite element formulation to minimize the potential energy of the system, and in discretized form are

$$\mathcal{R}_{\text{high}}(\mathbf{x}, \mathbf{u}) = \mathbf{K}(\mathbf{x})\mathbf{u} - \mathbf{f} = 0, \tag{4.23}$$

where $\mathbf{K}(\mathbf{x})$ is the stiffness matrix and $\mathbf{f}$ is the vector of applied nodal forces. The formal structural optimization problem to minimize the deflection of the hook subjected to a bearing load is,

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^{26}, \mathbf{u}\in\mathbb{R}^{l}} \quad & \mathbf{C}(\mathbf{x})\mathbf{u} \\ \text{s.t. } & \mathbf{K}(\mathbf{x})\mathbf{u} - \mathbf{f} = 0 \\ & w(\mathbf{x}) - w_{\max} \leq 0 \\ & g(\mathbf{x}) \leq 0, \end{aligned} \tag{4.24}$$

where $w(\mathbf{x})$ is the weight of the structure, $w_{\max}$ is the maximum allowable weight, and $g(\mathbf{x}) \leq 0$ represents 31 constraints ensuring geometric feasibility of the hook — positive thickness and clearance to apply the bearing load.

To use our multifidelity optimization method, we combine our objective function $\mathcal{J}_{\text{high}}(\mathbf{x}, \mathbf{u})$ and state equations $\mathcal{R}_{\text{high}}(\mathbf{x}, \mathbf{u}) = 0$ into a single function of the design variables $f_{\text{high}}(\mathbf{x})$ that is the value of our objective function after the state equations have been satisfied. To compute the gradient of $f_{\text{high}}(\mathbf{x})$, we solve the adjoint equation,

$$\frac{\partial \mathcal{J}_{\text{high}}}{\partial \mathbf{u}} = \Psi^{\top} \frac{\partial \mathcal{R}_{\text{high}}}{\partial \mathbf{u}}, \tag{4.25}$$

for the adjoint variables, Ψ. For this structural optimization problem the adjoint equation is,

$$\mathbf{K}^\top(\mathbf{x})\Psi(\mathbf{x}) = \mathbf{C}^\top(\mathbf{x}). \tag{4.26}$$

The gradient of our objective function with respect to the design variables can be written as,

$$\frac{d\mathcal{J}_{\text{high}}}{d\mathbf{x}} = \frac{\partial \mathcal{J}_{\text{high}}}{\partial \mathbf{x}} + \frac{\partial \mathcal{J}_{\text{high}}}{\partial \mathbf{u}}\frac{\partial \mathbf{u}}{\partial \mathbf{x}} \tag{4.27}$$

and similarly, the gradient of the equations of state with respect to the design variables can be written,

$$\frac{d\mathcal{R}_{\text{high}}}{d\mathbf{x}} = \frac{\partial \mathcal{R}_{\text{high}}}{\partial \mathbf{x}} + \frac{\partial \mathcal{R}_{\text{high}}}{\partial \mathbf{u}}\frac{\partial \mathbf{u}}{\partial \mathbf{x}} = 0. \tag{4.28}$$

Substituting Eqs. 4.25 and 4.28 into Eq. 4.27, we obtain the gradient of our objective function with the state equations satisfied as

$$\frac{df_{\text{high}}}{d\mathbf{x}} = \frac{\partial \mathcal{J}_{\text{high}}}{\partial \mathbf{x}} - \Psi^\top \frac{\partial \mathcal{R}_{\text{high}}}{\partial \mathbf{x}}, \tag{4.29}$$

or for this structural optimization problem,

$$\frac{df_{\text{high}}}{d\mathbf{x}} = \left(\frac{\partial \mathbf{C}(\mathbf{x})}{\partial \mathbf{x}} - \Psi^\top(\mathbf{x})\frac{\partial \mathbf{K}(\mathbf{x})}{\partial \mathbf{x}}\right)\mathbf{u}(\mathbf{x}). \tag{4.30}$$

The objective function being linear in $\mathbf{u}(\mathbf{x})$ is a simplification, and a typical problem will likely be to minimize the maximum stress in the material. Although maximum stress is a non-smooth objective, Kreisselmeier−Steinhauser functions could be used to lump element stresses into a single smooth maximum stress objective [72, 73, 87], or a high-norm, such as $\|\cdot\|_8$, could be used to smoothly approximate the maximum stress [115]. Using these functions as objectives does not alter the derivation of the gradient, provided the chain rule is used in computing the derivatives. In the general case, this adjoint-based gradient calculation

requires only two forward solves, or one inversion, of the stiffness matrix to compute the gradient of the objective, regardless of the number of design variables. By comparison, the method of direct sensitivities requires evaluation of the term $\frac{\partial \mathbf{K}^{-1}(\mathbf{x})}{\partial \mathbf{x}}$, and finite difference approximations require at least $n+1$, where n is the number of design variables, function evaluations.

To find the optimal hook design, we use both high- and low-fidelity finite element models of the hook. The two models are shown in Figure 4-2, the high-fidelity model has a state vector with 1770 degrees of freedom (dof) and the low-fidelity model has a state vector with 276 dof. For the optimal hook design, the coarse discretization of the low-fidelity model

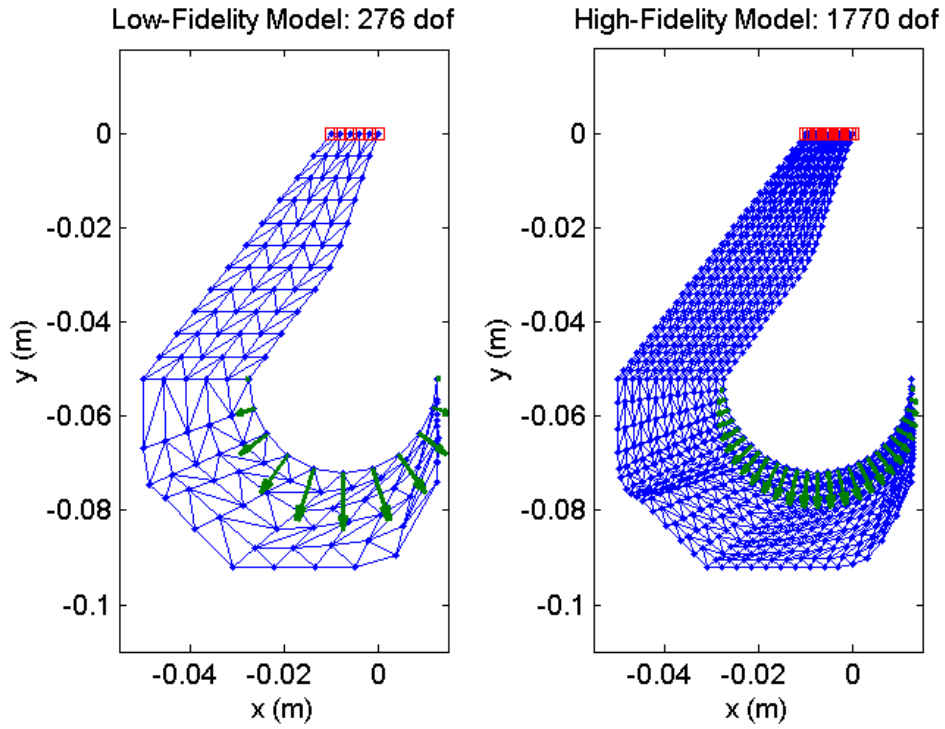


Figure 4-2: Comparison of the high- and low-fidelity structural models. Both models have the same 26 design variables.

leads to a 31.6% lower deflection estimate than the high-fidelity model. In addition, for this hook design, Figure 4-3 shows that the two models predict significantly different stress distributions.

Table 4.2 shows that using this coarsely discretized low-fidelity model, the conventional

	SQP	First-Order TR	Calibration
Mean	232 (-)	40 (-83%)	27.5 (-88%)
Std. Dev.	171	12.7	10.2

Table 4.2: The average number of high-fidelity function evaluations to minimize the deflection of a hook subjected to a bearing load from random initial geometries. The numbers in parentheses indicate the percentage reduction in high-fidelity function evaluations relative to SQP.

first-order consistent trust-region and our Bayesian calibration approach are able to significantly reduce the number of high-fidelity function calls compared with a single-fidelity SQP method. In addition, the Bayesian model calibration approach reduces the number of high-fidelity function calls by 31% compared with the conventional first-order consistent trust region method. We note that when the geometry of the hook is physically infeasible, it is possible that the stiffness matrix is nearly singular and in these cases the gradient computed using the adjoint approach is inaccurate. Accordingly, our strategy is to not use the gradient for calibration at previously visited designs sites where the geometry was infeasible.

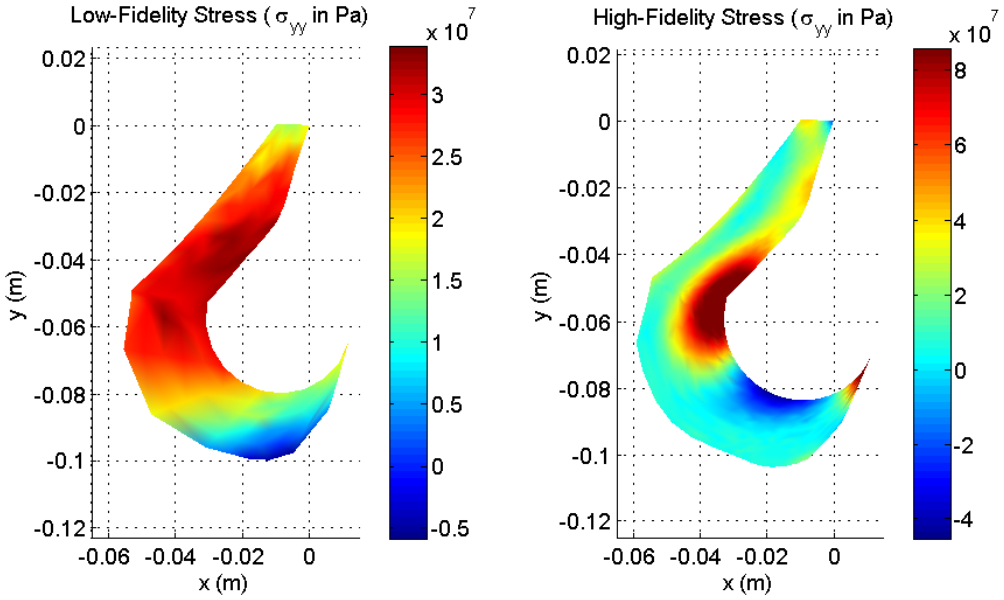


Figure 4-3: Comparison of the stress estimated by the high- and low-fidelity structural models shown on the deformed hook, the deformation is scaled by a factor of 123.5. The deflection at the midpoint of the bearing load application estimated by the two models differs by 31.6%.

To analyze the total computational effort required to solve this structural design problem using a multifidelity method in lieu of a single-fidelity scheme, we consider the total effort of the optimization. As a reference, a general purpose structural finite element solver, Nastran, uses a conjugate gradient method with an incomplete Cholesky factorization preconditioner to solve the linear finite-element system. So the effort for a forward solve scales with $\mathcal{O}(\text{dof}^2)$ [110]. Using this scaling, the forward solution for the high-fidelity structural model is about 41 times as costly as the forward solution for the low-fidelity structural model. In addition, a conservative estimate on the number of low-fidelity function calls used to find the optimal high-fidelity hook design is approximately $3,200$. The computational effort for these low-fidelity solves is equivalent to about 80 high-fidelity forward solves. Therefore the total effort of the multifidelity approach is equivalent to approximately 110 high-fidelity forward solves, and for comparison the single-fidelity SQP optimization required 232 high-fidelity evaluations. Therefore, excluding the cost of constructing the surrogate models the multifidelity approaches correspond to about a 50% decrease in computational effort compared with a single-fidelity approach.

4.3 Aerodynamic Design Problem

Aerodynamic design is a computationally expensive process because high-fidelity computational fluid dynamics (CFD) must repeatedly solve nonlinear governing equations for a large number of degrees of freedom. This section presents an adjoint-based formulation for an aerodynamic design problem described by a CFD model and then presents results of a multifidelity supersonic airfoil design problem.

To minimize the drag of a supersonic airfoil we formulate an optimization problem in the form given in Eq. 4.1. $\mathcal{J}_{\text{high}}(\mathbf{x}, \mathbf{u})$ is the surface integral of pressure acting in the flow direction, $\mathcal{R}_{\text{high}}(\mathbf{x}, \mathbf{u}) = 0$ represents the discretized Euler equations, and $\mathbf{g}(\mathbf{x}) \leq 0$ comprises two constraints on the airfoil geometry. The state variables $\mathbf{u}$ are the primal flow variables for all of the control volumes in a finite volume discretization of the governing equations. In order to use the multifidelity optimization technique presented, we need to compute the

gradient of $\mathcal{J}_{\text{high}}(\mathbf{x}, \mathbf{u})$ with respect to the design variables $\mathbf{x}$ given that $\mathcal{R}_{\text{high}}(\mathbf{x}, \mathbf{u}) = 0$. We start by writing the gradient of $\mathcal{J}_{\text{high}}(\mathbf{x}, \mathbf{u})$,

$$\frac{d\mathcal{J}_{\text{high}}}{d\mathbf{x}} = \frac{\partial \mathcal{J}_{\text{high}}}{\partial \mathbf{x}} + \frac{\partial \mathcal{J}_{\text{high}}}{\partial \mathbf{M}} \frac{\partial \mathbf{M}}{\partial \mathbf{x}} + \frac{\partial \mathcal{J}_{\text{high}}}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial \mathbf{x}}, \tag{4.31}$$

where $\mathbf{M}(\mathbf{x})$ represents the dependence of each nodal vertex in the volume mesh on the design vector. For this analysis, a meshing tool was developed that created a volume mesh around the airfoil with an analytical derivative. Accordingly, for any airfoil that could be generated with the parameterization used, the volume mesh and mesh derivative were known a priori.

Since the flow residual must always be zero for a converged solution we know that,

$$\frac{d\mathcal{R}_{\text{high}}}{d\mathbf{x}} = \frac{\partial \mathcal{R}_{\text{high}}}{\partial \mathbf{x}} + \frac{\partial \mathcal{R}_{\text{high}}}{\partial \mathbf{M}} \frac{\partial \mathbf{M}}{\partial \mathbf{x}} + \frac{\partial \mathcal{R}_{\text{high}}}{\partial \mathbf{u}} \frac{\partial \mathbf{u}}{\partial \mathbf{x}} = 0. \tag{4.32}$$

Therefore, combing Eqs. 4.31 and 4.32 with the adjoint equation, Eq. 4.25, we may write the gradient of an objective function with the state equations satisfied as

$$\frac{df_{\text{high}}}{d\mathbf{x}} = \frac{\partial \mathcal{J}_{\text{high}}}{\partial \mathbf{x}} + \frac{\partial \mathcal{J}_{\text{high}}}{\partial \mathbf{M}} \frac{\partial \mathbf{M}}{\partial \mathbf{x}} - \Psi^{\top} \left(\frac{\partial \mathcal{R}_{\text{high}}}{\partial \mathbf{x}} + \frac{\partial \mathcal{R}_{\text{high}}}{\partial \mathbf{M}} \frac{\partial \mathbf{M}}{\partial \mathbf{x}} \right). \tag{4.33}$$

This reformulation shows how to convert the constrained objective function into an objective function from which the state variables have been eliminated through the solution of the residual equations. Accordingly, the gradient $\frac{df_{\text{high}}}{d\mathbf{x}}$ represents the gradient of drag with respect to the design variables given that the discretized Euler equations are satisfied. For further discussion see Jameson [49] or Nemec *et al.* [82]. The computational effort required to compute this gradient requires one flow solution, one adjoint solution, one flow iteration per design variable (about 1/500 the effort of a flow solution), and the matrix multiplications shown above. Therefore, the gradient of the objective with respect to all of the design variables requires the computational effort of about 2 flow solutions plus n/500 flow solutions, where n is the number of design variables. For comparison, a finite difference gradient estimate requires at least $n+1$ flow solutions, so this is a considerable savings and shows the

cost of the gradient estimate using an adjoint solution is almost independent of the number of design variables.

In addition to the Euler equations as the high-fidelity method, we use a panel method as a low-fidelity analysis. A supersonic panel method can be derived from supersonic small-disturbance theory, and is only a function of the airfoil geometry, the freestream Mach number, M_∞, and the gas specific heat ratio. In small disturbance theory, the change in the pressure coefficient, δC_p is proportional, to the flow turning angle $\delta\vartheta$,

$$\delta C_p = \frac{-2\delta\vartheta}{\sqrt{M_\infty^2 - 1}}, \tag{4.34}$$

and by integrating the pressure coefficient around the airfoil surface the wave drag can be easily estimated [66]. Since the drag coefficient is only a function of the freestream Mach number and airfoil geometry, the analytical derivative of drag coefficient with respect to the airfoil shape design parameters is easy to compute.

The airfoil optimization problem is to minimize the drag of an airfoil at $M_\infty = 1.5$, by changing the angle of attack, five upper surface spline points, and five lower surface spline points. The airfoil is required to have positive thickness everywhere, and to have a maximum thickness to chord ratio that is at least five percent. Figure 4-4 shows the optimal airfoil and spline control points for the panel method. Figure 4-5 shows the optimal airfoil pressure contours and adjoint solution for the streamwise momentum from the Euler method solutions.

Table 4.3 presents the number of high-fidelity function calls to find the minimum drag airfoil with respect to the Euler code. The results show that the conventional first-order consistent trust-region and the Bayesian model calibration approach reduce the number of high-fidelity function calls by nearly the same amount, about 80%. In addition, because the low-fidelity model is computationally very inexpensive, and the dimension of the parameter space is small, this is nearly a 70% reduction in wall-clock time. However, it should be noted that the conventional trust-region approach does use on average fewer high-fidelity function calls than the Bayesian calibration method for the ten random initial airfoils.

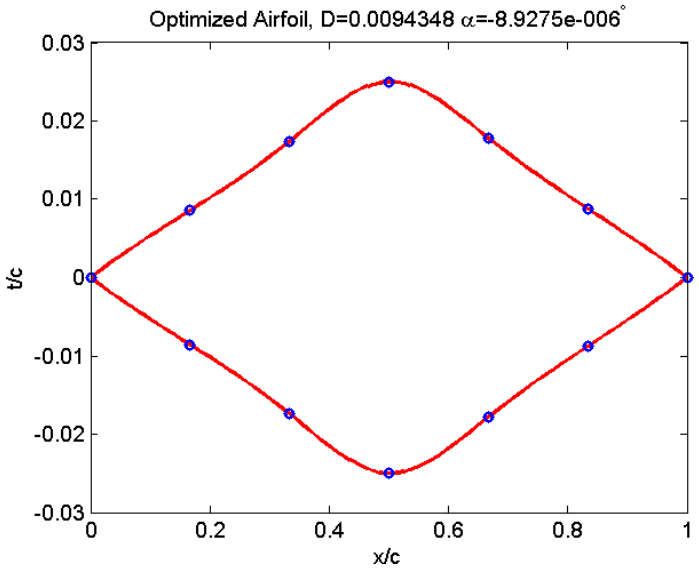


Figure 4-4: Minimum drag airfoil computed with the supersonic panel method showing the spline control points.

	SQP	First-Order TR	Calibration
Mean	81.5 (-)	12.9 (-84%)	14.7 (-82%)
Std. Dev.	14.0	2.86	4.19

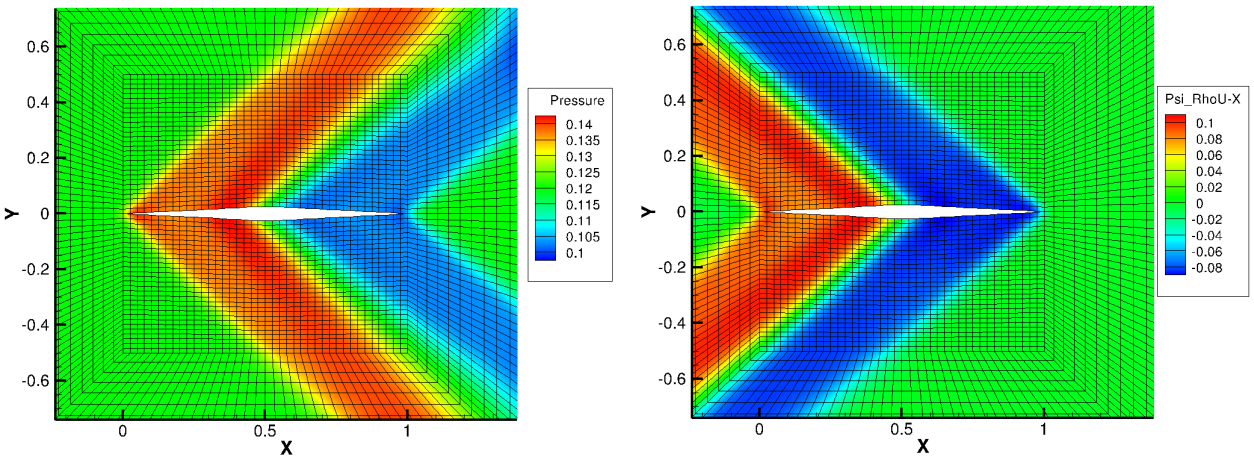
Table 4.3: The average number of high-fidelity function evaluations to minimize the drag of a supersonic airfoil with respect to an Euler solution using a panel method as a low-fidelity estimate. The numbers in parentheses indicate the percentage reduction in high-fidelity function evaluations relative to SQP.

The same optimization problem is solved using the same methods, but without any low-fidelity information. The results of this optimization using $f_{\text{low}}(\mathbf{x}) = 0$ as the low-fidelity function are presented in Table 4.4. In this case the Bayesian model calibration approach performed noticeably better than the conventional trust-region method. The results suggest that when using a "good" low-fidelity model, the Bayesian calibration approach is not necessary and the computational effort of constructing the Cokriging surrogates is not worthwhile. However, when the low-fidelity function is poor, or when the error between the high- and low-fidelity function behaves in a highly non-linear fashion, then the Bayesian calibration approach may provide a computational savings for low-dimensional optimization problems. A possible performance issue that affects Bayesian calibration approach significantly more than the conventional trust-region approach is when high-fidelity gradients are computed in-

	SQP	First-Order TR	Calibration
Mean	81.5 (-)	91.1 (+12%)	64.2 (-21%)
Std. Dev.	14.0	61.2	21.8

Table 4.4: The average number of high-fidelity function evaluations to minimize the drag of a supersonic airfoil with respect to an Euler solution. No low-fidelity information is used. The numbers in parentheses indicate the percentage reduction in high-fidelity function evaluations relative to SQP.

accurately. The calibration approach reuses high-fidelity gradient information, so incorrect gradient information will propagate into the surrogate models generated at future trust-region iterations. In contrast, the conventional approach only uses the high-fidelity gradient to create a single surrogate model so the approach is likely less affected by inaccurate gradient information.



(a) Optimal airfoil pressure solution.

(b) Optimal airfoil adjoint solution.

Figure 4-5: Minimum drag supersonic airfoil parameterized by 5 upper surface spline points, 5 lower surface spline points, and angle of attack.

4.4 Summary

This chapter has presented a multifidelity optimization algorithm using Cokriging-based Bayesian model calibration that is provably convergent to an optimum of the original high-fidelity optimization problem. Inexpensive derivative information was obtained through

adjoint solutions for both structural design and aerodynamic shape optimization problems. The derivative estimates and function values from previously visited design sites are used to calibrate the low-fidelity model to the high-fidelity function. Using this strategy, the optimization algorithm is likely to quickly find an optimum of the high-fidelity function. The sample results show that for poor low-fidelity models the Bayesian calibration exceeds performance of conventional trust-region algorithms; however, for cases with good low-fidelity models the performance of the two algorithms is similar. However, the multifidelity approaches still significantly outperform single-fidelity sequential quadratic programming methods. The calibration technique developed is recommended for low-dimensional optimization problems where the quality of the low-fidelity model is unknown or known to be poor in certain portions of the design space.

This algorithm complements the work in Chapters 2 and 3, in that although a first-order consistent calibration strategy satisfies the requirement for a fully linear calibration, an optimization algorithm based on a fully linear calibration does not take advantage of gradient information. The algorithm in this chapter attempts to truly exploit gradient information to find the optimal design as quickly as possible by efficiently using available gradient information. This strategy still supports the use of multiple lower-fidelity models through the filtering technique presented in Section 2.5, and both fully linear and first-order consistent surrogates can be combined with that technique.

Chapter 5

Multidisciplinary and Multifidelity Optimization

Chapters 2, 3, and 4 presented techniques for multifidelity optimization and demonstrated how to accelerate finding optimal designs using computationally expensive simulations. This chapter considers the case of designing a system with multiple interacting disciplines or subsystems, and focuses on the case where each is modeled with an expensive high-fidelity simulation. The challenge of optimizing multidisciplinary systems is two-fold. First, it is necessary to resolve the coupling between all disciplines in order to find a feasible design, one where the inputs and outputs from communicating disciplines match, and second, it is necessary to analyze the trade-offs among all disciplines in order to find an optimal design. For systems where each discipline analysis is a high-fidelity simulation, multifidelity optimization alone may be insufficient to make designing an optimal system tractable. Therefore, this chapter develops multidisciplinary optimization methods that enable multifidelity optimization at the discipline level and that enable the design and analysis of each subsystem to be conducted in parallel.

This chapter will first formulate a multidisciplinary optimization problem in Section 5.1. It then presents two multidisciplinary and multifidelity optimization methods. The first method uses an Individual Discipline Feasible (IDF) approach, which decomposes the system optimization problem into a collection of smaller discipline-level optimization problems that

may be conducted in parallel. The IDF approach is presented in Section 5.2. The second MDO optimization method is based on an All-At-Once (AAO) formulation, though it is in actuality just a gradient-free method for solving equality constrained optimization problems. The AAO approach is presented in Section 5.3. The two approaches are compared with some baseline methods on two analytical test problems in Section 5.4. This chapter concludes with Section 5.5 which contains a discussion about these methods, their applications, and possible extensions.

5.1 Problem Formulation

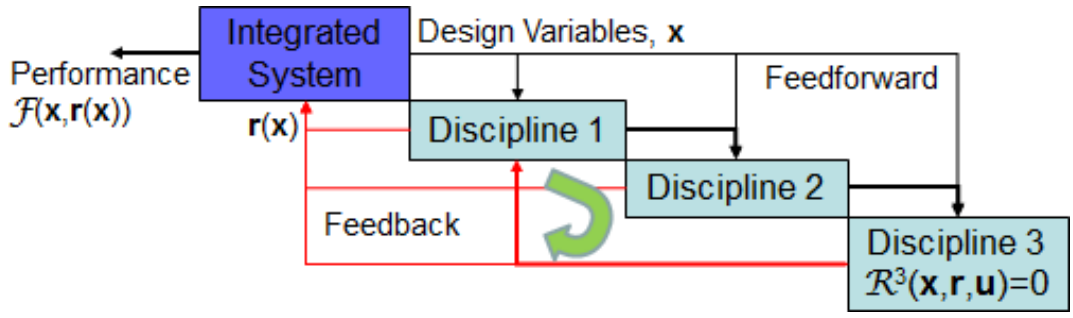
We consider a system design problem comprising m disciplines or subystems. Each discipline produces a response that is a function of both the system design and the output of the other disciplines. The response of a discipline may be thought of as the coupling between the disciplines. For example, the i^{th} discipline has a response, $\mathbf{r}^i = \mathbf{r}^i(\mathbf{x}, \mathbf{r}^{1,\dots,m\setminus i}, \mathbf{u}^i(\mathbf{x}, \mathbf{r}^{1,\dots,m\setminus i}))$, that is a function of the n-dimensional system design vector, $\mathbf{x}$, the responses of all other disciplines, $\mathbf{r}^{1,\dots,m\setminus i}$, and the discipline state variables, $\mathbf{u}^i(\mathbf{x}, \mathbf{r}^{1,\dots,m\setminus i})$. The vector $\mathbf{r}$ without superscript is the vector of all $\mathbf{r}^i$, $\mathbf{r} = [(\mathbf{r}^1)^\top, (\mathbf{r}^2)^\top, \dots, (\mathbf{r}^m)^\top]^\top$, and has length v. To find a multidisciplinary feasible design, an iterative solution scheme is typically required to find the set of state variables and discipline responses that enable a feasible coupling among all disciplines. This iterative solve is typically called a multidisciplinary analysis (MDA).

When optimizing a multidisciplinary system using a multidisciplinary feasible (MDF) formulation, the first step in evaluating the performance of a given design vector, $\mathbf{x}$, is the MDA, to calculate the state and coupling variables that satisfy the multidisciplinary feasibility [34, 111]. In this case, we may write an MDF optimization problem as

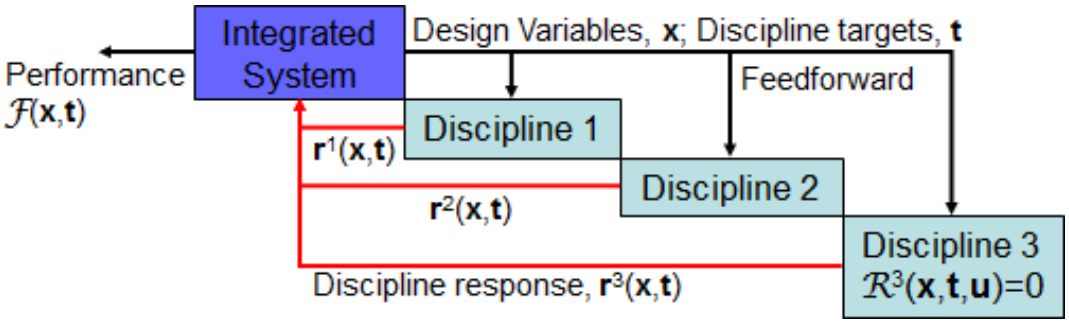
$$\min_{\mathbf{x}\in\mathbb{R}^n} \quad \mathcal{F}(\mathbf{x}, \mathbf{r}(\mathbf{x})) \tag{5.1a}$$

$$\text{s.t. } \mathcal{G}(\mathbf{x}, \mathbf{r}(\mathbf{x})) \leq 0 \tag{5.1b}$$

$$\mathcal{H}(\mathbf{x}, \mathbf{r}(\mathbf{x})) = 0, \tag{5.1c}$$



(a) Block diagram for an MDF framework.

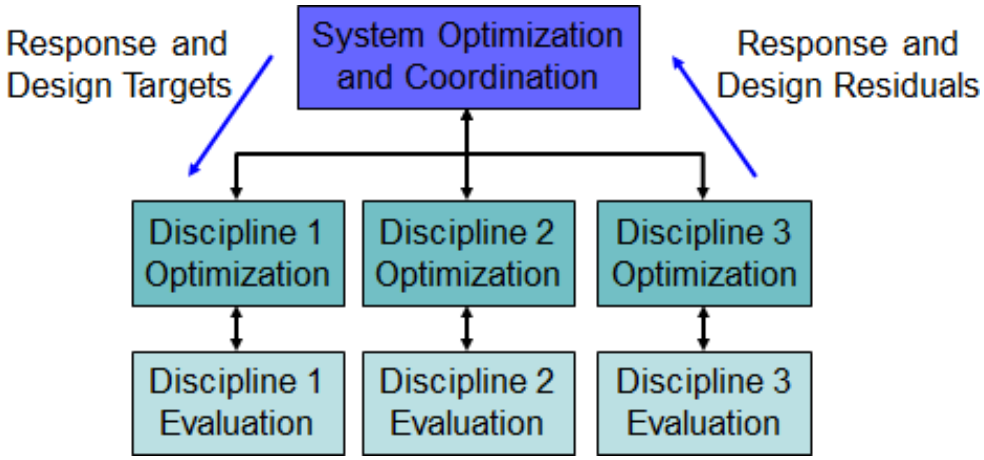


(b) Block diagram for a decoupled framework like IDF.

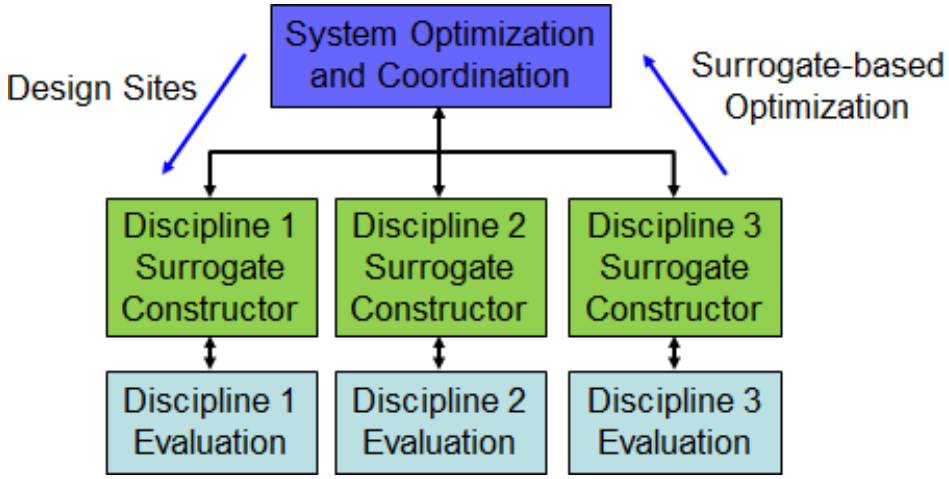
Figure 5-1: Block diagram for a system with, (a), and without, (b), interdisciplinary communication.

where we are minimizing a scalar objective function, $\mathcal{F}(\mathbf{x}, \mathbf{r}(\mathbf{x}))$, subject to inequality constraints, $\mathcal{G}(\mathbf{x}, \mathbf{r}(\mathbf{x})) \leq 0$, and equality constraints, $\mathcal{H}(\mathbf{x}, \mathbf{r}(\mathbf{x})) = 0$. In the MDF formulation we have omitted the dependence of the discipline responses on the state variables and other discipline responses since they are satisfied by construction (i.e., each discipline performs an internal iteration to satisfy its state constraints, $\mathcal{R}^i(\mathbf{x}, \mathbf{r}^{1,\ldots,m\setminus i}, \mathbf{u}^i(\mathbf{x}, \mathbf{r}^{1,\ldots,m\setminus i})) = 0$). This formulation is presented schematically as Figure 5-1(a). Each time the system performance, $\mathcal{F}(\mathbf{x}, \mathbf{r}(\mathbf{x}))$, is evaluated an iterative solution process amongst the disciplines has been undertaken to satisfy feasibility. Some challenges with a MDF framework are that if finite-differences are necessary to generate sensitivity information, then an iterative solve is necessary for each perturbed component of $\mathbf{x}$. In addition, an MDF formulation only supports multifidelity optimization and parallelization at the system level, evaluating performance of multiple complete system designs at the same time. To parallelize the system optimization we develop two alternative formulations of (5.1). The first decomposes the large system optimization problem into a collection of smaller subsystem optimization problems,

as shown in Figure 5-2(a). This framework is presented in Section 5.2. The other evaluates all computationally expensive functions in parallel, as shown in Figure 5-2(b). This framework is presented in Section 5.3.



(a) Parallel discipline optimizations.



(b) Parallel discipline evaluations.

Figure 5-2: Block diagram for optimizing a system with parallel discipline optimizations, (a), and with parallel discipline evaluations, (b). It may also be possible in both frameworks to evaluate multiple design sites of the same discipline in parallel during the discipline optimizations or during the discipline surrogate creation.

5.2 Parallel Discipline Optimizations

This section presents a method to solve a large system optimization problem by decomposing it into a collection of smaller optimization problems that can be solved in parallel. The framework develops a novel IDF formulation for (5.1), which is different from the formulations proposed by Cramer *et al.* [34] in that it is a bilevel programming problem. Standard bilevel programming solution methods, for example see [11, 25, 26, 29, 35], can be used to solve the optimization problem. However, solving bilevel programming problems is typically quite difficult, so although the discipline-level optimizations are fully parallelizable, finding solutions in the general case maybe nearly impossible. See [11, 35] for discussion, but some key difficulties in the general nonconvex case are that each subproblem may have multiple solutions, adding inactive constraints may change the solutions, and inequality constraints make finding solutions a combinatorial problem. Therefore, we develop an approach based on the Alternating Direction Method of Multipliers (ADMM) which can be viewed as a Gauss-Seidel iteration over the design variables, see [14, 17, 43, 46]. The bilevel IDF formulation is presented in Section 5.2.1, the ADMM method to solve the IDF formulation is presented in Section 5.2.2, and supporting theory for the method is presented in Section 5.2.3. A summary of the algorithm is presented as Algorithm 5.1.

5.2.1 Parallel Individual Discipline Feasible Formulation

To solve (5.1) using parallel discipline optimizations we relax the requirement that the system is multidisciplinary feasible. This means the input/output mapping between disciplines/subsystems does not have to be satisfied until termination of the optimization. This is now an IDF formulation. The formulation is achieved by creating a vector of discipline targets, $\mathbf{t} = [(\mathbf{t}^1)^\top, (\mathbf{t}^2)^\top, \ldots, (\mathbf{t}^m)^\top]^\top$, where $\mathbf{t}^i$ has the same dimension as $\mathbf{r}^i$, and making the discipline responses a function of the design variables and targets as opposed to the other discipline responses. In an IDF formulation, the discipline state equations and all other constraints that are contained within a discipline are satisfied locally by solving the discipline state equations. Only the coupling between the disciplines is relaxed during the

optimization. An IDF formulation is shown schematically in Figure 5-1(b). The advantage of the IDF formulation over the MDF formulation is the elimination of the iterative solve to resolve coupling among disciplines.

In this thesis, we develop a new IDF formulation to enable parallelized design. We introduce m vectors, $\tilde{\mathbf{x}}^i$, $i = 1, \ldots, m$, which are discipline-level copies of the system design vector, $\mathbf{x}$. We define each discipline response, $\mathbf{r}^i(\tilde{\mathbf{x}}^i, \mathbf{t}^{1,\ldots,m\setminus i})$, to be only a function of its own copy of the system design vector and target variables for the other disciplines (we have again omitted the dependence of the discipline responses on the state variables as they are satisfied by construction). Thus, the discipline responses may be optimized independently of all the other disciplines. The form of the IDF problem used in this thesis is,

$$\min_{\mathbf{x}\in\mathbb{R}^n, \mathbf{t}\in\mathbb{R}^v} \quad \mathcal{F}(\mathbf{x}, \mathbf{t}) \tag{5.2a}$$

$$\text{s.t. } \mathcal{G}(\mathbf{x}, \mathbf{t}) \leq 0 \tag{5.2b}$$

$$\mathcal{H}(\mathbf{x}, \mathbf{t}) = 0 \tag{5.2c}$$

$$\mathbf{r}^i(\tilde{\mathbf{x}}^i, \mathbf{t}^{1,\ldots,m\setminus i}) = \mathbf{t}^i, \qquad i = 1, \ldots, m \tag{5.2d}$$

$$\mathbf{x} = \tilde{\mathbf{x}}^i, \qquad i = 1, \ldots, m \tag{5.2e}$$

where

$$\tilde{\mathbf{x}}^i = \arg\min_{\tilde{\mathbf{x}}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{r}^i(\tilde{\mathbf{x}}, \mathbf{t}^{1,\ldots,m\setminus i}) - \mathbf{t}^i\|_2^2, \qquad i = 1, \ldots, m, \tag{5.2f}$$

which is different from conventional IDF formulations [34] in that it is a bilevel programming problem. The subsystem optimizations over $\tilde{\mathbf{x}}^i$, (5.2f), occur in parallel, and the constraints that the discipline outputs equal their targets, (5.2d), and that all of the design vector copies are equal to the system design vector, (5.2e), are at the system level. The system-level constraints are likely difficult to satisfy, so Section 5.2.2 offers a highly parallelizable solution technique for (5.2) based on the ADMM, shown in Figure 5-2(a).

The scheme uses an augmented Lagrangian defined as,

$$\mathfrak{L}(\mathbf{x}, \mathbf{t}, \tilde{\mathbf{x}}, \boldsymbol{\nu}, \boldsymbol{\lambda}, \boldsymbol{\mu}, \tilde{\boldsymbol{\mu}}) = \mathcal{F}(\mathbf{x}, \mathbf{t}) + \sum_{i}^{m} \left[(\boldsymbol{\lambda}^i)^\top \left(\mathbf{r}^i(\tilde{\mathbf{x}}^i, \mathbf{t}) - \mathbf{t}^i \right) + \frac{\mu^{(i)}}{2} \left\| \mathbf{r}^i(\tilde{\mathbf{x}}^i, \mathbf{t}) - \mathbf{t}^i \right\|^2 + (\boldsymbol{\nu}^i)^\top \left(\mathbf{x} - \tilde{\mathbf{x}}^i \right) + \frac{\tilde{\mu}^{(i)}}{2} \|\mathbf{x} - \tilde{\mathbf{x}}^i\|_2^2 \right], \tag{5.3}$$

where $\boldsymbol{\lambda}$, $\boldsymbol{\nu}$ are Lagrange multipliers and $\boldsymbol{\mu}, \tilde{\boldsymbol{\mu}} \in \mathbb{R}^m$ are penalty parameters. The vector, $\boldsymbol{\lambda} = [(\boldsymbol{\lambda}^1)^\top, (\boldsymbol{\lambda}^2)^\top, \ldots, (\boldsymbol{\lambda}^m)^\top]^\top$, where each $\boldsymbol{\lambda}^i$ has one element for each response of discipline i. The vector, $\boldsymbol{\nu} = [(\boldsymbol{\nu}^1)^\top, (\boldsymbol{\nu}^2)^\top, \ldots, (\boldsymbol{\nu}^m)^\top]^\top$, where each $\boldsymbol{\nu}^i$ has length n. The constraints, $\mathcal{G}(\mathbf{x}, \mathbf{t}) \leq 0$ and $\mathcal{H}(\mathbf{x}, \mathbf{t}) = 0$, are not included in the augmented Lagrangian as they are handled explicitly in the formulation. The ADMM updates the design variables in alternating directions, i.e., it updates $\mathbf{x}$, then $\tilde{\mathbf{x}}^i$, then $\mathbf{t}$. After all the updates have been computed the Lagrange multiplier estimates and penalty parameters are updated. This procedure is iterated until the variables become stationary, when a solution to Eq. 5.2 is found. The algorithm is presented in Section 5.2.2, and a theoretical foundation is presented in Section 5.2.3.

5.2.2 IDF Algorithm

At an iteration k with given variables $\mathbf{x}_k, \mathbf{t}_k, \tilde{\mathbf{x}}_k^i$, Lagrange multiplier estimates $\boldsymbol{\lambda}_k, \boldsymbol{\nu}_k$, and penalty parameters, $\boldsymbol{\mu}_k, \tilde{\boldsymbol{\mu}}_k$, we first minimize the augmented Lagrangian over the design variables, $\mathbf{x}$. This optimization does not include any subsystem performance estimates; all communication between the subsystem analyses comes through the target variables, design variable copies, and Lagrange multipliers. The optimization problem is,

$$\begin{aligned} \mathbf{x}_{k+1} = \arg\min_{\mathbf{x} \in \mathbb{R}^n} \; & \mathcal{F}(\mathbf{x}, \mathbf{t}_k) + \sum_{i}^{m} \left[(\boldsymbol{\nu}_k^i)^\top \left(\mathbf{x} - \tilde{\mathbf{x}}_k^i \right) + \frac{\tilde{\mu}_k^{(i)}}{2} \|\mathbf{x} - \tilde{\mathbf{x}}_k^i\|_2^2 \right] \\ \text{s.t. } & \mathcal{G}(\mathbf{x}, \mathbf{t}_k) \leq 0 \\ & \mathcal{H}(\mathbf{x}, \mathbf{t}_k) = 0, \end{aligned} \tag{5.4}$$

where the system level constraints, $\mathcal{G}(\mathbf{x}, \mathbf{t}_k) \leq 0, \mathcal{H}(\mathbf{x}, \mathbf{t}_k) = 0$, are enforced during this minimization. The second step is to update the copies of the design vector $\mathbf{x}$ that are used within the disciplines, $\tilde{\mathbf{x}}^i$. This is accomplished by minimizing the augmented Lagrangian with the updated values of the design variables, $\mathbf{x}_{k+1}$, incorporated,

$$\tilde{\mathbf{x}}^i_{k+1} = \arg\min_{\tilde{\mathbf{x}}^i \in \mathbb{R}^n} (\boldsymbol{\lambda}^i_k)^\top \left(\mathbf{r}^i(\tilde{\mathbf{x}}^i, \mathbf{t}_k) - \mathbf{t}^i_k\right) + \frac{\mu^{(i)}_k}{2} \left\|\mathbf{r}^i(\tilde{\mathbf{x}}^i, \mathbf{t}_k) - \mathbf{t}^i_k\right\|^2 + (\boldsymbol{\nu}^i_k)^\top \left(\mathbf{x}_{k+1} - \tilde{\mathbf{x}}^i\right) + \frac{\tilde{\mu}^{(i)}_k}{2} \|\mathbf{x}_{k+1} - \tilde{\mathbf{x}}^i\|_2^2. \quad (5.5)$$

The minimization for each $\tilde{\mathbf{x}}^i$ can be solved concurrently. Each discipline is completely decoupled from the other disciplines; the discipline objective is a weighted combination of how close the discipline-level performance is to the target and how close the discipline-level design is to the system-level design. Upon completion of the local design variable updates, the subsystem responses, $\mathbf{r}^i(\tilde{\mathbf{x}}^i_{k+1}, \mathbf{t}_k)$, are evaluated.

With updated system and subsystem designs, the discipline-level performance targets are all updated. This is again accomplished by minimizing the augmented Lagrangian with the updated system-level design and updated subsystem-level designs subject to the system-level constraints,

$$\begin{aligned} \mathbf{t}_{k+1} = \underset{\mathbf{t} \in \mathbb{R}^v}{\operatorname{argmin}} \; & \mathcal{F}(\mathbf{x}_{k+1}, \mathbf{t}) + \sum_i^m \left[(\boldsymbol{\lambda}^i_k)^\top \left(\mathbf{r}^i(\tilde{\mathbf{x}}^i_{k+1}, \mathbf{t}) - \mathbf{t}^i\right) + \frac{\mu^{(i)}_k}{2} \left\|\mathbf{r}^i(\tilde{\mathbf{x}}^i_{k+1}, \mathbf{t}) - \mathbf{t}^i\right\|^2 \right] \quad (5.6) \\ \text{s.t. } & \mathcal{G}(\mathbf{x}_{k+1}, \mathbf{t}) \leq 0 \\ & \mathcal{H}(\mathbf{x}_{k+1}, \mathbf{t}) = 0. \end{aligned}$$

An important point is that the subsystem response, though a function of the targets, is not updated during this optimization. The Lagrange multiplier estimates are then updated with the standard first-order scheme,

$$\boldsymbol{\lambda}^i_{k+1} = \boldsymbol{\lambda}^i_k + \mu^{(i)}_k \left(\mathbf{r}^i(\tilde{\mathbf{x}}^i_{k+1}, \mathbf{t}_k) - \mathbf{t}^i_{k+1}\right), \quad (5.7)$$

$$\boldsymbol{\nu}^i_{k+1} = \boldsymbol{\nu}^i_k + \tilde{\mu}^{(i)}_k (\mathbf{x}_{k+1} - \tilde{\mathbf{x}}^i_{k+1}). \quad (5.8)$$

To update the penalty parameters, a strategy discussed in [12] is used. The target-response penalty parameters are updated by,

$$\mu_{k+1}^{(i)} = \begin{cases} \gamma\mu_k^{(i)} & \text{if} \|\mathbf{r}^i(\tilde{\mathbf{x}}_{k+1}^i, \mathbf{t}_k) - \mathbf{t}_{k+1}^i\| > \eta \|\mathbf{r}^i(\tilde{\mathbf{x}}_k^i, \mathbf{t}_{k-1}) - \mathbf{t}_k^i\| \\ \mu_k^{(i)} & \text{otherwise} \end{cases} \tag{5.9}$$

and the system-subsystem design penalty parameters are updated by,

$$\tilde{\mu}_{k+1}^{(i)} = \begin{cases} \gamma\tilde{\mu}_k^{(i)} & \text{if} \|\mathbf{x}_{k+1} - \tilde{\mathbf{x}}_{k+1}^i\| > \eta \|\mathbf{x}_k - \tilde{\mathbf{x}}_k^i\| \\ \tilde{\mu}_k^{(i)} & \text{otherwise,} \end{cases} \tag{5.10}$$

where $\eta \in (0, 1)$, $\gamma > 1$ are parameters with suggested values $\eta = 0.75$, $\gamma = 5$. An implementation of the algorithm is presented as Algorithm 5.1.

This bilevel programming formulation offers an advantage over many other multidisciplinary optimization methods, since each discipline can be optimized using any appropriate optimization technique. This enables exploitation of aspects of the individual disciplines that may speed the discipline-level optimizations, for example, the ability to use gradient-based optimization, gradient-free optimization or multifidelity optimization.

5.2.3 Parallel Discipline Optimizations Supporting Theory

In practice Algorithm 5.1 has been observed to converge reasonably quickly to an optimal multidisciplinary system design. This section provides theory to support the observed behavior of the algorithm. It also discusses the restrictive assumptions necessary to analyze the convergence behavior and application of Algorithm 5.1 to a broader class of optimization problems where the assumptions do not hold. There are two mechanisms by which Algorithm 5.1 solves optimization problems, (1) asymptotically exact minimization in the method of multipliers, and (2) the quadratic penalty function. The reason for the two mechanisms is the penalty updating scheme, which ensures either (a) the constraint violation goes to zero and the penalty parameters remain finite or (b) the penalty parameter grows without bound.

Algorithm 5.1: Parallel Individual Discipline Feasible Algorithm

0: Choose, $\mathbf{x}_0, \mathbf{t}_0, \tilde{\mathbf{x}}_0^i$, and Lagrange multiplier estimates $\boldsymbol{\lambda}_0, \boldsymbol{\nu}_0$. Choose initial penalty parameters, $\boldsymbol{\mu}, \tilde{\boldsymbol{\mu}} \in \mathbb{R}^m > 0$, penalty update criterion, $\eta \in (0,1)$, and penalty growth factor, $\gamma > 1$. (Recommended values are given in Table 5.1.)

1: Update the system-level design variables by solving,

$$\begin{aligned}\mathbf{x}_{k+1} = \underset{\mathbf{x}\in\mathbb{R}^n}{\arg\min}\, & \mathcal{F}(\mathbf{x}, \mathbf{t}_k) + \sum_i^m \left[(\boldsymbol{\nu}_k^i)^\top \left(\mathbf{x} - \tilde{\mathbf{x}}_k^i\right) + \frac{\tilde{\mu}_k^{(i)}}{2} \|\mathbf{x} - \tilde{\mathbf{x}}_k^i\|_2^2 \right] \\ \text{s.t. } & \mathcal{G}(\mathbf{x}, \mathbf{t}_k) \le 0 \\ & \mathcal{H}(\mathbf{x}, \mathbf{t}_k) = 0.\end{aligned}$$

2: Update the discipline-level design variables copies by solving,

$$\begin{aligned}\tilde{\mathbf{x}}_{k+1}^i = \arg\min_{\tilde{\mathbf{x}}^i\in\mathbb{R}^n} & (\boldsymbol{\lambda}_k^i)^\top \left(\mathbf{r}^i(\tilde{\mathbf{x}}^i, \mathbf{t}_k) - \mathbf{t}_k^i\right) + \frac{\mu_k^{(i)}}{2} \left\|\mathbf{r}^i(\tilde{\mathbf{x}}^i, \mathbf{t}_k) - \mathbf{t}_k^i\right\|^2 + \\ & (\boldsymbol{\nu}_k^i)^\top \left(\mathbf{x}_{k+1} - \tilde{\mathbf{x}}^i\right) + \frac{\tilde{\mu}_k^{(i)}}{2} \|\mathbf{x}_{k+1} - \tilde{\mathbf{x}}^i\|_2^2.\end{aligned}$$

2a: Evaluate $\mathbf{r}^i(\tilde{\mathbf{x}}_{k+1}^i, \mathbf{t}_k)$.

3: Update the system-level discipline targets by solving,

$$\begin{aligned}\mathbf{t}_{k+1} = \underset{\mathbf{t}\in\mathbb{R}^v}{\operatorname{argmin}}\, & \mathcal{F}(\mathbf{x}_{k+1}, \mathbf{t}) + \sum_i^m \left[(\boldsymbol{\lambda}_k^i)^\top \left(\mathbf{r}^i(\tilde{\mathbf{x}}_{k+1}^i, \mathbf{t}) - \mathbf{t}^i\right) + \frac{\mu_k^{(i)}}{2} \left\|\mathbf{r}^i(\tilde{\mathbf{x}}_{k+1}^i, \mathbf{t}) - \mathbf{t}^i\right\|^2 \right] \\ \text{s.t. } & \mathcal{G}(\mathbf{x}_{k+1}, \mathbf{t}) \le 0 \\ & \mathcal{H}(\mathbf{x}_{k+1}, \mathbf{t}) = 0.\end{aligned}$$

4: Update the Lagrange multiplier estimates by evaluating,

$$\begin{aligned}\boldsymbol{\lambda}_{k+1}^i &= \boldsymbol{\lambda}_k^i + \mu_k^{(i)} \left(\mathbf{r}^i(\tilde{\mathbf{x}}_{k+1}^i, \mathbf{t}_k) - \mathbf{t}_{k+1}^i\right), \\ \boldsymbol{\nu}_{k+1}^i &= \boldsymbol{\nu}_k^i + \tilde{\mu}_k^{(i)} (\mathbf{x}_{k+1} - \tilde{\mathbf{x}}_{k+1}^i).\end{aligned}$$

5: Update the target-response penalty parameters,

$$\mu_{k+1}^{(i)} = \begin{cases} \gamma \mu_k^{(i)} & \text{if} \|\mathbf{r}^i(\tilde{\mathbf{x}}_{k+1}^i, \mathbf{t}_k) - \mathbf{t}_{k+1}^i\| > \eta \|\mathbf{r}^i(\tilde{\mathbf{x}}_k^i, \mathbf{t}_{k-1}) - \mathbf{t}_k^i\| \\ \mu_k^{(i)} & \text{otherwise.} \end{cases}$$

6: Update the system design penalty parameters,

$$\tilde{\mu}_{k+1}^{(i)} = \begin{cases} \gamma \tilde{\mu}_k^{(i)} & \text{if} \|\mathbf{x}_{k+1} - \tilde{\mathbf{x}}_{k+1}^i\| > \eta \|\mathbf{x}_k - \tilde{\mathbf{x}}_k^i\| \\ \tilde{\mu}_k^{(i)} & \text{otherwise.} \end{cases}$$

7: Check for convergence, if all of the controllable variables are stationary, $\|\mathbf{x}_k - \mathbf{x}_{k+1}\| \le \epsilon$, $\|\tilde{\mathbf{x}}_{k+1}^i - \tilde{\mathbf{x}}_k^i\| \le \epsilon$, $\|\mathbf{t}_{k+1} - \mathbf{t}_k\| \le \epsilon$, $\|\boldsymbol{\lambda}_{k+1} - \boldsymbol{\lambda}_{k+1}\| \le \epsilon$ and $\|\boldsymbol{\nu}_{k+1} - \boldsymbol{\nu}_k\| \le \epsilon$, then the method has converged. Otherwise, increment k and go to step 1.

The former case, (a), corresponds to asymptotically exact minimization in the method of multipliers, and the latter case, (b), corresponds to a quadratic penalty function.

To simplify the analysis we treat the target and response for each discipline as a scalar and use a single scalar penalty parameter, μ. The scalar penalty parameter update is,

$$\mu_{k+1} = \begin{cases} \gamma\mu_k & \text{if} \|\mathbf{q}_{k+1}\|_2 \geq \eta \|\mathbf{q}_k\|_2 \\ \mu_k & \text{otherwise,} \end{cases} \tag{5.11}$$

where the feasibility measure

$$\mathbf{q}_k = \left[\left(\mathbf{r}(\tilde{\mathbf{x}}_k, \mathbf{t}_{k-1}) - \mathbf{t}_k\right)^\top, \|\mathbf{x}_k - \tilde{\mathbf{x}}_k^1\|, \ldots, \|\mathbf{x}_k - \tilde{\mathbf{x}}_k^m\| \right]^\top. \tag{5.12}$$

In addition, we use $\mathbf{z}^\top = [\mathbf{x}^\top, \mathbf{t}^\top]$, and $\bar{\mathbf{z}}^\top = [\mathbf{x}^\top, \mathbf{t}^\top, \tilde{\mathbf{x}}^\top]$, where $\tilde{\mathbf{x}}$ without a discipline index denotes a column vector of all the $\tilde{\mathbf{x}}^i$'s, $\tilde{\mathbf{x}} = [(\tilde{\mathbf{x}}^1)^\top, (\tilde{\mathbf{x}}^2)^\top, \ldots, (\tilde{\mathbf{x}}^m)^\top]^\top$. The assumptions used in this section are:

IDF1. All functions in Eq. 5.2 are twice continuously differentiable and have finite upper bounds for the 2-norm of their gradients and Hessians. (This condition implies that the functions have Lipschitz continuous first derivatives.)

IDF2. The algorithm starts sufficiently close to an optimal design, $\bar{\mathbf{z}}^*$. This includes the initial Lagrange multipliers, the initial design vector, and the initial subsystem designs. Further, this optimum is a regular point at which the second order optimality conditions hold strictly, and the initial penalty value is sufficiently large such that the augmented Lagrangian is convex.[1]

IDF3. Subproblem optimizations are solved exactly.

IDF4. The Lagrange multiplier estimates remain bounded for all k.

IDF5. $\nabla_{\tilde{\mathbf{x}}^i\tilde{\mathbf{x}}^i}\mathfrak{L}(\mathbf{x}_{k+1}, \mathbf{t}_k, \tilde{\mathbf{x}}^i, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k)$ is positive definite for all k. (We will discuss the implications of this assumption.)

[1]The second order optimality condition is sufficient to ensure existence of such a penalty value [83, Theorem 17.5].

Asymptotically Exact Minimization in the Method of Multipliers

Following [12, Section 2.2], there exists a sufficiently large value of μ, say $\hat{\mu}$, and an open sphere around all strict local optima of Eq. 5.2 which are regular points and satisfy the second order sufficient conditions for optimality, in which the augmented Lagrangian, Eq. 5.3, is convex [83, Theorem 17.5]. The convergence proof of asymptotically exact minimization in the method of multipliers states when started in the convex region of the augmented Lagrangian near an optimal solution, $\bar{\mathbf{z}}^*$, if the augmented Lagrangian remains convex and at each iteration the design variables are updated such that $\|\nabla_{\bar{\mathbf{z}}}\mathfrak{L}(\bar{\mathbf{z}}_{k+1}, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k)\| \leq \tau_k$, where $\lim_{k\to\infty} \tau_k = 0$, then $\lim_{k\to\infty} \bar{\mathbf{z}}_k = \bar{\mathbf{z}}^*$ [12, Section 2.5].

The essential reason the convergence proof of the asymptotically exact method of multipliers does not apply directly to our IDF algorithm is that the IDF algorithm updates the design variables along alternating directions. After each iteration of Algorithm 5.1, we have that,

$$\begin{aligned}
\nabla_{\mathbf{x}}\mathfrak{L}(\mathbf{x}_{k+1}, \mathbf{t}_k, \tilde{\mathbf{x}}^i_k, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k) &= 0 \\
\nabla_{\tilde{\mathbf{x}}^i}\mathfrak{L}(\mathbf{x}_{k+1}, \mathbf{t}_k, \tilde{\mathbf{x}}^i_{k+1}, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k) &= 0 \\
\nabla_{\mathbf{t}}\mathfrak{L}(\mathbf{x}_{k+1}, \mathbf{t}_{k+1}, \tilde{\mathbf{x}}^i_{k+1}, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k) &= 0,
\end{aligned}$$

but we desire,

$$\begin{aligned}
\nabla_{\mathbf{x}}\mathfrak{L}(\mathbf{x}_{k+1}, \mathbf{t}_{k+1}, \tilde{\mathbf{x}}^i_{k+1}, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k) &= 0 \\
\nabla_{\tilde{\mathbf{x}}^i}\mathfrak{L}(\mathbf{x}_{k+1}, \mathbf{t}_{k+1}, \tilde{\mathbf{x}}^i_{k+1}, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k) &= 0 \\
\nabla_{\mathbf{t}}\mathfrak{L}(\mathbf{x}_{k+1}, \mathbf{t}_{k+1}, \tilde{\mathbf{x}}^i_{k+1}, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k) &= 0.
\end{aligned}$$

However, in case (a) where $\lim_{k\to\infty} \|\mathbf{q}_k\| = 0$ and $\lim_{k\to\infty} \mu_k \in (0, \infty)$, the mean value theorem and IDF1 tell us there exist positive constants, $\kappa_{\tilde{x}}$ and κ_t, such that $\|\nabla_{\bar{\mathbf{z}}}\mathfrak{L}(\bar{\mathbf{z}}_{k+1}, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k)\| \leq \kappa_{\tilde{x}}\|\tilde{\mathbf{x}}_{k+1} - \tilde{\mathbf{x}}_k\|\mu_k + \kappa_t\|\mathbf{t}_{k+1} - \mathbf{t}_k\|\mu_k = \tau_k$, where $\lim_{k\to\infty} \tau_k = 0$. Therefore, we now address the convexity hypothesis.

To show the objective functions used to compute the updates in Algorithm 5.1 are convex,

we show that the Hessians are positive definite. The Hessian for the $\mathbf{t}$ update, Eq. 5.6, is

$$\nabla_{\mathbf{tt}}\mathfrak{L}(\mathbf{x}_{k+1}, \mathbf{t}, \tilde{\mathbf{x}}^i_{k+1}, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k) = \nabla_{\mathbf{tt}}\mathcal{F}(\mathbf{x}_{k+1}, \mathbf{t}) + \mu_k \mathbf{I}, \tag{5.13}$$

which is positive definite for all $\mathbf{t}_k$ sufficiently close to $\mathbf{t}^*$ and all k if it is positive definite for $k = 0$.[2] In addition, the Hessian for the $\mathbf{x}$ update, Eq. 5.4, is

$$\nabla_{\mathbf{xx}}\mathfrak{L}(\mathbf{x}, \mathbf{t}_k, \tilde{\mathbf{x}}^i_k, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k) = \nabla_{\mathbf{xx}}\mathcal{F}(\mathbf{x}, \mathbf{t}_k) + \sum_{i=1}^{m} \mu_k \mathbf{I}, \tag{5.16}$$

which like the Hessian for the $\mathbf{t}$ update is positive definite for all $\mathbf{x}_k$ sufficiently close to $\mathbf{x}^*$ and for all k if it is positive definite for $k = 0$.

The convexity issues arise in the Hessians for the $\tilde{\mathbf{x}}^i$ updates, Eq. 5.5,

$$\begin{aligned}\nabla_{\tilde{\mathbf{x}}^i\tilde{\mathbf{x}}^i}\mathfrak{L}(\mathbf{x}_{k+1}, \mathbf{t}_k, \tilde{\mathbf{x}}^i, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k) = \left[\lambda^i_k + \mu_k(r^i(\tilde{\mathbf{x}}^i, \mathbf{t}_k) - t^i_k)\right] \nabla_{\tilde{\mathbf{x}}^i\tilde{\mathbf{x}}^i} r^i(\tilde{\mathbf{x}}^i, \mathbf{t}_k) + \\ \mu_k \frac{\partial r^i(\tilde{\mathbf{x}}^i, \mathbf{t}_k)}{\partial \tilde{\mathbf{x}}^i} \frac{\partial r^i(\tilde{\mathbf{x}}^i, \mathbf{t}_k)}{\partial \tilde{\mathbf{x}}^i}^\top + \mu_k \mathbf{I}.\end{aligned} \tag{5.17}$$

The convexity of this update, IDF5, can breakdown in practice, and we will discuss ways to mitigate this concern in Section 5.2.3.

[2] Note because $r^i(\tilde{\mathbf{x}}^i_{k+1}, \mathbf{t}_k)$ is fixed in Eq. 5.6, the gradient with respect to $\mathbf{t}$ is

$$\begin{aligned}\nabla_{\mathbf{t}}\mathfrak{L}(\mathbf{x}_{k+1}, \mathbf{t}_{k+1}, \tilde{\mathbf{x}}^i_{k+1}, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k) = \frac{\partial \mathcal{F}(\mathbf{x}_{k+1}, \mathbf{t}_{k+1})}{\partial \mathbf{t}} - \\ \sum_{i=1}^{m} \left[\lambda^i_k \mathbf{e}^i + \mu_k(r^i(\tilde{\mathbf{x}}^i_{k+1}, \mathbf{t}_k) - t^i_{k+1})\mathbf{e}^i\right].\end{aligned} \tag{5.14}$$

However, in the actual system problem $r^i(\tilde{\mathbf{x}}^i_{k+1}, \mathbf{t})$ changes and the actual system gradient with respect to $\mathbf{t}$ should be

$$\begin{aligned}\nabla_{\mathbf{t}}\mathfrak{L}(\mathbf{x}_{k+1}, \mathbf{t}_{k+1}, \tilde{\mathbf{x}}^i_{k+1}, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k) = \frac{\partial \mathcal{F}(\mathbf{x}_{k+1}, \mathbf{t}_{k+1})}{\partial \mathbf{t}} + \\ \sum_{i=1}^{m} \left[\lambda^i_k \frac{\partial r^i(\tilde{\mathbf{x}}^i_{k+1}, \mathbf{t}_{k+1})}{\partial \mathbf{t}} - \lambda^i_k \mathbf{e}^i + \mu_k(r^i(\tilde{\mathbf{x}}^i_{k+1}, \mathbf{t}_{k+1}) - t^i_{k+1})\left(\frac{\partial r^i(\tilde{\mathbf{x}}^i_{k+1}, \mathbf{t}_{k+1})}{\partial \mathbf{t}} - \mathbf{e}^i\right)\right].\end{aligned} \tag{5.15}$$

Fixing $\mathbf{t}$ at the system level makes the system-level optimization problem more convex, however, it causes the terminal value of $\boldsymbol{\lambda}^*$ to be different from the true Lagrange multipliers for Eq. 5.2.

Quadratic Penalty Function Driven Convergence

In case (b), where we have that $\lim_{k\to\infty}\mu_k = \infty$, convergence of our IDF algorithm is caused by the quadratic penalty function. Under IDF1, the sequence of iterates generated by minimizing a quadratic penalty function terminates at a point minimizing the 2-norm of the constraint violation or if the constraints are linearly independent at the terminal point then the iterates terminate at a KKT point [83, Theorem 17.2]. This means that the IDF algorithm will converge to a locally optimal solution, $\bar{\mathbf{z}}^*$, if the constraints are linearly independent everywhere in the domain or if like asymptotically exact minimization in the method of multipliers the algorithm starts sufficiently close to locally optimal solution and always finds the optimal solution within the interior of the level set at $\bar{\mathbf{z}}_k$.

Consider the optimality conditions achieved in Algorithm 5.1, for the $\mathbf{t}$ update, Eq. 5.6,

$$0 = \nabla_{\mathbf{t}}\mathfrak{L}(\mathbf{x}_{k+1}, \mathbf{t}_{k+1}, \tilde{\mathbf{x}}^i_{k+1}, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k) = \frac{\partial \mathcal{F}(\mathbf{x}_{k+1}, \mathbf{t}_{k+1})}{\partial \mathbf{t}} - \sum_{i=1}^{m}\left[\lambda^i_k \mathbf{e}^i + \mu_k(r^i(\tilde{\mathbf{x}}^i_{k+1}, \mathbf{t}_k) - t^i_{k+1})\mathbf{e}^i\right], \tag{5.18}$$

for the $\tilde{\mathbf{x}}^i$ updates, Eq. 5.5,

$$0 = \nabla_{\tilde{\mathbf{x}}^i}\mathfrak{L}(\mathbf{x}_{k+1}, \mathbf{t}_k, \tilde{\mathbf{x}}^i_{k+1}, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k) = \left[\lambda^i_k + \mu_k(r^i(\tilde{\mathbf{x}}^i_{k+1}, \mathbf{t}_k) - t^i_k)\right]\frac{\partial r^i(\tilde{\mathbf{x}}^i_{k+1}, \mathbf{t}_k)}{\partial \tilde{\mathbf{x}}^i} - \boldsymbol{\nu}^i_k - \mu_k(\mathbf{x}_{k+1} - \tilde{\mathbf{x}}^i_{k+1}), \tag{5.19}$$

and for the $\mathbf{x}$ update, Eq. 5.4,

$$0 = \nabla_{\mathbf{x}}\mathfrak{L}(\mathbf{x}_{k+1}, \mathbf{t}_k, \tilde{\mathbf{x}}^i_k, \boldsymbol{\nu}_k, \boldsymbol{\lambda}_k, \mu_k) = \frac{\partial \mathcal{F}(\mathbf{x}_{k+1}, \mathbf{t}_k)}{\partial \mathbf{x}} + \sum_{i=1}^{m}\left[\boldsymbol{\nu}^i_k + \mu_k(\mathbf{x}_{k+1} - \tilde{\mathbf{x}}^i_k)\right]. \tag{5.20}$$

When these optimality conditions are written with all of the penalized terms on one side,

$$\begin{aligned}
\frac{1}{\mu_k}\left[\frac{\partial \mathcal{F}(\mathbf{x}_{k+1},\mathbf{t}_{k+1})}{\partial \mathbf{t}} - \sum_{i=1}^{m}\lambda_k^i \mathbf{e}^i\right] &= \sum_{i=1}^{m}(r^i(\tilde{\mathbf{x}}_{k+1}^i,\mathbf{t}_k) - t_{k+1}^i)\mathbf{e}^i \\
\frac{1}{\mu_k}\left[\lambda_k^i \frac{\partial r^i(\tilde{\mathbf{x}}_{k+1}^i,\mathbf{t}_k)}{\partial \tilde{\mathbf{x}}^i} - \boldsymbol{\nu}_k^i\right] &= -(r^i(\tilde{\mathbf{x}}_{k+1}^i,\mathbf{t}_k) - t_k^i)\frac{\partial r^i(\tilde{\mathbf{x}}_{k+1}^i,\mathbf{t}_k)}{\partial \tilde{\mathbf{x}}^i} + (\mathbf{x}_{k+1} - \tilde{\mathbf{x}}_{k+1}^i) \\
\frac{1}{\mu_k}\left[\frac{\partial \mathcal{F}(\mathbf{x}_{k+1},\mathbf{t}_k)}{\partial \mathbf{x}} + \sum_{i=1}^{m}\boldsymbol{\nu}_k^i\right] &= -\sum_{i=1}^{m}(\mathbf{x}_{k+1} - \tilde{\mathbf{x}}_k^i),
\end{aligned}$$

we see that with bounded Lagrange multiplier estimates, IDF4, and linearly independent constraint qualification, that $\bar{\mathbf{z}}^*$ is feasible, i.e. that $\lim_{k\to\infty}\|\mathbf{q}_k\| = 0$. In addition, combining the optimality conditions Eqs. 5.18, 5.19 and 5.20 with the Lagrange multiplier updates, Eqs. 5.7 and 5.8, we see that when $\|\mathbf{q}_k\| \to 0$,

$$\begin{aligned}
\frac{\partial \mathcal{F}(\mathbf{x}_{k+1},\mathbf{t}_{k+1})}{\partial t^i} &= \lambda_{k+1}^i \\
\left[\lambda_{k+1}^i + \mu_k(t_{k+1}^i - t_k^i)\right]\frac{\partial r^i(\tilde{\mathbf{x}}_{k+1}^i,\mathbf{t}_k)}{\partial \tilde{\mathbf{x}}^i} &= \boldsymbol{\nu}_{k+1}^i \\
\frac{\partial \mathcal{F}(\mathbf{x}_{k+1},\mathbf{t}_k)}{\partial \mathbf{x}} &= -\sum_{i=1}^{m}\left[\boldsymbol{\nu}_{k+1}^i + \mu_k(\tilde{\mathbf{x}}_{k+1}^i - \tilde{\mathbf{x}}_k^i)\right],
\end{aligned}$$

that the Lagrange multiplier estimates converge. Therefore, under a set of restrictive assumptions, this section has provided theory to support the observed convergence behavior of Algorithm 5.1.

Discussion of Assumptions

We now assess the practical limitations of the assumptions that (i) the Lagrange multiplier estimates remain bounded and (ii) the augmented Lagrangian is convex in $\tilde{\mathbf{x}}^i$ for all iterations. In practice (ii) should enforce (i), but a proof of that is intricate. With $\mu_k > \hat{\mu}$ the terms in $\nabla_{\tilde{\mathbf{x}}^i\tilde{\mathbf{x}}^i}\mathfrak{L}(\mathbf{x}_{k+1},\mathbf{t}_k,\tilde{\mathbf{x}}^i,\boldsymbol{\nu}_k,\boldsymbol{\lambda}_k,\mu_k)$, Eq. 5.17, are all guaranteed to be positive definite except for

$$\left[\lambda_k^i + \mu_k(r^i(\tilde{\mathbf{x}}^i,\mathbf{t}_k) - t_k^i)\right]\nabla_{\tilde{\mathbf{x}}^i\tilde{\mathbf{x}}^i} r^i(\tilde{\mathbf{x}}^i,\mathbf{t}_k). \tag{5.21}$$

So the convexity requirement implies that this term needs to either be positive definite or small compared to the other terms in Eq. 5.17. The convexity requirement is certain if $r^i(\tilde{\mathbf{x}}^i, \mathbf{t}_k)$ is linear, or when $\tilde{\mathbf{x}}^i_k$, $\mathbf{t}_k$ and λ^i_k are sufficiently close to the optimal solutions. A practical limitation is that increasing μ_k too quickly or large updates in $\boldsymbol{\lambda}^i_k$ can violate the convexity requirement. It is informative to consider this term written as

$$\left[\lambda^i_{k+1} + \mu_k(t^i_k - t^i_{k+1})\right] \nabla_{\tilde{\mathbf{x}}^i\tilde{\mathbf{x}}^i} r^i(\tilde{\mathbf{x}}^i, \mathbf{t}_k). \tag{5.22}$$

So, if $\nabla_{\tilde{\mathbf{x}}^i\tilde{\mathbf{x}}^i} r^i(\tilde{\mathbf{x}}^i, \mathbf{t}_k)$ is not positive definite (or if it is positive definite but the coefficient in front is negative) and $\mu_k \to \infty$ faster than $\|\mathbf{t}_k - \mathbf{t}_{k+1}\| \to 0$ the convexity requirement may be violated. A first solution is to perform cycles of Eqs. 5.6, 5.5, and 5.4 without updating the Lagrange multipliers or penalty parameter. If at each iteration a stationary point of the augmented Lagrangian is found, the stricter convergence result of [12, Proposition 2.4], will hold. Another potential solution to maintain the convexity requirement is to use a slow growth rate for μ_k, which unfortunately is unknown *a priori*. So, if convergence difficulties occur, it should be assumed the discipline performance metrics have large curvature and the growth rate of μ_k, γ, should be lowered or additional cycles of Eqs. 5.6, 5.5, and 5.4 should be used without changing either the penalty parameter or Lagrange multiplier estimates.

5.3 Parallel Function Evaluations

The method developed in this section is a parallelized method for solving multidisciplinary system design optimization problems that enables multiple fidelity levels and does not require high-fidelity gradient-information. The optimization problem solved is an objective function subject to a vector of equality constraints, so this algorithm is applicable to general equality constrained optimization problems in addition to system design problems.

The concept behind this optimization method (as shown in Figure 5-2(b)) is to create surrogate models of all expensive functions in an optimization problem in parallel, and to solve inexpensive surrogate-based optimization problems. The surrogate models used are at least fully linear, satisfying Eqs. 2.7 and 2.8. This means that first-order consistent surro-

gates and inexpensive original functions can be included in the surrogate-based optimization if that is desired to speed finding an optimal solution. The fully linear models are all constructed in parallel, and the design sites used to create the surrogates may also be evaluated in parallel. The surrogate-based optimization strategy is a composite-step trust-region algorithm based on the Celis-Dennis-Tapia (CDT) trust-region algorithm [21] and the filter method of Fletcher and Leyffer [40]. Section 5.3.1 presents an alternative formulation of the system design problem, (5.1), that can be solved by evaluating expensive functions in parallel. Section 5.3.2 presents the trust-region algorithm to solve the design problem in parallel, and Section 5.3.3 presents theory supporting the method. Additional details of the theory are left to Appendix A. An implementation of the algorithm is presented as Algorithm 5.2. Section 5.3.4 presents the changes to Algorithm 5.2 necessary when all surrogate models are first-order consistent or better, and an overview of the gradient-exploiting version of Algorithm 5.2 is presented as Algorithm 5.3.

5.3.1 Parallel All-At-Once Formulation

To solve (5.1) using parallel function evaluations we relax both the multidisciplinary feasibility and also the discipline-level feasibility. This formulation is often referred to as an All-At-Once (AAO) formulation because the optimizer handles all of the constraints. In Figure 5-1(b) this corresponds to allowing the constraints within disciplines, i.e., discipline-level state equations, $\mathcal{R}^i(\mathbf{x}, \mathbf{t}^{1,\ldots,m\setminus i}, \mathbf{u}^i(\mathbf{x}, \mathbf{r}^{1,\ldots,m\setminus i})) = 0$, to be violated during the optimization in addition to the interdisciplinary coupling relaxations in the IDF formulation. The All-At-Once (AAO) formulation used in this thesis is an equality constrained optimization problem,

$$\begin{aligned}
\min_{\mathbf{x}\in\mathbb{R}^n, \mathbf{t}\in\mathbb{R}^v, \mathbf{u}\in\mathbb{R}^l, \mathbf{p}\in\mathbb{R}^q} \quad & \mathcal{F}(\mathbf{x}, \mathbf{t}) \\
\text{s.t. } & \mathcal{G}(\mathbf{x}, \mathbf{t}) + \mathbf{p}^2 = 0 \\
& \mathcal{H}(\mathbf{x}, \mathbf{t}) = 0 \\
& \mathbf{r}^i(\mathbf{x}, \mathbf{t}^{1,\ldots,m\setminus i}, \mathbf{u}^i(\mathbf{x}, \mathbf{r}^{1,\ldots,m\setminus i})) - \mathbf{t}^i = 0, \qquad \forall i = 1, \ldots, m \\
& \mathcal{R}^i(\mathbf{x}, \mathbf{t}^{1,\ldots,m\setminus i}, \mathbf{u}^i(\mathbf{x}, \mathbf{r}^{1,\ldots,m\setminus i})) = 0, \qquad \forall i = 1, \ldots, m
\end{aligned} \tag{5.23}$$

where $\mathbf{p}$ is a vector of slack variables and $\mathbf{p}^2$ denotes squaring each element of $\mathbf{p}$. To solve this optimization problem in parallel we create surrogate models of all expensive functions by evaluating them at many locations concurrently. This corresponds to Schnabel's first method of parallelizing an optimization, evaluating the functions at many locations simultaneously [102], and is shown in Figure 5-2(b). The method to solve (5.23) by parallel construction of gradient-free surrogate models is presented in Section 5.3.2.

5.3.2 AAO Trust-Region Algorithm

To present this algorithm we consider the generic equality constrained optimization problem,

$$\begin{aligned} &\min_{\mathbf{x}} \quad \mathcal{F}(\mathbf{x}) \\ &\text{s.t. } \mathcal{H}(\mathbf{x}) = 0. \end{aligned} \tag{5.24}$$

The AAO optimization problem, (5.23), can easily be written in this standard form, with $\mathbf{x}$ now representing the composite vector of design variables, targets, state variables, and slack variables, and $\mathcal{F}(\mathbf{x})$ and $\mathcal{H}(\mathbf{x})$ being defined appropriately.

The AAO trust-region algorithm decouples the constrained optimization into two parts, first a normal step, $\mathbf{s}_k^{\perp}$, that decreases the constraint violation, and second, a tangential step, $\mathbf{s}_k^{\parallel}$, that decreases the objective function value while maintaining some of the constraint violation reduction from the normal step.[3] The trust-region update step is accordingly the sum of the normal and tangential steps, $\mathbf{s}_k = \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\parallel}$. The trust region step is accepted if the pair $(\mathcal{F}(\mathbf{x}_k + \mathbf{s}_k), \|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\|)$ is Pareto dominant compared with all previous iterates $\mathbf{x}_k$, and rejected otherwise. We will refer to the fully linear surrogate model at an iteration k for $\mathcal{F}(\mathbf{x})$ as $\bar{\mathcal{F}}_k(\mathbf{x})$, and the fully linear surrogates for $\mathcal{H}(\mathbf{x})$ as $\bar{\mathcal{H}}_k(\mathbf{x})$.

The first step of the algorithm is to find a normal step that reduces the constraint violation. To do this we compute a step minimizing the 2-norm of the constraint violation

[3] The terms normal and tangential do not imply that the steps are orthogonal [31, Page 658].

using the surrogate models,

$$\begin{aligned} \min_{\mathbf{s}_k^{\perp}} \quad & \frac{1}{2}\|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp})\|_2^2 \\ \text{s.t.} \quad & \|\mathbf{s}_k^{\perp}\| \leq \Delta_k. \end{aligned} \tag{5.25}$$

We require that the normal step satisfies a condition slightly stricter than the fraction of Cauchy decrease condition, therefore the solution to Eq. 5.25 must satisfy,

$$\|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp})\|_2^2 \geq \kappa_{amm-H}\left[\|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp *})\|_2^2\right] \tag{5.26}$$

for a constant $\kappa_{amm-H} \in (0, 1]$, where $\mathbf{s}_k^{\perp *}$ is the exact solution to Eq. 5.25. An immediate consequence of Eq. 5.26 is the fraction of Cauchy decrease condition,

$$\begin{aligned} &\|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp})\|_2^2 \geq \\ &\qquad 2\kappa_{fcd}\|\nabla\bar{\mathcal{H}}(\mathbf{x}_k)^{\top}\bar{\mathcal{H}}(\mathbf{x}_k)\| \min\left\{\Delta_k, \frac{\|\nabla\bar{\mathcal{H}}(\mathbf{x}_k)^{\top}\bar{\mathcal{H}}(\mathbf{x}_k)\|}{\|\nabla\bar{\mathcal{H}}(\mathbf{x}_k)^{\top}\nabla\bar{\mathcal{H}}(\mathbf{x}_k) + \sum_{i=1}^{m}\bar{h}_k^i(\mathbf{x}_k)\nabla^2\bar{h}_k^i(\mathbf{x}_k)\|}\right\}, \end{aligned} \tag{5.27}$$

where $\bar{h}_k^i(\mathbf{x})$ is the i^{th} fully linear surrogate model of $\bar{\mathcal{H}}_k(\mathbf{x})$ and $\kappa_{fcd} \in (0, 1)$, [31, Theorem 6.3.5].

After computing the normal step to reduce the constraint violation we seek a tangential step that reduces the objective function value without giving up all of the progress achieved towards feasibility. To find the tangential step, $\mathbf{s}_k^{\parallel}$, we solve a trust-region problem that is based on the Celis-Dennis-Tapia trust-region algorithm [21]. We require at the complete step, normal plus tangential, that the fraction of Cauchy decrease condition for the constraint violation is still satisfied. Therefore, we define an additional constant, $\eta \in (0, 1)$, in order to require $\|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\parallel})\|_2^2 \leq \|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \eta\left(\|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp})\|_2^2\right)$. The constant, η, can be considered the weighting placed on whether the optimizer focuses on maintaining the constraint violation reduction, $\eta \sim 1$, or reducing the objective function, $\eta \sim 0$; however, regardless of the value chosen, the algorithm will converge to a feasible point that minimizes

the objective function. This tangential step trust-region subproblem is now,

$$\begin{aligned} \min_{\mathbf{s}_k^{\|}} \quad & \bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\|}) \\ \text{s.t.} \quad & \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\|})\|_2^2 \leq \|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \eta\left(\|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp})\|_2^2\right) \\ & \|\mathbf{s}_k^{\perp} + \mathbf{s}_k^{\|}\| \leq \Delta_k. \end{aligned} \tag{5.28}$$

An alternative tangential step trust region subproblem, which is easier to compute, is based on the Gauss-Newton approximation. The Gauss-Newton normal step using the surrogate models of the constraints, $\tilde{\mathbf{s}}_k^{\perp}$, is computed as

$$\begin{aligned} \min_{\tilde{\mathbf{s}}_k^{\perp}} \quad & \frac{1}{2}\|\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)\tilde{\mathbf{s}}_k^{\perp} + \bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 \\ \text{s.t.} \quad & \|\tilde{\mathbf{s}}_k^{\perp}\| \leq \Delta_k. \end{aligned} \tag{5.29}$$

The value of the Gauss-Newton step is that the fraction of Cauchy decrease is the same for it as for (5.25) when the trust region is small. The fraction of Cauchy decrease requirement for (5.29) is

$$b_k(\mathbf{0}) - b_k(\tilde{\mathbf{s}}_k^{\perp}) \geq \kappa_{fcd}\|\nabla\bar{\mathcal{H}}(\mathbf{x}_k)^{\top}\bar{\mathcal{H}}(\mathbf{x}_k)\| \min\left\{\Delta_k, \frac{\|\nabla\bar{\mathcal{H}}(\mathbf{x}_k)^{\top}\bar{\mathcal{H}}(\mathbf{x}_k)\|}{\|\nabla\bar{\mathcal{H}}(\mathbf{x}_k)^{\top}\nabla\bar{\mathcal{H}}(\mathbf{x}_k)\|}\right\}, \tag{5.30}$$

where the merit function is

$$\begin{aligned} b_k(\mathbf{s}^{\perp}) &= \frac{1}{2}\|\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)\mathbf{s}^{\perp} + \bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 \\ &= \frac{1}{2}\bar{\mathcal{H}}_k(\mathbf{x}_k)^{\top}\bar{\mathcal{H}}_k(\mathbf{x}_k) + \left(\mathbf{s}^{\perp}\right)^{\top}\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)^{\top}\bar{\mathcal{H}}_k(\mathbf{x}_k) + \frac{1}{2}\left(\mathbf{s}^{\perp}\right)^{\top}\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)^{\top}\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)\mathbf{s}^{\perp}. \end{aligned} \tag{5.31}$$

This slightly weaker fraction of Cauchy decrease condition is significantly less computationally expensive than the fraction of Cauchy decrease requirement in Eq. 5.27. The alternative

tangential step calculation is,

$$\begin{aligned} \min_{\mathbf{s}_k^{\|}} \quad & \bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\|}) \\ \text{s.t.} \quad & b_k(\mathbf{s}_k^{\perp} + \mathbf{s}_k^{\|}) \leq b_k(\mathbf{0}) - \eta[b_k(\mathbf{0}) - b_k(\mathbf{s}_k^{\perp})] \\ & \|\mathbf{s}_k^{\perp} + \mathbf{s}_k^{\|}\| \leq \Delta_k. \end{aligned} \tag{5.32}$$

For whichever tangential step trust region problem is used, we require that the tangential step optimization be solved such that,

$$\bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp}) - \bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\|}) \geq \kappa_{amm-F} \left[\bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp}) - \bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\|*}) \right], \tag{5.33}$$

where $\mathbf{s}_k^{\|*}$ is the exact solution to either Eq. 5.32 or Eq. 5.28, and κ_{amm-F} is a constant in $(2/(2+\sqrt{1-\eta^2}), 1)$.

The step acceptance and trust-region update criterion for this algorithm are a filter based on [40], and enhancements from [39, 41]. The filter uses ideas from multiobjective optimization and considers reducing the objective $\mathcal{F}(\mathbf{x})$ and constraint violation $\|\mathcal{H}(\mathbf{x})\|$ as two goals of the optimization. The filter stores the Pareto front of pairs $(\mathcal{F}(\mathbf{x}_k), \|\mathcal{H}(\mathbf{x}_k)\|)$, and a trust-region step $\mathbf{x}_k + \mathbf{s}_k$ is acceptable to the filter if it is not dominated by a previous iterate, i.e., it will appear on the filter Pareto front. The filter algorithm also requires a sufficient decrease condition so filter points cannot be arbitrarily close. Therefore a point $\mathbf{x}_k + \mathbf{s}_k$ is *non-dominated* and acceptable for the filter if for all $\mathbf{x}_k$ visited $(\mathbf{x}_{\hat{k}}, \hat{k} = 0, \ldots, k)$,

$$\mathcal{F}(\mathbf{x}_k + \mathbf{s}_k) \leq \mathcal{F}(\mathbf{x}_{\hat{k}}) - \beta\hat{\Delta}_k^2 \quad \text{or} \quad \|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\| \leq \|\mathcal{H}(\mathbf{x}_{\hat{k}})\| - \beta\hat{\Delta}_k^2\|\mathcal{H}(\mathbf{x}_{\hat{k}})\|, \quad \forall \hat{k} \leq k, \tag{5.34}$$

for a constant $\beta \in (0, \frac{1}{2})$ and where $\hat{\Delta}_k = \min\{\Delta_k, 1\}$. We typically initiate the filter with the point $(-\infty, \|\mathcal{H}(\mathbf{x}_0)\|)$ in lieu of $(\mathcal{F}(\mathbf{x}_0), \|\mathcal{H}(\mathbf{x}_0)\|)$ to ensure the constraint violation remains bounded. The points acceptable to the filter are shown graphically in Figure 5-3. If a point is acceptable to the filter, it is added to the filter and any previous points in the filter that

are then dominated are removed.

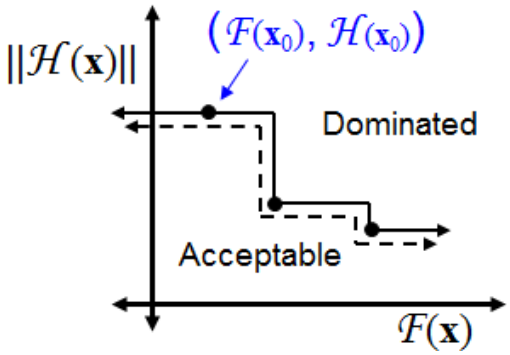


Figure 5-3: Graphic of points acceptable to the filter.

The trust region updates are all based on the whether or not a trial point is acceptable to the filter. If the trial point $\mathbf{x}_k+\mathbf{s}_k$ is acceptable to the filter, then the design vector is updated, $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$, but if it is not acceptable the step, $\mathbf{s}_k$, is rejected. Similarly, if the step is accepted the size of the trust region is increased so that $\Delta_{k+1} = \min\{\gamma_1\Delta_k, \Delta_{\max}\}$ and if the step is rejected the size of the trust region is decreased so that $\Delta_{k+1} = \gamma_0\Delta_k$ with $\gamma_1 > 1$ and $\gamma_0 \in (0, 1)$. $\bar{\mathcal{F}}(\mathbf{x})$ and $\bar{\mathcal{H}}(\mathbf{x})$ are then updated to be fully linear on $\|\mathbf{x} - \mathbf{x}_{k+1}\| \leq \Delta_{k+1}$. We require that the surrogate models have the correct value at $\mathbf{x}_{k+1}$, i.e. $\mathcal{H}(\mathbf{x}_{k+1}) = \bar{\mathcal{H}}(\mathbf{x}_{k+1})$ and $\mathcal{F}(\mathbf{x}_{k+1}) = \bar{\mathcal{F}}_{k+1}(\mathbf{x}_{k+1})$. The surrogate models for each constraint are all created concurrently. Moreover, if no designs have been evaluated in the vicinity of the current trust region, then construction of a fully linear model requires $n + 1$ designs to be evaluated. All of these designs may also be evaluated concurrently.

To establish optimality of $\mathbf{x}_k$ and terminate the algorithm, it is necessary to verify that both the constraint violation and the projection of the objective function gradient onto the nullspace of the constraint Jacobian are sufficiently small. In the general gradient-free case, the second condition cannot be verified directly. Therefore, we require three criteria to terminate the algorithm, (1) $\|\mathcal{H}(\mathbf{x}_k)\| \leq \epsilon$, (2) $\|\bar{P}_k\nabla\bar{\mathcal{F}}_k(\mathbf{x}_k)\| \leq \epsilon$, and (3) $\Delta_k \leq \epsilon$, where

$$\bar{P}_k = I - \nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)^\top \left(\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\right)^{-1} \nabla\bar{\mathcal{H}}_k(\mathbf{x}_k). \tag{5.35}$$

Using fully linear surrogate models, conditions (2) and (3) are equivalent to the projection of

the objective function gradient onto the nullspace of the constraint Jacobian being sufficiently small [38]. The complete trust-region algorithm is summarized as Algorithm 5.2.

5.3.3 AAO Trust-Region Algorithm Supporting Theory

In practice we observe that Algorithm 5.2 is able to find optimal solutions to (5.24) without using gradients of either the objective function or constraints. This section develops some theory to support the observed behavior of Algorithm 5.2. The explanation is separated into four parts, we first discuss (i) the constraint violation reduction obtained at each iteration, and then discuss (ii) the interaction between the constraint violation and the trust region size caused by the filter. We then switch to the objective function and discuss (iii) the reduction in the objective function obtained at each iteration when the constraint violation is small, and conclude by showing (iv) that the filter acceptance criteria ensures that Algorithm 5.2 will asymptotically approach an optimal solution to (5.24). Details of the analysis are left to Appendix A. We also use four assumptions in this section and for shorthand refer to them by number.

Assumptions:

AAO1. All functions in Eq. 5.24 are twice continuously differentiable and have finite upper bounds for the 2-norm of their gradients and Hessians. (This condition implies that the functions have Lipschitz continuous first derivatives.)

AAO2. The sequence of iterates $\{\mathbf{x}_k\}$ lies in a compact convex domain Ω.

AAO3. $\mathcal{F}(\mathbf{x})$ is bounded within Ω.

AAO4. $\nabla\mathcal{H}(\mathbf{x}_k)$ has full row rank for all k. Specifically, there exists $\kappa_\tau > 0$ such that for any $\mathbf{p} \neq 0$, $\min \|\nabla\mathcal{H}(\mathbf{x}_k)^\top \mathbf{p}\|/\|\mathbf{p}\| > \kappa_\tau$. Therefore, if $\|\mathcal{H}(\mathbf{x}_k)\| > 0$, then $\|\nabla\mathcal{H}(\mathbf{x}_k)^\top \mathcal{H}(\mathbf{x}_k)\| = \varrho(\mathbf{x}_k) > \kappa_\tau \|\mathcal{H}(\mathbf{x}_k)\|$.

Algorithm 5.2: Gradient-free AAO Trust-Region Algorithm

0: Choose, $\mathbf{x}_0, \Delta_0, \Delta_{\max}, \eta, \beta$, and tolerance ϵ. (Recommended values are given in Table 5.2.) Set the initial filter to $\{(-\infty, \|\mathcal{H}(\mathbf{x}_0)\|)\}$. Create $\bar{\mathcal{F}}_0(\mathbf{x})$ and $\bar{\mathcal{H}}_0(\mathbf{x})$ to be fully linear models of $\mathcal{F}(\mathbf{x})$ and $\mathcal{H}(\mathbf{x})$ on $\|\mathbf{x}-\mathbf{x}_0\| \leq \Delta_0$, and ensure $\mathcal{H}(\mathbf{x}_0) = \bar{\mathcal{H}}_0(\mathbf{x}_0)$, $\mathcal{F}(\mathbf{x}_0) = \bar{\mathcal{F}}_0(\mathbf{x}_0)$.

1: Compute the normal step, $\mathbf{s}_k^{\perp}$, to reduce the constraint violation by solving,

$$\begin{aligned} \min_{\mathbf{s}_k^{\perp}} \quad & \frac{1}{2}\|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp})\|_2^2 \\ \text{s.t.} \quad & \|\mathbf{s}_k^{\perp}\| \leq \Delta_k. \end{aligned}$$

2: Compute the tangential step, $\mathbf{s}_k^{\parallel}$, to reduce the objective function while maintaining sufficient decrease in the constraint violation by solving,

$$\begin{aligned} \min_{\mathbf{s}_k^{\parallel}} \quad & \bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\parallel}) \\ \text{s.t.} \quad & \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\parallel})\|_2^2 \leq \|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \eta\left(\|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp})\|_2^2\right) \\ & \|\mathbf{s}_k^{\perp} + \mathbf{s}_k^{\parallel}\| \leq \Delta_k. \end{aligned}$$

3: Where $\mathbf{s}_k = \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\parallel}$, evaluate $\mathcal{F}(\mathbf{x}_k + \mathbf{s}_k)$ and $\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)$.

4: Check if the pair $(\mathcal{F}(\mathbf{x}_k+\mathbf{s}_k), \|\mathcal{H}(\mathbf{x}_k+\mathbf{s}_k)\|)$, is *non-dominated* and therefore acceptable to the filter, i.e. if

$$\mathcal{F}(\mathbf{x}_k + \mathbf{s}_k) \leq \mathcal{F}(\mathbf{x}_{\hat{k}}) - \beta\hat{\Delta}_k^2 \quad \text{or} \quad \|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\| \leq \|\mathcal{H}(\mathbf{x}_{\hat{k}})\| - \beta\hat{\Delta}_k^2\|\mathcal{H}(\mathbf{x}_{\hat{k}})\|, \ \hat{k} = 0, \ldots, k.$$

5: Update the trust region based on whether or not $\mathbf{x}_k + \mathbf{s}_k$ is acceptable,

$$(\mathbf{x}_{k+1}, \Delta_{k+1}) \quad = \quad \begin{cases} \mathbf{x}_k + \mathbf{s}_k, \ \min\{2\Delta_k, \Delta_{\max}\} & \text{if acceptable} \\ \mathbf{x}_k, \ \gamma_0\Delta_k & \text{otherwise.} \end{cases}$$

5a: If $\mathbf{x}_k + \mathbf{s}_k$ is accepted, add $(\mathcal{F}(\mathbf{x}_k + \mathbf{s}_k), \|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\|)$ to the filter and remove all dominated entries.

5b: Create $\bar{\mathcal{F}}_{k+1}(\mathbf{x})$ and $\bar{\mathcal{H}}_{k+1}(\mathbf{x})$ fully linear on $\|\mathbf{x} - \mathbf{x}_{k+1}\| \leq \Delta_{k+1}$, and ensure $\mathcal{H}(\mathbf{x}_{k+1}) = \bar{\mathcal{H}}_{k+1}(\mathbf{x}_{k+1})$, $\mathcal{F}(\mathbf{x}_{k+1}) = \bar{\mathcal{F}}_{k+1}(\mathbf{x}_{k+1})$.

6: If $\|\mathcal{H}(\mathbf{x}_{k+1})\| \leq \epsilon$, $\bar{P}_{k+1}\nabla\bar{\mathcal{F}}_{k+1}(\mathbf{x}_{k+1}) \leq \epsilon$ and $\Delta_k \leq \epsilon$ the algorithm has converged, otherwise go to step 1.

Constraint Violation Decrease

In this section we demonstrate that the squared constraint violation decreases proportionally to the trust region size for each iteration of Algorithm 5.2 when the trust region is sufficiently small. From AAO1 and AAO4, if the constraint violation is non-zero there exists a positive constant, $\varrho(\mathbf{x}_k)$, such that $\|\nabla\mathcal{H}(\mathbf{x}_k)^\top\mathcal{H}(\mathbf{x}_k)\| = \varrho(\mathbf{x}_k) > 0$. This fact enables us to bound the error between the surrogate quantity used in Eqs. 5.27 and 5.30, $\|\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\mathcal{H}(\mathbf{x}_k)\|$, and the high-fidelity counterpart, $\nabla\mathcal{H}(\mathbf{x}_k)^\top\mathcal{H}(\mathbf{x}_k)$. (Note: $\bar{\mathcal{H}}(\mathbf{x}_k) = \mathcal{H}(\mathbf{x}_k)$ by construction.) Accordingly, by the definition of a fully linear model, for each Δ_k sufficiently small, say $\Delta_k \in (0, \delta_2(\mathbf{x}_k))$, $\delta_2(\mathbf{x}_k) > 0$, the steps computed by Algorithm 5.2 ensure

$$\|\mathcal{H}(\mathbf{x}_k)\|^2 - \|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\|^2 \geq \eta\kappa_{fcd}\Delta_k\varrho(\mathbf{x}_k). \tag{5.36}$$

This means that the steps generated by Algorithm 5.2 decrease the squared constraint violation proportionally to the size of the trust region when the trust region size is small enough. For details of this analysis see Appendix A.1. We now demonstrate this decrease must also be accepted by the filter for small Δ_k.

Constraint Violation and Filter

Figure 5-4 shows a schematic of a filter with four accepted iterates with four areas labeled, each representing the minimal possible additional area that must be enclosed by the filter if an iterate is accepted in that interval. As an example, area 2 represents the minimal additional area the filter will gain if an iterate is accepted between $(\mathcal{F}(\mathbf{x}_1), \|\mathcal{H}(\mathbf{x}_1)\|)$ and $(\mathcal{F}(\mathbf{x}_2), \|\mathcal{H}(\mathbf{x}_2)\|)$. The dimension of this minimal additional area is $\beta\Delta_k^2 \times \beta\Delta_k^2\|\mathcal{H}(\mathbf{x}_1)\|$. This holds for all intervals except for the final, between $(\mathcal{F}(\mathbf{x}_3), \|\mathcal{H}(\mathbf{x}_3)\|)$ and $(\mathcal{F}(\mathbf{x}_{\max}), 0)$—area 4. On this interval the reduction in the constraint violation must be at least $\beta\Delta_k^2\|\mathcal{H}(\mathbf{x}_3)\|$. The fact that the area enclosed by the filter increases every time an iterate is accepted, but the total area the filter may enclose is bounded by properties of (5.24) establishes that

$$\lim_{k\to\infty} \|\mathcal{H}(\mathbf{x}_k)\|\Delta_k = 0. \tag{5.37}$$

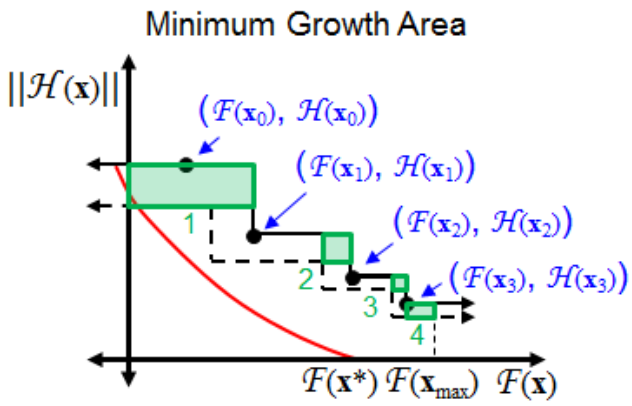


Figure 5-4: Minimum size of possible areas that will be dominated due to acceptance of another iterate by the filter.

Accordingly, if an infinite number of iterates are accepted by the filter then either the constraint violation or trust region size must go to zero. Details of this analysis are presented in Appendix A.2.

Using this property of the filter acceptance criteria, we now demonstrate that for each $\mathbf{x}_k \in \Omega$ there is a minimum size of the trust region for which the filter accepts a nearby point with lower constraint violation. The filter accepts any step provided that $\|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\| \leq (1 - \beta\hat{\Delta}_k^2)\|\mathcal{H}(\mathbf{x}_k)\|$. Meanwhile, Eq. 5.36 demonstrated that for each $\Delta_k \in (0, \delta_2(\mathbf{x}_k))$,

$$\|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\| \leq \|\mathcal{H}(\mathbf{x}_k)\|\sqrt{1 - \frac{\eta\kappa_{fcd}\Delta_k\varrho(\mathbf{x}_k)}{\|\mathcal{H}(\mathbf{x}_k)\|^2}}. \tag{5.38}$$

Therefore, if $\sqrt{1 - \frac{\eta\kappa_{fcd}\Delta_k\varrho(\mathbf{x}_k)}{\|\mathcal{H}(\mathbf{x}_k)\|^2}} \leq 1 - \beta\hat{\Delta}_k^2$, then $\mathbf{x}_k + \mathbf{s}_k$ is acceptable to the filter, $\mathbf{x}_{k+1} \neq \mathbf{x}_k$ and $\Delta_{k+1} > \Delta_k$. As both the left and right sides of the inequality are less than one, we can establish that $\mathbf{x}_k + \mathbf{s}_k$ is accepted by the filter when $\frac{\eta\kappa_{fcd}\varrho(\mathbf{x}_k)}{\|\mathcal{H}(\mathbf{x}_k)\|^2} \geq 2\beta\Delta_k - \beta^2\Delta_k^3$. As the positive real root(s) of

$$\Delta_k\left(1 - \frac{\beta}{2}\Delta_k^2\right) = \frac{\eta\kappa_{fcd}\kappa_\tau}{2\beta\|\mathcal{H}(\mathbf{x}_k)\|}$$

are between $\sqrt{2}$ and $+\infty$, each $\Delta_k \in (0, \min\{\delta_2(\mathbf{x}_k), \sqrt{2}\})$ ensures $\mathbf{x}_k + \mathbf{s}_k$ is acceptable to

the filter and that $\Delta_{k+1} > \Delta_k$. The result is that the constraint violation of the sequence of iterates generated by Algorithm 5.2 goes to zero.

Objective Function Decrease

We now demonstrate that when the constraint violation and the trust region size are small, then each step of Algorithm 5.2 decreases the objective function value proportionally to the size of the trust region. To demonstrate this we consider $\mathbf{s}_k$ projected in the range-space and nullspace of the surrogate constraint Jacobian. In Appendix A.3, we demonstrate that there exists an ϵ such that when $\|\mathcal{H}(\mathbf{x}_k)\| < \epsilon$ and Δ_k is sufficiently small, then

$$\bar{\mathcal{F}}_k(\mathbf{x}_k) - \bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k) \geq \kappa_{fcd2} \|\bar{P}_k \nabla \bar{\mathcal{F}}_k(\mathbf{x}_k)\| \Delta_k. \tag{5.39}$$

We define the projection onto the nullspace of the constraint Jacobian as,

$$P_k = I - \nabla \mathcal{H}(\mathbf{x}_k)^\top \left(\nabla \mathcal{H}(\mathbf{x}_k) \nabla \mathcal{H}(\mathbf{x}_k)^\top \right)^{-1} \nabla \mathcal{H}(\mathbf{x}_k). \tag{5.40}$$

By bounding the error $\|P_k \nabla \mathcal{F}(\mathbf{x}_k) - \bar{P}_k \nabla \bar{\mathcal{F}}(\mathbf{x}_k)\|$, we are able to show that for small Δ_k, Eq. 5.39 ensures that

$$\mathcal{F}(\mathbf{x}_k) - \mathcal{F}(\mathbf{x}_k + \mathbf{s}_k) \geq \frac{\kappa_{fcd2}}{2} \|P_k \nabla \mathcal{F}(\mathbf{x}_k)\| \Delta_k. \tag{5.41}$$

This shows that for $\|\mathcal{H}(\mathbf{x}_k)\| < \epsilon$, then each Δ_k sufficiently small decreases the objective function proportionally to the projection of the objective gradient onto the nullspace of the constraint Jacobian and the trust region size. Complete details of this analysis are presented in Appendix A.3.

Objective Function and Filter

We now show that the objective function decrease established in (5.41) ensures that if $\|\mathcal{H}(\mathbf{x}_k)\| < \epsilon$, then if $\|P_k \nabla \mathcal{F}(\mathbf{x}_k)\| > 0$ there exists $\delta_{\min}(\mathbf{x}_k) > 0$ which is a lower bound for Δ_k at which Algorithm 5.2 must generate a step acceptable to the filter that improves the ob-

jective function value. Since the current $\mathbf{x}_k$ is always on the current filter, we need to consider two cases, (i) the constraint violation of the current iterate is arbitrarily small but greater than zero, i.e. $\|\mathcal{H}(\mathbf{x}_k)\| \in (0, \epsilon_4)$ for some $\epsilon_4 > 0$ and (ii) the iterate exactly satisfies the constraints. When Δ_k is small enough that (5.41) holds, then each $\Delta_k < \kappa_{fcd}\|P_k \nabla \mathcal{F}(\mathbf{x}_k)\|/2\beta$ ensures the objective function reduction requirement of the filter is met. However, to show that $\mathbf{x}_k + \mathbf{s}_k$ is non-dominated, we also must show the constraint violation does not increase significantly, i.e., it is still less than $1 - \beta\Delta_k^2$ times the constraint violation of the point on the filter with the next highest constraint violation. Any $\Delta_k \in (0, \delta_2(\mathbf{x}_k))$ ensures $\|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\| < \|\mathcal{H}(\mathbf{x}_{k-1})\|$, so case (i) is complete. In case (ii) we require that the increase in the constraint violation from zero caused by the surrogate modeling and Gauss-Newton inaccuracies in Algorithm 5.2 be less than $1-\beta\Delta_k^2$ times the constraint violation of the point on the filter with the second lowest constraint violation. As demonstrated in Appendix A.4, a positive Δ_k ensures this. Therefore, any $\Delta_k > \delta_{\min}(\mathbf{x}_k)$ ensures the size of the trust region is bounded from below unless $\|P_k \nabla \mathcal{F}(\mathbf{x}_k)\| = 0$.

To demonstrate that all stationary points of Algorithm 5.2 have $\|P_k \nabla \mathcal{F}(\mathbf{x}_k)\| = 0$ and $\|\mathcal{H}(\mathbf{x}_k)\| = 0$ we assume, for purpose of contradiction, that $\lim_{k\to\infty} \|P_k \nabla \mathcal{F}(\mathbf{x}_k)\| = \epsilon_3 > 0$. If $\|P_k \nabla \mathcal{F}(\mathbf{x}_k)\| > 0$ then there exists $\delta_{\min}(\mathbf{x}_k) > 0$ and let $\Delta_{\min}$ be the minimum of $\delta_{\min}(\mathbf{x}_k)$ for $\mathbf{x}_k \in \Omega$. Then for each k sufficiently large, $\mathcal{F}(\mathbf{x}_k) - \mathcal{F}(\mathbf{x}_k + \mathbf{s}_k) \geq \frac{1}{2}\kappa_{fcd2}\epsilon_3\Delta_{\min}$. This is a necessary contradiction since the filter acceptance criteria contains a sufficient decrease condition that requires for $\mathbf{x}_{k+1} \neq \mathbf{x}_k$ then either the constraint violation or the objective function value has been reduced by $\beta\Delta_k^2$. As $\|\mathcal{H}(\mathbf{x})\| = 0$ at all cluster points, then either $\{\Delta_k\} \to 0$ or $\{\mathcal{F}(\mathbf{x}_k)\}$ is a decreasing sequence. However, $\mathcal{F}(\mathbf{x}_k)$ is bounded from below on Ω. Therefore we must also have that $\lim_{k\to\infty} |\mathcal{F}(\mathbf{x}_k) - \mathcal{F}(\mathbf{x}_{k+1})| = 0$. Therefore at any limit point of the sequence of iterates generated by Algorithm 5.2 must have both $\|\mathcal{H}(\mathbf{x}^*)\| = 0$ and $\|P(\mathbf{x}^*)\nabla \mathcal{F}(\mathbf{x}^*)\| = 0$, i.e., that they satisfy the constraints and are first-order optimal. Therefore, the sequence of iterates generated by Algorithm 5.2 asymptotically approaches an optimal solution to (5.24). Complete details of this analysis are presented in Appendix A.4.

5.3.4 Gradient-Based AAO Trust-Region Algorithm

If all of the surrogate models used to approximate the functions in (5.24) are first-order consistent then small changes to Algorithm 5.3 are required. The cause of the changes is that in the gradient-free setting it is necessary for the size of the trust region to go to zero in order to demonstrate that the surrogate models are accurate. However, in the gradient-exploiting case we are ensured that the surrogates are locally accurate. Therefore, if all surrogates are first-order consistent, then the trust region size will be bounded away from zero and the gradient-free termination criteria will never be met. To terminate the algorithm in the gradient-based case we only require that the constraint violation is small and that the projection of the objective function gradient onto the constraint Jacobian nullspace is small. The other change we recommend is to modify the trust region update. In the gradient-free case we increase or decrease the size of the trust-region by a constant multiple because the trust-region step can be highly inaccurate. However, in the gradient-exploiting case we know that the decreasing direction of the surrogate must be the same as for the original functions. Therefore we modify the update in Algorithm 5.2 to be similar to that in [5], where the trust-region is allowed to decrease faster than by a given constant fraction. The modified gradient-free algorithm designed for purely gradient-exploiting optimization is presented as Algorithm 5.3.

5.4 Test Problems

This section presents results for the two gradient-free methods developed in this chapter, the parallel IDF formulation and the parallel AAO formulation, on two analytical test problems. The first problem, presented in Section 5.4.1, is a simple analytical problem from the literature that is nonlinear and nonconvex. The second problem, presented in Section 5.4.2, is for the design of a gearbox. Except where otherwise noted, the parameters used for the IDF algorithm are listed in Table 5.1 and the parameters used in the AAO algorithm are listed in Table 5.2. When the IDF algorithm is used in conjunction with the gradient-free algorithms of Chapters 2 and 3, the fully linear models are fit to the variables $[\tilde{\mathbf{x}}, \mathbf{t}]$, however,

Algorithm 5.3: Gradient-based AAO Trust-Region Algorithm

0: Choose, $\mathbf{x}_0, \Delta_0, \Delta_{\max}, \eta, \beta$, and tolerance ϵ. (Recommended values are given in Table 5.2, except $\beta = 0.25$ is recommended when gradients are available.) Set the initial filter to $\{(-\infty, \|\mathcal{H}(\mathbf{x}_0)\|)\}$. Create $\bar{\mathcal{F}}_0(\mathbf{x})$ and $\bar{\mathcal{H}}_0(\mathbf{x})$ to be fully linear models of $\mathcal{F}(\mathbf{x})$ and $\mathcal{H}(\mathbf{x})$ on $\|\mathbf{x} - \mathbf{x}_0\| \leq \Delta_0$, and ensure $\mathcal{H}(\mathbf{x}_0) = \bar{\mathcal{H}}_0(\mathbf{x}_0)$, $\mathcal{F}(\mathbf{x}_0) = \bar{\mathcal{F}}_0(\mathbf{x}_0)$.

1: Compute the normal step, $\mathbf{s}_k^{\perp}$, to reduce the constraint violation by solving,

$$\begin{aligned} \min_{\mathbf{s}_k^{\perp}} \quad & \frac{1}{2}\|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp})\|_2^2 \\ \text{s.t.} \quad & \|\mathbf{s}_k^{\perp}\| \leq \Delta_k. \end{aligned}$$

2: Compute the tangential step, $\mathbf{s}_k^{\parallel}$, to reduce the objective function while maintaining sufficient decrease in the constraint violation by solving,

$$\begin{aligned} \min_{\mathbf{s}_k^{\parallel}} \quad & \bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\parallel}) \\ \text{s.t.} \quad & \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\parallel})\|_2^2 \leq \|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \eta\left(\|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp})\|_2^2\right) \\ & \|\mathbf{s}_k^{\perp} + \mathbf{s}_k^{\parallel}\| \leq \Delta_k. \end{aligned}$$

3: Where $\mathbf{s}_k = \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\parallel}$, evaluate $\mathcal{F}(\mathbf{x}_k + \mathbf{s}_k)$ and $\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)$.

4: Check if the pair $(\mathcal{F}(\mathbf{x}_k+\mathbf{s}_k), \|\mathcal{H}(\mathbf{x}_k+\mathbf{s}_k)\|)$, is *non-dominated* and therefore acceptable to the filter, i.e. if

$$\mathcal{F}(\mathbf{x}_k + \mathbf{s}_k) \leq \mathcal{F}(\mathbf{x}_{\hat{k}}) - \beta\hat{\Delta}_k^2 \quad \text{or} \quad \|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\| \leq \|\mathcal{H}(\mathbf{x}_{\hat{k}})\| - \beta\hat{\Delta}_k^2\|\mathcal{H}(\mathbf{x}_{\hat{k}})\|, \ \hat{k} = 0, \ldots, k.$$

5: Update the trust region based on whether or not $\mathbf{x}_k + \mathbf{s}_k$ is acceptable,

$$(\mathbf{x}_{k+1}, \Delta_{k+1}) \quad = \quad \begin{cases} \mathbf{x}_k + \mathbf{s}_k, \ \min\{2\Delta_k, \Delta_{\max}\} & \text{if acceptable} \\ \mathbf{x}_k, \ \min\{\gamma_0\|\mathbf{s}_k\|, \gamma_0\Delta_k\} & \text{otherwise.} \end{cases}$$

5a: If $\mathbf{x}_k + \mathbf{s}_k$ is accepted, add $(\mathcal{F}(\mathbf{x}_k + \mathbf{s}_k), \|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\|)$ to the filter and remove all dominated entries.

5b: Create $\bar{\mathcal{F}}_{k+1}(\mathbf{x})$ and $\bar{\mathcal{H}}_{k+1}(\mathbf{x})$ first-order consistent at $\mathbf{x}_{k+1}$.

6: If $\|\mathcal{H}(\mathbf{x}_{k+1})\| \leq \epsilon$ and $P_k\nabla\mathcal{F}(\mathbf{x}_{k+1}) \leq \epsilon$ the algorithm has converged, otherwise go to step 1.

Constant	Description	Value
$\tilde{\mathbf{x}}_0^i$	Initial subsystem designs	$\mathbf{x}_0$
$\boldsymbol{\lambda}_0, \boldsymbol{\nu}_0^i$	Initial Lagrange multiplier estimates	$\mathbf{0}$
$\mathbf{t}_0$	Initial discipline targets	$\mathbf{0}$
ϵ	Termination tolerance	5×10^{-5}
γ	Penalty parameter growth ratio	5
η	Constraint decrease criterion	0.75
μ_0	Initial penalty parameter	0.01

Table 5.1: List of parameters used in the parallel IDF algorithm.

Constant	Description	Value
β	Filter acceptance criteria	1×10^{-4}
Δ_0	Initial trust region radius	$\max\{1, \|\mathbf{x}_0\|_\infty\}$
$\Delta_{\max}$	Maximum trust region size	$100\Delta_0$
ϵ	Termination Tolerance	1×10^{-4}
γ_0	Trust region contraction ratio	0.5
γ_1	Trust region expansion ratio	2
η	Constraint decrease requirement	0.995

Table 5.2: List of parameters used in the gradient-free AAO algorithm. The parameters used in constructing the radial basis function error models are given in Table 2.2.

the optimizations are only performed over $\tilde{\mathbf{x}}$. All results in this section are single-fidelity, i.e., $f_{\text{low}}(\mathbf{x}) = 0$.

5.4.1 Analytical Test Problem

This test problem is a nonlinear, nonconvex system design problem from the literature. The problem is originally from Sellar *et al.* [104], however, it was also used as an example in

[111],

$$
\begin{aligned}
\min_{\mathbf{x}\in\mathbb{R}^3,\mathbf{t}\in\mathbb{R}^2} \quad & \mathcal{F}(\mathbf{x},\mathbf{t}) = (x^{(1)})^2 + x^{(2)} + t^{(1)} + e^{-t^{(2)}} \\
\text{s.t. } & \mathcal{G}^1(\mathbf{x},\mathbf{t}) = 1 - \frac{t^{(1)}}{3.16} \leq 0 \\
& \mathcal{G}^2(\mathbf{x},\mathbf{t}) = \frac{t^{(2)}}{24} - 1 \leq 0 \\
& \mathcal{G}^3(\mathbf{x},\mathbf{t}) = -t^{(1)} \leq 0 \\
& 0 \leq x^{(1)} \leq 10 \\
& 0 \leq x^{(2)} \leq 10 \\
& -10 \leq x^{(3)} \leq 10 \\
& r^1(\mathbf{x},\mathbf{t}) = x^{(1)} + x^{(2)} + (x^{(3)})^2 - 0.2t^{(2)} \\
& r^2(\mathbf{x},\mathbf{t}) = x^{(2)} + x^{(3)} + \sqrt{t^{(1)}} \\
& \mathbf{r} = \mathbf{t}.
\end{aligned}
\tag{5.42}
$$

This problem is solved with four multidisciplinary optimization methods: a standard MDF approach with finite-difference gradient estimates, the IDF approach in Section 5.2 using a gradient-based optimizer to solve the discipline-level optimizations, the IDF approach in Section 5.2 using the gradient-free method of Chaper 3 to solve the discipline-level optimizations, and the parallel AAO approach in Section 5.3. Table 5.3 presents the number of iterations, discipline evaluations, and percentage of time the global minimum was found, starting from random initial points. For this problem and with randomly generated starting points within the given bounds, our IDF method always converges to a locally optimal solution. The IDF approach uses approximately twice the number of discipline-level evaluations as the MDF and AAO approaches. Compared with other decoupled approaches, this is an encouraging result as CO and CSSO require up to four times number of discipline-level evaluations as the MDF approach [111]. The AAO approach uses the fewest function evaluations to find the optimal design.

Figure 5-5 presents the number of discipline-level evaluations for the parallel IDF ap-

Approach	$r^1(\mathbf{x}, \mathbf{t})$ Evals.	$r^2(\mathbf{x}, \mathbf{t})$ Evals.	%-True Optimum
MDF	236	236	47%
Parallel IDF	578	592	50%
Parallel IDF, gradient-free	432	443	38%
Parallel AAO	230	230	64%

Table 5.3: Results of the Sellar's test problem from 50 random initial starting points drawn uniformly from the bounds given in (5.42). The bounds are enforced on all subproblem solutions. The results presented are the number of top-level iterations of the algorithm, the number of discipline level evaluations, and the percent of starting points that converged to the global optimum, $\mathbf{x}^* = [0, 0, 1.9776]^\top$ with $\mathcal{F}(\mathbf{x}, \mathbf{t}) = 3.1834$.

proach using a gradient-based optimizer, the parallel IDF approach using the gradient-free optimizer of Chapter 3, and the MDF approach. The decoupled IDF approach clearly uses more function evaluations than the MDF approach. However, all of the discipline-level evaluations for the MDF approach must be conducted in serial, whereas, the cost for an iteration of the IDF approach is the cost of the most expensive discipline-level optimization, since both optimizations are performed in parallel. The right side of the figure considers the scenario where evaluations of a discipline are conducted in serial, but evaluations of different disciplines can be conducted in parallel. In this scenario, where the parallelization of Algorithm 5.1 is taken into account, the combination of Algorithm 5.1 and the gradient-free optimization of Chapter 3 requires the time equivalent to approximately 10% more discipline-level evaluations than the MDF approach.

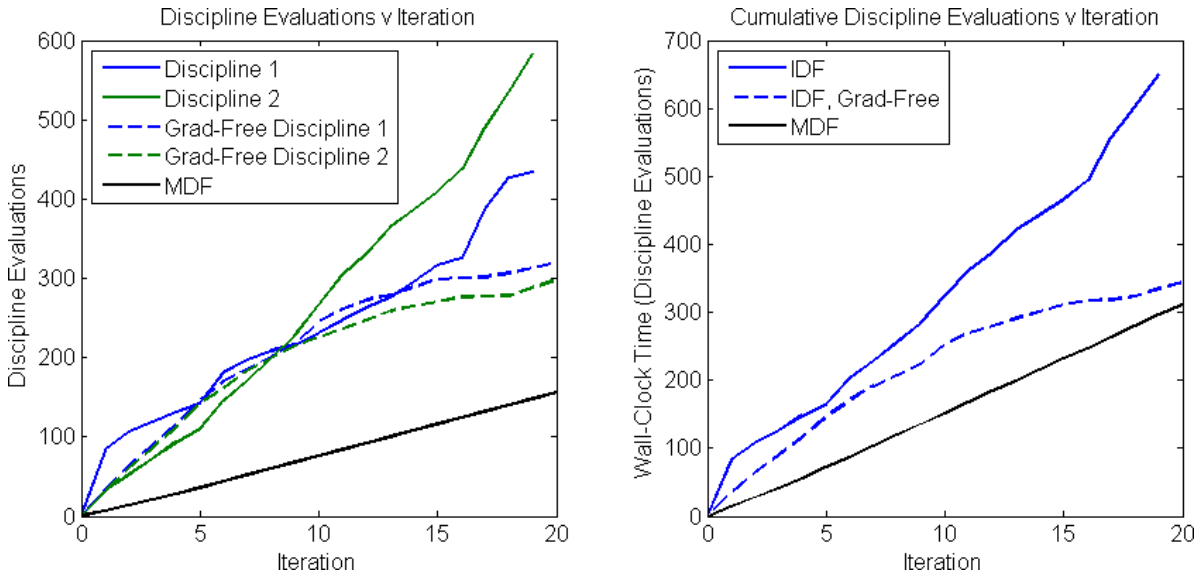


Figure 5-5: Left, function evaluations versus iteration for the parallel IDF algorithm on Sellar's test problem starting from $\mathbf{x}_0 = [5, 2, 1]^\top$. Right, the parallelized scaling of the IDF algorithm starting from $\mathbf{x}_0 = [5, 2, 1]^\top$. The results show that when the parallelizability of the IDF algorithm is compared with the MDF approach, which evaluates the disciplines in serial, the IDF algorithm requires only the equivalent of approximately 10% more discipline-level evaluations.

5.4.2 Gearbox Design

This section presents optimization results for the design of a gearbox, commonly known as Golinski's speed reducer [47]. The AAO formulation considered here is,[4]

$$
\begin{aligned}
\min_{\mathbf{x}\in\mathbb{R}^7,\mathbf{p}\in\mathbb{R}^{11}} \quad & \mathcal{F}(\mathbf{x}) = 0.7854x^{(1)}(x^{(2)})^2(3.3333(x^{(3)})^2 + 14.9334x^{(3)} - 43.0934) \\
& \quad -1.5709x^{(1)}((x^{(6)})^2 + (x^{(7)})^2) + 7.477((x^{(6)})^3 + (x^{(7)})^3) \\
& \quad +0.7854(x^{(4)}(x^{(6)})^2 + x^{(5)}(x^{(7)})^2) \\
\text{s.t. } & \mathcal{H}^1(\mathbf{x}) = 27(x^{(1)})^{-1}(x^{(2)})^{-2}(x^{(3)})^{-1} - 1 + p^{(1)} = 0 \\
& \mathcal{H}^2(\mathbf{x},\mathbf{p}) = 397.5(x^{(1)})^{-1}(x^{(2)})^{-2}(x^{(3)})^{-2} - 1 + p^{(2)} = 0 \\
& \mathcal{H}^3(\mathbf{x},\mathbf{p}) = 1.93(x^{(2)})^{-1}(x^{(3)})^{-1}(x^{(4)})^3(x^{(6)})^{-4} - 1 + p^{(3)} = 0 \\
& \mathcal{H}^4(\mathbf{x},\mathbf{p}) = 1.93(x^{(2)})^{-1}(x^{(3)})^{-1}(x^{(5)})^3(x^{(7)})^{-4} - 1 + p^{(4)} = 0 \\
& \mathcal{H}^5(\mathbf{x},\mathbf{p}) = \left[\left(745x^{(4)}(x^{(2)})^{-1}(x^{(3)})^{-1}\right)^2 + 16.9\times 10^6\right]^{\frac{1}{2}} / \left[110.0(x^{(6)})^3\right] - 1 + p^{(5)} = 0 \\
& \mathcal{H}^6(\mathbf{x},\mathbf{p}) = \left[\left(745x^{(5)}(x^{(2)})^{-1}(x^{(3)})^{-1}\right)^2 + 157.5\times 10^6\right]^{\frac{1}{2}} / \left[85.0(x^{(7)})^3\right] - 1 + p^{(6)} = 0 \\
& \mathcal{H}^7(\mathbf{x},\mathbf{p}) = x^{(2)}x^{(3)}/40 - 1 + p^{(7)} = 0 \\
& \mathcal{H}^8(\mathbf{x},\mathbf{p}) = 5x^{(2)}/x^{(1)} - 1 + p^{(8)} = 0 \\
& \mathcal{H}^9(\mathbf{x},\mathbf{p}) = x^{(1)}/12x^{(2)} - 1 + p^{(9)} = 0 \\
& \mathcal{H}^{10}(\mathbf{x},\mathbf{p}) = \left(1.5x^{(6)} + 1.9\right)(x^{(4)})^{-1} - 1 + p^{(10)} = 0 \\
& \mathcal{H}^{11}(\mathbf{x},\mathbf{p}) = \left(1.1x^{(7)} + 1.9\right)(x^{(5)})^{-1} - 1 + p^{(11)} = 0 \\
& \mathcal{H}^{12}(\mathbf{x},\mathbf{p}) = x^{(3)} - \lfloor x^{(3)} + .5\rfloor)^2 = 0 \\
& [2.6, 0.7, 17, 7.3, 2.9, 5.0]^\top \leq \mathbf{x}^\top \leq [3.6, 0.8, 28, 8.3, 3.9, 5.5]^\top \\
& \mathbf{0} \leq \mathbf{p}.
\end{aligned}
\tag{5.43}
$$

[4]The constraint that $x^{(3)}$ is an integer in [47] is replaced with the constraint $(x^{(3)} - \lfloor x^{(3)} + .5\rfloor)^2 = 0$.

Following [65], in multidisciplinary form the system-level optimization problem is,

$$\begin{aligned}
&\min_{\mathbf{x}\in\mathbb{R}^7,\mathbf{t}\in\mathbb{R}^3} \quad \mathcal{F}(\mathbf{t}) = 1000(t^{(1)} + t^{(2)} + t^{(3)}) \\
&\text{s.t. } \mathcal{G}^1(\mathbf{x}) = x^{(2)}x^{(3)} - 40 \leq 0 \\
&\qquad [0.7, 17, 7.3, 7.3]^\top \leq [x^{(2)}, x^{(3)}, x^{(4)}, x^{(5)}]^\top \leq [0.8, 28, 8.3, 8.3]^\top \\
&\qquad \mathbf{t} = \mathbf{r} \\
&\qquad \mathbf{x} = \tilde{\mathbf{x}}^i, \quad i = 1, 2, 3.
\end{aligned} \tag{5.44}$$

The first discipline problem is,

$$\begin{aligned}
r^1(\tilde{\mathbf{x}}) \;=\; &\min_{\tilde{\mathbf{x}}\in\mathbb{R}^7} \; \frac{1}{1000}\left[0.7854\tilde{x}^{(1)}(\tilde{x}^{(2)})^2\left(3.3333(\tilde{x}^{(3)})^2 + 14.9334\tilde{x}^{(3)} - 43.0934\right)\right] \\
&\text{s.t. } 27(\tilde{x}^{(2)})^{-2}(\tilde{x}^{(3)})^{-1} - \tilde{x}^{(1)} \leq 0 \\
&\qquad 397.5(\tilde{x}^{(2)})^{-2}(\tilde{x}^{(3)})^{-2} - \tilde{x}^{(1)} \leq 0 \\
&\qquad 5\tilde{x}^{(2)} - \tilde{x}^{(1)} \leq 0 \\
&\qquad \tilde{x}^{(1)} - 12\tilde{x}^{(2)} \leq 0 \\
&\qquad 2.6 \leq \tilde{x}^{(1)} \leq 3.6.
\end{aligned} \tag{5.45}$$

The second discipline problem is,

$$\begin{aligned}
r^2(\tilde{\mathbf{x}}) \;=\; &\min_{\tilde{\mathbf{x}}\in\mathbb{R}^7} \; \frac{1}{1000}\left[-1.508\tilde{x}^{(1)}(\tilde{x}^{(6)})^2 + 7.477(\tilde{x}^{(6)})^3 + 0.7854\tilde{x}^{(4)}(\tilde{x}^{(6)})^2\right] \\
&\text{s.t. } \left(\frac{1}{110}\left[\left(745x^{(4)}(x^{(2)})^{-1}(x^{(3)})^{-1}\right)^2 + 16.9\times 10^6\right]^{\frac{1}{2}}\right)^{1/3} - \tilde{x}^{(6)} \leq 0 \\
&\qquad \left(1.93(\tilde{x}^{(2)})^{-1}(\tilde{x}^{(3)})^{-1}(\tilde{x}^{(4)})^3\right)^{0.25} - \tilde{x}^{(6)} \leq 0 \\
&\qquad \tilde{x}^{(6)} - (\tilde{x}^{(4)} - 1.9)/1.5 \leq 0 \\
&\qquad 2.9 \leq \tilde{x}^{(6)} \leq 3.9.
\end{aligned} \tag{5.46}$$

The third discipline problem is,

$$
\begin{aligned}
r^3(\tilde{\mathbf{x}}) \;=\; & \min_{\tilde{\mathbf{x}}\in\mathbb{R}^7} \; \frac{1}{1000}\left[-1.508\tilde{x}^{(1)}(\tilde{x}^{(7)})^2 + 7.477(\tilde{x}^{(7)})^3 + 0.7854\tilde{x}^{(5)}(\tilde{x}^{(7)})^2\right] \qquad (5.47)\\
& \text{s.t.} \; \left(\frac{1}{85}\left[\left(745x^{(5)}(x^{(2)})^{-1}(x^{(3)})^{-1}\right)^2 + 157.5\times 10^6\right]^{\frac{1}{2}}\right)^{1/3} - \tilde{x}^{(7)} \le 0\\
& \left(1.93(\tilde{x}^{(2)})^{-1}(\tilde{x}^{(3)})^{-1}(\tilde{x}^{(5)})^3\right)^{0.25} - \tilde{x}^{(7)} \le 0\\
& \tilde{x}^{(7)} - (\tilde{x}^{(5)} - 1.9)/1.1 \le 0\\
& 5.0 \le \tilde{x}^{(7)} \le 5.5.
\end{aligned}
$$

Table 5.4 presents the number of discipline evaluations and the percentage of time the global minimum was found starting from random initial points. Results are included for SQP [76] using finite-difference gradient estimates and the parallel AAO approach in Section 5.3 which both solve the problem in (5.43). In addition, results are included for a standard MDF approach, the parallel IDF approach in Section 5.2 using a gradient-based optimizer to solve the discipline-level optimizations, the parallel IDF approach in Section 5.2 using the gradient-free method of Chapter 3 to solve the discipline-level optimizations, and the parallel IDF approach in Section 5.2 with unscaled discipline objectives, which all use the decoupled problem in (5.44)–(5.47). The unscaled discipline objectives correspond to removing the factor of 1,000 from the system and discipline-level problems and is included to demonstrate the effect of having a multiple order of magnitude difference in scaling between the discipline responses and design variables. The results show that the parallel AAO approach uses the fewest function evaluations of all the methods tested. In addition, compared with SQP, the parallel AAO approach finds the global minimum more frequently. This problem has multiple local solutions, many of which are caused by the constraint $x^{(3)} - \lfloor x^{(3)} + .5\rfloor)^2 = 0$. The gradient-free method uses response surfaces which are less sensitive to local sensitivity information than the more local gradient-based methods. The parallel IDF approach uses more function evaluations than the other methods, and is clearly sensitive to the scaling, since the unscaled version of the problem requires about 50% more evaluations. We observe that the reuse of previous discipline evaluations associated with the

gradient-free method of Chapter 3 reduces the number of discipline evaluations by approximately 60%. If discipline were conducted in parallel than the combination of the parallel IDF approach and the gradient-free method of Chapter 3 would require less wall-clock time then the MDF method.

Approach	$r^1(\mathbf{x},\mathbf{t})$ Evals.	$r^2(\mathbf{x},\mathbf{t})$ Evals.	$r^3(\mathbf{x},\mathbf{t})$ Evals.	%-True Optimum
SQP	337	337	337	2%
Parallel AAO	294	294	294	60%
MDF	416	416	416	96%
Parallel IDF	2,092	2,550	2,056	100%
Parallel IDF, gradient-free	810	969	953	100%
Parallel IDF, unscaled	3,488	3,306	3,241	94%

Table 5.4: Results of optimizing Golinski's speed reducer from 50 random initial starting points drawn uniformly from the given bounds. The results presented are the number of discipline level evaluations and the percent of starting points that converged to the global optimum, $\mathbf{x}^* = [3.5, 0.7, 17, 7.3, 7.71532, 3.3502, 5.2867]^\top$ with $\mathcal{F}(\mathbf{x},\mathbf{t}) = 2994.35$. The unscaled solution corresponds to removing the factor of 1,000 in the discipline responses.

5.5 Discussion

This chapter has presented two methods for multidisciplinary optimization that enable multifidelity optimization and parallelization without the need to compute a high-fidelity gradient. However, if gradients are available, both methods are also able to exploit that information to speed finding an optimal design. The IDF approach has been shown to have good parallel scaling in terms of discipline-level function evaluations when coupled with multifidelity optimization and reuse of previous design evaluation information. However, the method is sensitive to the scaling between the design variables and discipline coupling variables. There are three solutions to addressing this sensitivity: appropriately scaling all variables, which is possibly quite difficult *a priori*, increasing the penalty parameter slowly ($\gamma \sim 1$), and using a more robust IDF formulation. Developing a more robust framework is an avenue for future research, perhaps by coupling an augmented Lagrangian trust region framework [30, 31] with the parallelized structure of the method presented in this chapter a method with global convergence properties could be obtained.

The gradient-free AAO optimization method developed in this chapter performs comparably with SQP for many problems, even without the addition of a low-fidelity model. The flexibility and observed performance of this method suggest many potential benefits for the multidisciplinary design optimization community. Table 5.5 provides a summary of the advantages and disadvantages of the algorithms developed in this chapter to optimize a multidisciplinary system. The metrics included are: whether or not the algorithm requires sensitivity information, the robustness of the algorithm to disciplines failing to return a result, the robustness of the algorithm to the initial design, the multifidelity and parallelization capabilities, the flexibly of the algorithms to effectively utilize all available processors, the computational overhead of the methods, the expected number of function calls with and without available sensitivity information, the ability of the algorithms to optimize systems with both a large and small number of discipline-coupling variables, and the difficulty to implement the algorithms on new problems.

	SQP/MDF	IDF Formulation	Parallel Evaluations
Sensitivity information	Required	Not required	Not required
Robustness (evaluation failure)	Middle	Worst	Best
Robustness (initial design)	Best	Worst	Best
Multifidelity capabilities	System level only	All levels	All levels
Parallelization	System level only	All levels	All levels
Processor utilization	Flexible	Flexible	Most flexible
Method overhead	Low	Low	Surrogate models
Function calls (gradients)	Least	Most	Middle
Function calls (no gradients)	N/A	Most	Least
High-bandwidth coupling	Fastest	Slowest	Fast or slow
Low-bandwidth coupling	Fast	Fast	Fast
Implementation Difficulty	Easy	Moderate	Easy

Table 5.5: A comparison of using sequential quadratic programming and an MDF formulation with the MDO methods developed in this chapter to optimize a multidisciplinary system.

Chapter 6

Case Studies in Aerostructural Optimization

This chapter demonstrates the multidisciplinary and multifidelity optimization methods developed in Chapters 2, 3, 4 and 5 on two aerostructural optimization problems. Aerostructural optimization is an especially difficult problem for multidisciplinary optimization strategies because of the high degree of coupling between the aerodynamics and structural disciplines. In classical aircraft design, an aerodynamics group designs the shape of the aerodynamic surfaces and a structures group designs the internal support structure in an attempt to support the aerodynamic forces and keep the weight low. The challenge in decoupling these two disciplines and optimizing the aerodynamic shape and structure separately is that the coupling is a pressure field and a displacement field. These are continuous fields which are represented by discrete approximations. However, these approximations are still high dimensional. For instance in a computational design framework, the exchange between the disciplines is the pressure at every aerodynamic node on the surface and the deflection and rotations at every structural node on the surface.

With these challenges in mind, we present two case studies in aerostructural optimization. The first case study represents the conventional approach taken in industry. The aerodynamic analysis method, Panair, and the structural analysis method, Nastran, are represen-

tative industrial tools. These tools offer numerous optimization difficulties, single-precision[1], frequent failures to return a solution, and issues finding a coupled aerostructural solution. The single-precision and frequent failure to return a solution suggest that gradient-based optimization may not work for this optimization problem and that a gradient-free strategy may be needed for robustness. The second case study uses multifidelity optimization on state-of-the-art aerostructural analysis software. The analysis routines solve a coupled aerostructural system using a tailored parallel solution scheme and provide an adjoint-based gradient estimate for use in optimization.

6.1 Gradient-free Aerostructural Optimization

This wing design problem is to find an optimal wing typical of a small unmanned aerial vehicle designed for long range. The objective is to find the wing with the best range factor, minimum weight divided by the lift-to-drag ratio. The requirements are that at the design point, the maximum stress in the wing structure is less than the material yield stress and the wing geometry is sensible, i.e. positive thickness and stringers do not penetrate skin panels. There are 26 design variables for the wing. The aerodynamic design variables are the wing angle of attack and three parameters representing NACA 4-series airfoils, the maximum camber of the airfoil, the location of the maximum camber, and the maximum thickness to chord ratio. These variables are all treated as continuous variables, but the airfoils are constructed in the same way as the NACA 4-series. For example, an airfoil with design variables 0,0,12 is the same as a NACA 0012 airfoil, however, we allow for an airfoil that is 12.25% thick. There are 22 structural design variables, the skin thickness, spanwise material thickness taper ratio, the chordwise location of two spars, the material thickness of the spars, the spanwise location of four ribs, the material thickness of the ribs, the chordwise location of eight stringers (upper and lower surface total), and four variables describing the shape of the stringers. The material thickness is constant between each of the ribs, however, for each spanwise section the material thickness is reduced according to the material thickness taper

[1] it is possible to use Nastran in double-precision mode

ratio. So the thickness of the skin panels, spars and ribs all decrease approximately linearly spanwise. The stringers are hat sections, with the material thickness, flange, width, and height for design variables. A schematic of the wing cross-section is shown in Figure 6-1, the figure shows the skin, spars, and stringers, the ribs fill the entire cross-sectional area inside of the skin and are considered welded to the spars along rib-spar intersections.

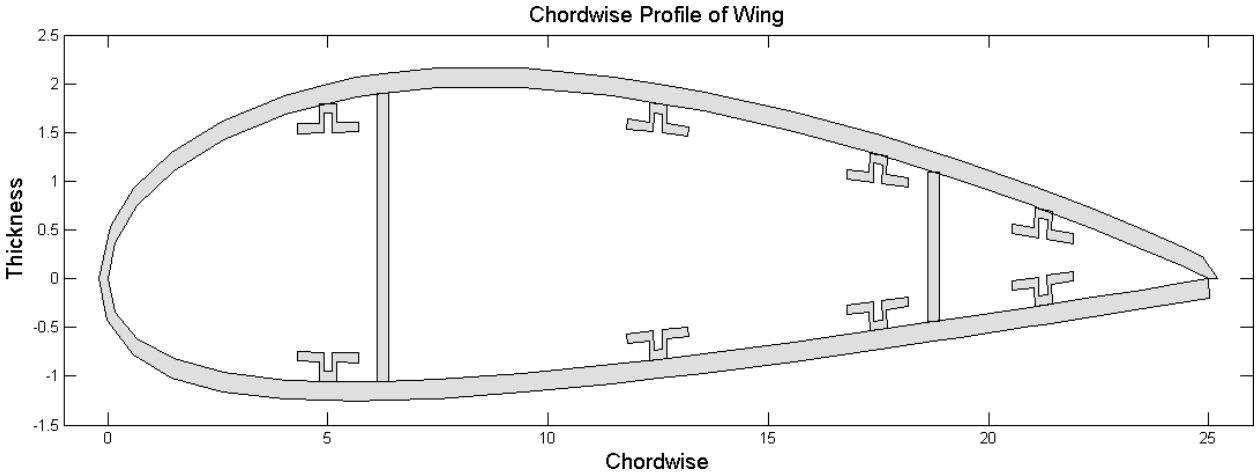


Figure 6-1: Wing cross-section showing the constant thickness skin, the two spars, and the four hat stringers. The ribs are not shown, but they fill the entire cross-section of the wing inside of the skin panels.

The high-fidelity aerodynamic analysis tools are Panair [99] and Friction [74]. Panair computes the flow field around the wing assuming the flow is inviscid. Friction computes the skin friction drag on the wing by estimating the skin friction coefficient and form factor. The structural model in Nastran [110] applies the calculated pressures from Panair to the skin panel elements and the skin friction drag force from Friction to the trailing edge nodes. The two optimization frameworks presented in Chapter 5 are demonstrated on this problem. First the parallel IDF approach of Section 5.2, and second, the parallel filter approach of Section 5.3 is used to solve a MDF formulation of the aerostructural problem.

6.1.1 IDF Framework

Aerostructural optimization is a challenging problem with which to use an IDF formulation because of the high-dimensionality of the coupling between the disciplines. To perform

structural optimization and minimize the wing weight, the structural analysis requires the pressure field around the wing. Similarly, to perform aerodynamic optimization and maximize the lift-to-drag ratio the aerodynamic analysis requires the deformed shape of the wing. Even in discretized form this coupling is likely too high-dimensional for optimization to be tractable, so the first step of the aerostructural optimization using an IDF formulation is to develop a reduced-dimensional coupling. In the following we formulate a low-dimensional C_p distribution representation, then a low-dimensional deflection distribution representation, and finally formulate this aerostructural optimization in an IDF framework.

Low-dimensional C_p Distribution

To represent the pressure field around the wing, the C_p distribution is approximated by splines. The wing root C_p distribution is approximated with two piecewise cubic Hermite interpolating polynomials, one for the upper surface and another for the lower surface. The splines each have 8 control points, but the trailing edge control point is made a stagnation point, $C_p = 1$. The spanwise variation in the C_p distribution is approximated with a smoothing cubic spline. The locations of the root chord control points, with the exception of the trailing edge, are scaled according to the spline. The spanwise scaling spline has 4 control points, but the root control point is fixed at 1 so only 3 are parameters. The complete C_p distribution around the wing is represented with 17 parameters. The root-mean-square error of the of the actual C_p value on the wing and this reduced parameterization is about 0.02 for an assortment of airfoils and twist distributions. Figure 6-2 compares the actual C_p distribution for a typical wing with the approximate C_p distribution generated by this parameterization.

Low-dimensional Deflection Distribution

To represent the deformed shape of the wing, the deflections of the leading and trailing edges are approximated by cubic splines. This parameterization assumes the wing contains enough ribs such that the cross-sectional shape of the wing does not deform significantly. The splines each have 4 control points, though the root control points are fixed to have zero deflection.

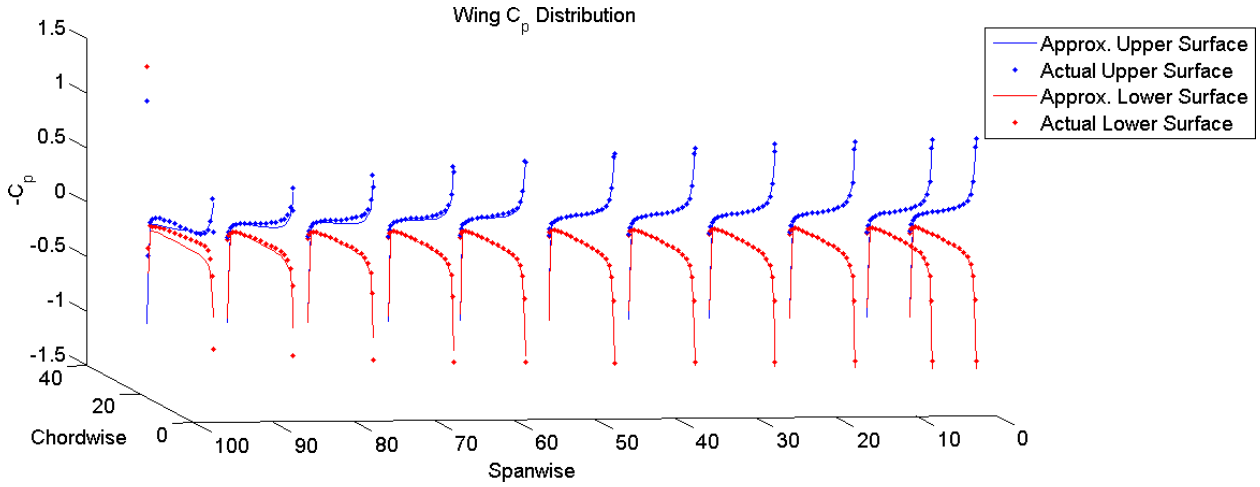


Figure 6-2: Comparison of the actual and low-dimensional approximation of the C_p distribution for a typical wing. The approximate C_p distribution does not accurately capture the wingtip effects.

Therefore the wing deflection is parameterized by a total of 18 parameters, 3 control points for three spatial dimensions for the leading and trailing edges. The root-mean-square error between the actual deflection of the wing and the reduced dimensional approximation is less than 0.5% of the span. Figure 6-3 compares the actual deformed shape of a typical wing with the approximately deformed wing generated with this reduced-dimension approximation.

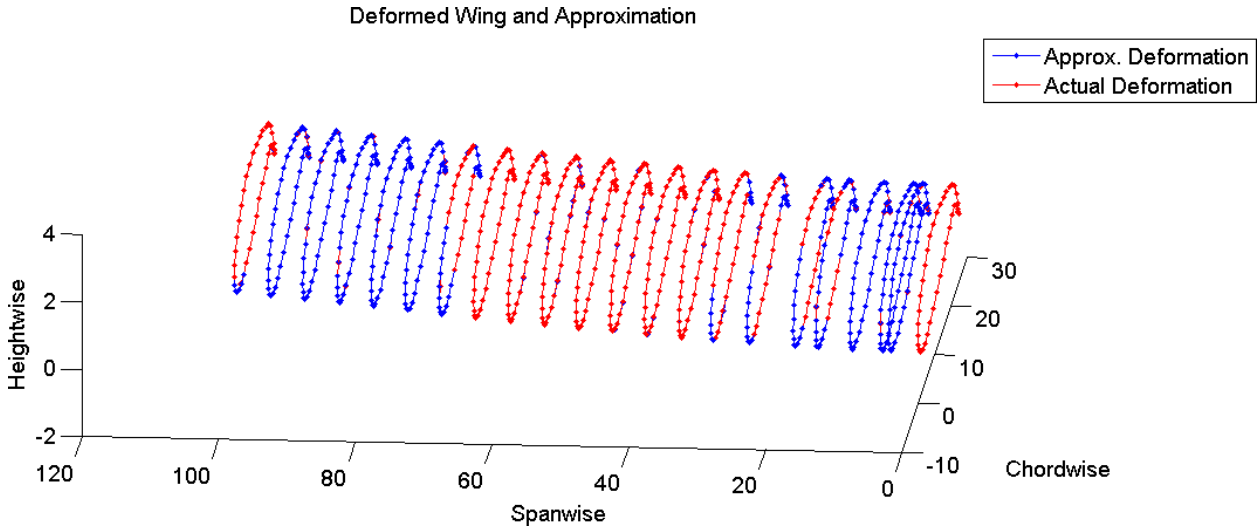


Figure 6-3: Comparison of the actual and low-dimensional approximation of the deflection for a typical wing.

Aerostructural IDF Optimization Problem

The aerostructural optimization written as an IDF optimization problem using the reduced dimensional coupling is

$$\min_{\mathbf{x}\in\mathbb{R}^{26},\mathbf{t}\in\mathbb{R}^{35}} \quad w(\mathbf{x})/L\!/D(\mathbf{x},\mathbf{t}) \tag{6.1a}$$

$$\text{s.t. } \max\{\sigma_{vonMises}(\mathbf{x},\mathbf{t}) - \sigma_{yield}, 0\} = 0 \tag{6.1b}$$

$$\mathbf{r}_{C_p}(\tilde{\mathbf{x}}^1, \mathbf{t}_{Def}) = \mathbf{t}_{C_p} \tag{6.1c}$$

$$\mathbf{r}_{Def}(\tilde{\mathbf{x}}^2, \mathbf{t}_{C_p}) = \mathbf{t}_{Def} \tag{6.1d}$$

$$\mathbf{x} = \tilde{\mathbf{x}}^1 = \tilde{\mathbf{x}}^2 \tag{6.1e}$$

where

$$\tilde{\mathbf{x}}^1 = \arg\min_{\tilde{\mathbf{x}}\in\mathbb{R}^{26}} \frac{1}{2}\|\mathbf{r}_{C_p}(\tilde{\mathbf{x}}, \mathbf{t}_{Def}) - \mathbf{t}_{C_p}\|_2^2 \tag{6.1f}$$

$$\tilde{\mathbf{x}}^2 = \arg\min_{\tilde{\mathbf{x}}\in\mathbb{R}^{26}} \frac{1}{2}\|\mathbf{r}_{Def}(\tilde{\mathbf{x}}, \mathbf{t}_{C_p}) - \mathbf{t}_{Def}\|_2^2. \tag{6.1g}$$

The objective is to find the wing with the minimum weight divided by lift-to-drag ratio, $L\!/D(\mathbf{x},\mathbf{t})$, where the maximum von Mises stress in the wing, $\sigma_{vonMises}(\mathbf{x},\mathbf{t})$, is less than the material yield stress, σ_{yield}. The aerodynamics discipline, (6.1f), uses Panair for an inverse design problem: given a fixed deflection, minimize the deviation of the actual C_p distribution over the wing, $\mathbf{r}_{C_p}(\tilde{\mathbf{x}}^1, \mathbf{t}_{Def})$, from the target distribution, $\mathbf{t}_{C_p}$. Similarly, the structural discipline, (6.1g), minimizes the deviation of the actual deflection of the wing, $\mathbf{r}_{Def}(\tilde{\mathbf{x}}^2, \mathbf{t}_{C_p})$, from the target deflection distribution, $\mathbf{t}_{Def}$, at a fixed C_p distribution. Since the stress within the wing is only computed in Nastran and not at the system level we enforce the maximum stress constraint with a quadratic penalty function at the discipline level in (6.1g) (the penalty multiplier used is unity).

The aerostructural IDF optimization problem, (6.1), is solved using Algorithm 5.1 with the parameters listed in Table 5.1. The system-level optimization problems are solved with sequential quadratic programming [76]. The discipline-level optimization problems are solved using the gradient-free method presented in Algorithm 2.1 (with a minor change in the con-

struction of fully linear models discussed in the following section). A gradient-free optimizer is needed because the poor behaviors of the discipline-level simulations cause inaccuracies in finite-difference gradient estimates. The system-level optimization problems are both smooth and well-behaved because the weight is computed analytically and the lift-to-drag ratio as a function of the target C_p distribution is also computed analytically.

6.1.2 MDF Framework

The MDF framework for this problem is setup so that all expensive computations and numerical challenges are in the constraints. The optimization problem is

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^{26}, t_{LD}\in\mathbb{R}, t_w\in\mathbb{R}} \quad & t_w/t_{LD} \\ \text{s.t.} \quad & t_w - w(\mathbf{x}) = 0 \\ & t_{LD} - LD(\mathbf{x}) = 0 \\ & \max\{\sigma_{yield}, \sigma_{vonMises}(\mathbf{x})\} - \sigma_{yield} = 0 \\ & h(\mathbf{x}) = 0, \end{aligned} \tag{6.2}$$

where the objective is to minimize a target weight, t_w, divided by a target lift-to-drag ratio, t_{LD}, subject to constraints that the structural weight is equal to the target weight, the lift-to-drag ratio is equal to the target lift-to-drag ratio, the maximum stress in the structure is less than the yield stress, and some simple geometric constraints, represented as $h(\mathbf{x}) = 0$, are satisfied. Each constraint evaluation requires an iterative solve between Panair and Nastran to find the lift-to-drag ratio of the deformed wing and the maximum von Mises stress. The constraint evaluation only returns a value approximately 81% of the time due to (1) Nastran failures, (2) Panair failures, and (3) aerostructural solver convergence failures. The aerostructural solver is a Gauss-Seidel iteration with a successive over-relaxation factor of 0.9. Moreover, the output of Panair only has six decimal places and the input file for Nastran has five decimal places so the value of the constraints and objectives are highly non-smooth. For example, Figure 6-4 shows finite-difference estimates of the directional

derivative, $\left[f(\mathbf{x}_0 + \delta_x \mathbf{p}/\|\mathbf{p}\|) - f(\mathbf{x}_0)\right]/\delta_x$, for various step sizes, δ_x, in random directions, $\mathbf{p}$, from four random initial designs, $\mathbf{x}_0$. We observe that the directional derivative for

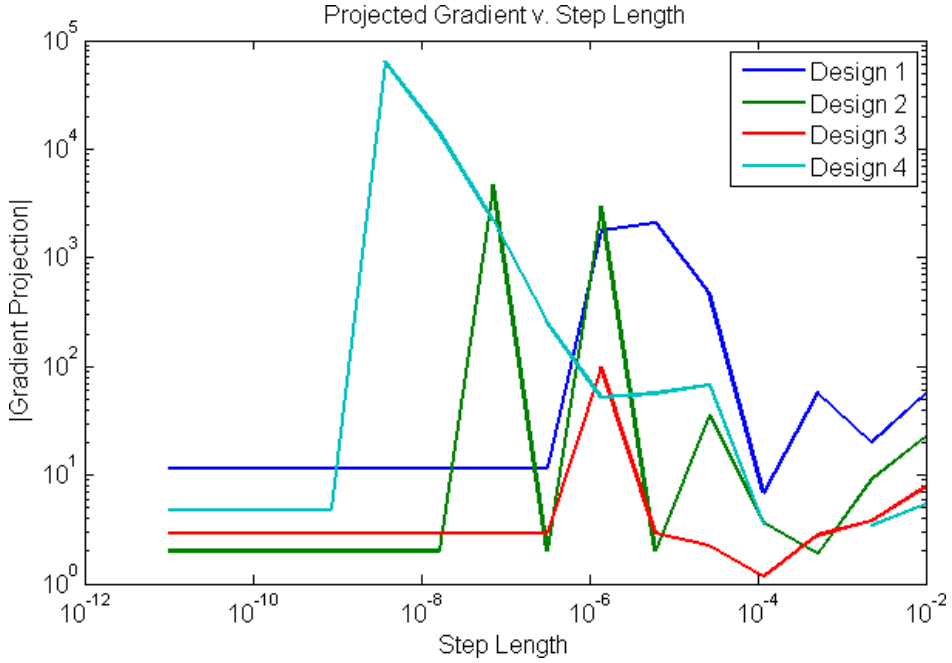


Figure 6-4: Projected gradient in random direction for four random initial wing designs for weight divided by the lift-to-drag ratio. Missing points on the plot are caused by evaluation failures. The flat portion of the projected gradient are when changes in the lift-to-drag ratio are below the precision of Panair and only the weight of the wing is changing.

weight divided by the lift-to-drag ratio varies by over four orders of magnitude for typical finite-difference step sizes. The derivative estimate becomes flat when the change in the lift-to-drag ratio is below the precision of Panair and only the weight, computed analytically, is changing. The poor behavior of the constraints mean that a gradient-based optimization method cannot be used. Therefore, we will solve this aerostructural optimization problem using Algorithm 5.2, which enables gradient-free optimization. However, due to the high failure rate of the evaluations, we need some small changes to the method of building fully linear models, Algorithm 2.2, and to the filter and trust region updates in Algorithm 5.2. We first discuss the modifications required to build fully linear models in the presence of evaluation failures, and then we discuss the modifications to the update steps in Algorithm 5.2.

Fully Linear Model Construction with Failures

The algorithm used to create fully linear models in the previous chapters, Algorithm 2.2, assumed that the function being approximated returns a result every time. The flexibility in only using a fully linear calibration scheme enables a calibrated surrogate model to be created even if the function occasionally fails to return a result, because the calibration points can be relocated. The only modifications required to Algorithm 2.2 are in step 2f where the function is evaluated at locations in the nullspace of the span of vectors in $\mathcal{Y}$. This step needs to accommodate potential evaluation failures. When we encounter an evaluation failure we select a basis for the nullspace of the span of vectors in $\mathcal{Y}$. For each basis vector we then draw a random number, $t \in [0.25, 0.75]$ and evaluate the function at the points,

$$\mathbf{x}_k + (-1)^{l-1} t^{\lceil l/2 \rceil - 1} \Delta_k \hat{\mathbf{p}}, \quad l = 1, 2, 3, \ldots \tag{6.3}$$

where $\hat{\mathbf{p}}$ are the unit directions in the nullspace, until a successful evaluation is found. This operation starts at the edge of the trust region and progressively searches closer to the current iterate (which must have been successful) on the right and left sides until another successful calibration point is found. For example, if $t = \frac{1}{2}$, the sequence $(-1)^{l-1} t^{\lceil l/2 \rceil - 1}$ evaluates to $1, -1, \frac{1}{2}, -\frac{1}{2}, \frac{1}{4}, \ldots$.

Trust Region Updating with Failures

Three challenges with Algorithm 5.2 are apparent in the presence of evaluation failures, (1) we need the filter acceptance criteria to verify that $\mathbf{x}_k + \mathbf{s}_k$ returned a valid result, (2) it is significantly cheaper to attempt evaluating the function at a few new points as opposed to simply rejecting $\mathbf{s}_k$ and constructing a new fully linear model, and (3) the assumption that the constraint Jacobian has full rank, is no longer valid and robustness should be addressed. To mitigate challenges (1) and (2) we assume the step $\mathbf{s}_k$ is an improving direction and evaluate the high-fidelity functions at locations along $\mathbf{s}_k$ up to a set number of times. Our algorithm draws a random number $t \in [0.5, 1)$ and evaluates $\mathcal{F}(\mathbf{x}_k + t^{l-1}\mathbf{s}_k)$ and $\mathcal{H}(\mathbf{x}_k + t^{l-1}\mathbf{s}_k)$ for $l = 1, 2, 3, \ldots$ until either $l_{\max}$ is reached or a successful evaluation is found. However, if

the functions are evaluated $l_{\max}$ times without success, then $\mathbf{s}_k$ is rejected.

To address the faulty assumption that the constraint Jacobian has full rank we limit the number of iterates contained on the filter. Without this assumption it is possible for Algorithm 5.2 to terminate at a point that has nonzero constraint violation. To ensure the constraint violation is generally decreasing, we impose a decreasing upper limit for the constraint violation. We define $\hat{k}_{\max}$ to be a user-set parameter for the maximum number of points on the filter, and establish an upper limit by treating the $\hat{k}_{\max}^{th}$ point on the filter as the initial iterate, i.e., by removing all iterates on the filter with constraint violation higher than the $\hat{k}_{\max}^{th}$ point and adding $(-\infty, \|\mathcal{H}(\mathbf{x}_{\hat{k}_{\max}})\|)$ to the filter. (The presence of $(-\infty, \|\mathcal{H}(\mathbf{x}_{\hat{k}_{\max}})\|)$ on the filter prevents the filter from accepting any iterates with constraint violation larger than $\|\mathcal{H}(\mathbf{x}_{\hat{k}_{\max}})\|$.) We only truncate the filter if the constraint violation at $\mathbf{x}_k$ is less than the constraint violation at $\mathbf{x}_{\hat{k}_{\max}}$, otherwise the filter acceptance criteria will block desirable future iterates. This decreasing upper bound for the constraint violation helps to drive Algorithm 5.2 towards feasibility even when the surrogate Jacobian does not have full rank throughout the design space. The parameter values used in Algorithm 5.2 are presented in Table 5.2. We also use $l_{\max} = 2$ and $\hat{k}_{\max} = 3$.

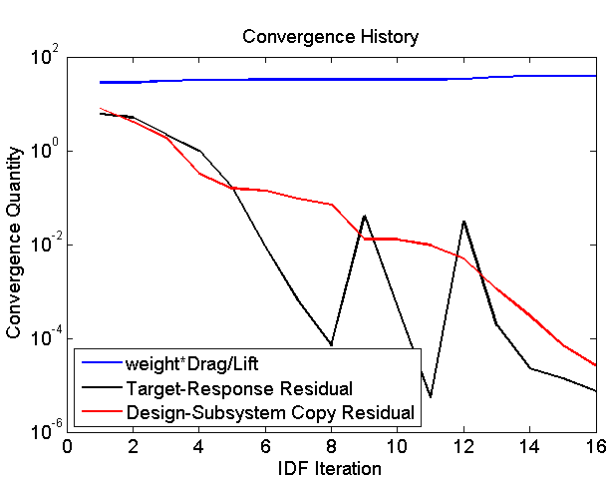
6.1.3 Results

Both multidisicplinary optimization methods successfully optimized the wing despite the numerous numerical challenges. In contrast, none of the constrained optimization solvers in [76] (interior point, SQP, or active-set) were able to solve this problem. No global optimization methods were attempted due to the number of design variables and computational expense of the simulations, Panair and Nastran evaluations take on average approximately 30 seconds and 25 seconds to solve, respectively. Many locally optimal designs were found, which is expected given the design space has islands of feasibility. The results presented below are the average number of simulations and MDAs for Algorithms 5.1 and 5.2 from random initial designs. Algorithm 5.1 is started from a multidisciplinary feasible design, though feasibility is lost during the first iteration of the algorithm and regained prior to termination. As presented in Table 6.1, Algorithm 5.1 took on average 9,100 structural anal-

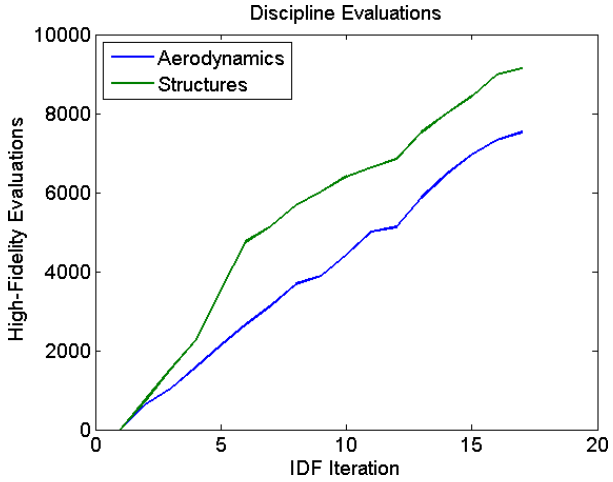
yses and 7,700 aerodynamic analyses to find an optimal wing. Since both the aerodynamic and structural analyses are conducted in parallel, the solution time scales with the number of aerodynamic analyses conducted. Figure 6-5(a) shows a representative convergence history of Algorithm 5.1, and Figure 6-5(b) shows a representative plot of the number of discipline-level evaluations needed. No low-fidelity analyses were used for this test case.

Case	Algorithm	Low-Fidelity Model	MDAs	Nastran Evals.	Panair Evals.
0	Parallel IDF	None	1	9,073	7,688
i	Gradient–free MDF	None	692	7,425	7,425
ii	Gradient–free MDF	AVL/Beam Surrogate	997	5,412	5,412
iii	Gradient–free MDF	Kriging Surrogate	431	3,232	3,232

Table 6.1: Results of the number of simulations required during the optimization to find an optimal wing design. The number of evaluations does not include the offline cost of creating surrogate models. The number of MDAs refers to the number of designs for which iterative aerostructural solves were completed successfully. The IDF method chooses initial targets from an MDA of the initial design.



(a) Parallel IDF method convergence plot.



(b) Parallel IDF method discipline evaluations

Figure 6-5: Sample results for the parallel IDF method.

Three low-fidelity models were used to solve the MDF formulation with Algorithm 5.2, (i) No low-fidelity model, (ii) a Kriging surrogate made from 2,000 MDAs of the wing using a two-dimensional vortex lattice method, Athena Vortex Lattice (AVL), and a one-dimensional beam model in Nastran (95% of the analyses were successful), and (iii) a Kriging surrogate

made from 500 MDAs of the wing using the high-fidelity methods (81% of the analyses were successful). An MDA refers to solving for a set of discipline responses that satisfy the coupling relationship between disciplines, but the system level-constraints, (5.1b) and (5.1c), are not necessarily satisfied. Parallelization was not implemented with this method because of the number of Nastran licenses available. Table 6.1 lists the average numbers of successful MDAs, structural analyses, and aerodynamic analyses required to optimize the wing from random initial designs. Figure 6-6 shows a representative convergence history of Algorithm 5.2 using the Kriging surrogate of the high-fidelity MDAs.

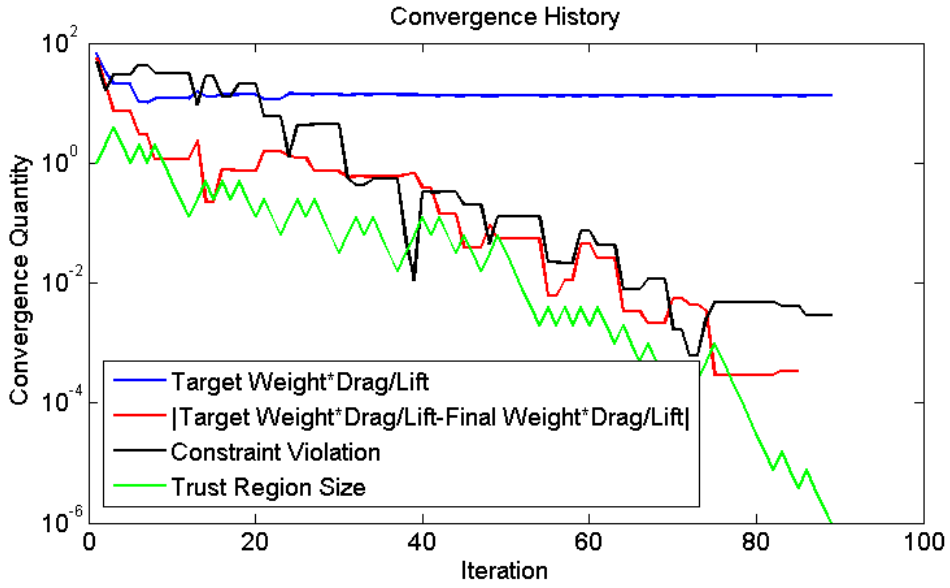


Figure 6-6: Representative convergence history of Algorithm 5.2 on the aerostructural design problem. The low-fidelity model is a Kriging surrogate built from 500 high-fidelity evaluations, of which 81% were successful.

The results presented in Table 6.1 show that with no low-fidelity analysis the parallel IDF method required 22% more structural evaluations and 3% more aerodynamic evaluations than the MDF method with no low-fidelity simulation, case (i). Compared with other decoupled multidisciplinary methods this is an exciting result because methods like CSSO and CO typically require at least twice as many evaluations as single-level methods [111]. Moreover, when examining the multifidelity results obtained with Algorithm 5.2 we observe that compared with using no low-fidelity model, using a mediocre low-fidelity model, case

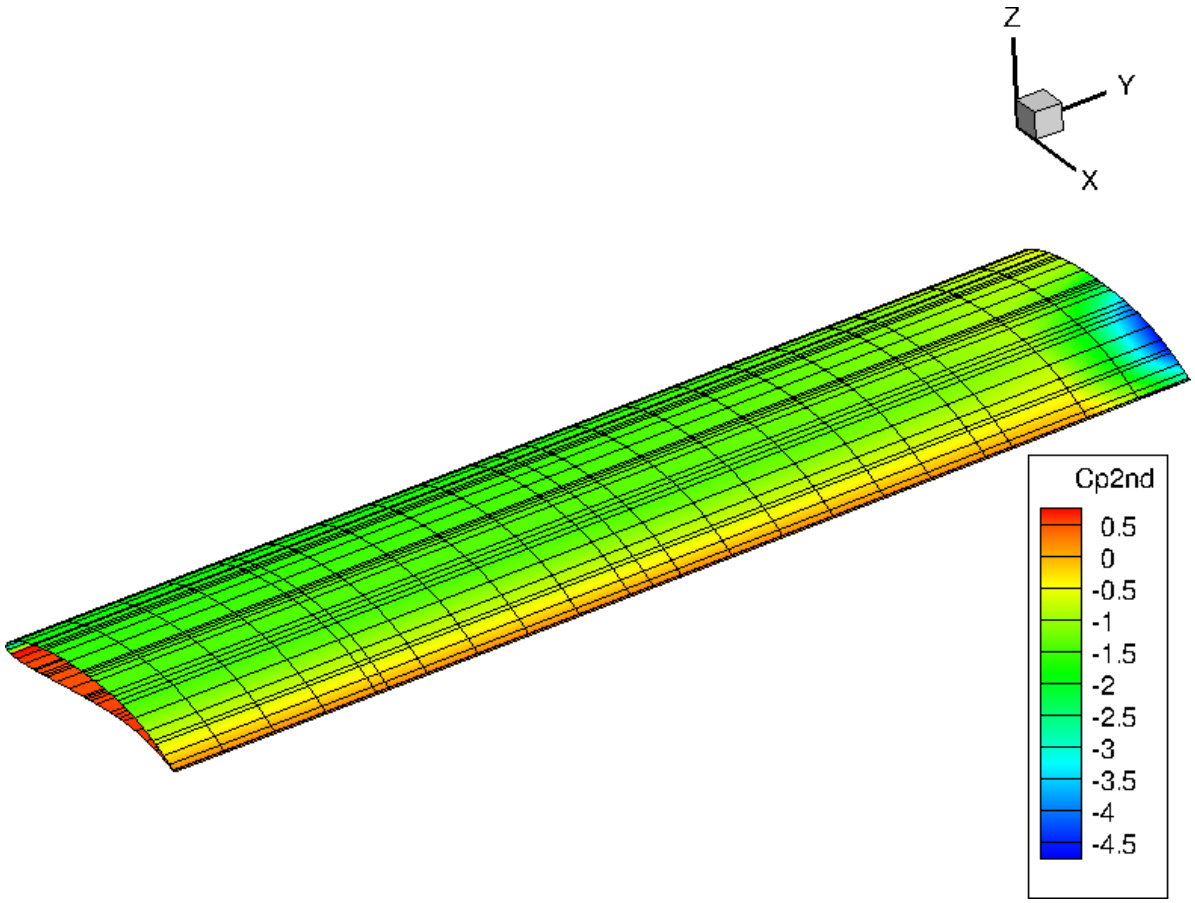
(ii), the number of high-fidelity evaluations decreases by approximately 27%, and with a good low-fidelity model, case (iii), by approximately 66%. (The decrease in case (iii) is approximately 50% if you include the offline cost of the results used to create the surrogate model.) It is illustrative to consider the solution times of these algorithms in a scenario where only one analysis per discipline can be conducted at a time, a situation akin to engineers designing a multidisciplinary system. The wall-clock solution times in days for the test cases presented in Table 6.1 for this scenario are (0) 2.67, (i) 4.73, (ii) 3.45, and (iii) 2.06.[2] In this scenario, we have clearly established a benefit for system decomposition, a 44% reduction in time, and multifidelity optimization, a 56% reduction in time. A locally optimal wing design is shown in Figure 6-7, the aerodynamic results are in Figure 6-7(a), and structural results are shown in Figure 6-7(b).

6.2 Adjoint-based Coupled Aerostructural Optimization

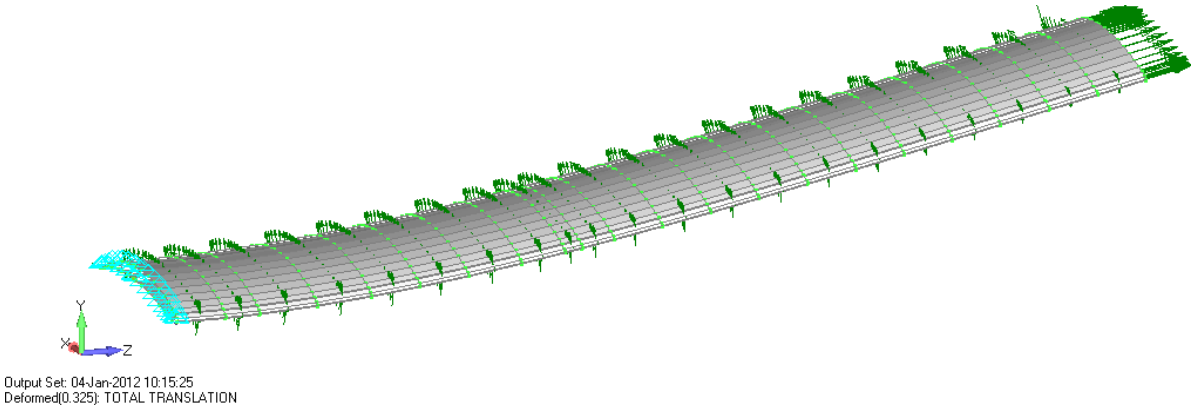
This wing design problem is based on the Bombardier Q400 turboprop airplane, and is from Kennedy *et al.* [54]. The optimization problem is to find the minimum drag wing that supports the weight of the Q400 flying at $1g$ cruise. Any weight savings in the wing structure is subtracted from the weight of the airplane. The parameters describing the aircraft and cruise condition are presented in Table 6.2.

The design problem has 104 design variables and 77 constraints. The design variables are the angle of attack, eight twist angles for the free-form deformation (FFD) control volume [57], and 19 variables for each of the upper surface skin thicknesses, lower surface skin thicknesses, forward spar thicknesses, aft spar thicknesses, and rib thicknesses. The 19 material thickness variables represent one constant thickness panel of metal between and for each rib. Each variable has appropriate upper and lower bounds that account for minimum gauge constraints. The nonlinear constraints are that the lift of the airplane is equal to the

[2]Using 2 cores for a Windows 7 virtual machine running on a 3.40GHz Intel i7-2600 system in Ubuntu 11.04.



(a) High-Fidelity C_p distribution on deformed wing.



(b) High-Fidelity structural model results. Deflection is amplified by a factor of 32.5.

Figure 6-7: High-fidelity aerostructural wing design shown in $1g$ flight condition and Mach 0.25.

Parameter	value
Maximum takeoff weight	64,500 lbm
Cruise Mach number	0.6
Cruise Altitude	22,000 ft
Cruise weight fraction	0.8
Load case	1g
Wingspan	93.2 ft
Planform area	737 ft^2
Aspect ratio	11.78
Initial airfoil stack	NACA 2412
Wing trapezoidal break [% semi-span]	40%
Taper ratio	0.54
Spars [% chord]	15%, 50%
Ribs [#]	20
Ribs [% chord]	15%-75%
Structural material	Aluminum 7075
Elastic modulus	10,150 ksi
Poisson's ratio	0.33
Yield stress	46.4 ksi

Table 6.2: Q400-type airplane properties and flight condition.

weight, four Kreisselmeier–Steinhauser (KS) stress lumping functions, one for the ribs, one for the spars, one for the upper surface skin panels, and one for the lower-surface skin panels, and 72 relative thickness constraints. The relative thickness constraints ensure that the change in thickness between adjacent panels is small enough to prevent stress concentrations.

In the following subsections we discuss the multifidelity setup and the creation of high- and low-fidelity aerostructural models. We then discuss the optimization methods used, and finally the results.

6.2.1 Multifidelity Setup

To demonstrate the use of multifidelity optimization on a multidisciplinary system we use two-fidelity levels for each the aerodynamics and structural disciplines. However, we only use a multidisciplinary feasible (MDF) strategy, which means that for every wing design evaluated the coupled aerostructural system is solved iteratively. In this framework, we only interface low-fidelity aerodynamics with low-fidelity structures and high-fidelity aerodynam-

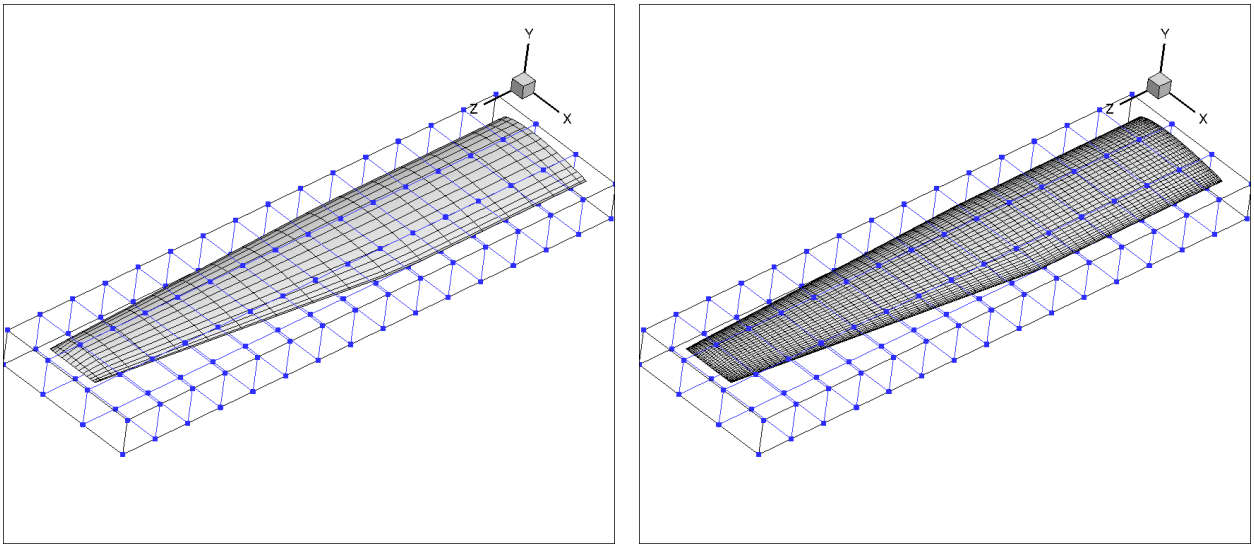
ics with high-fidelity structures. The high-fidelity methods are a panel method for aerodynamics and a 2^{nd} order finite-element model including all of the ribs, spars, and skins as shell elements. The low-fidelity methods are the same methods but using significantly coarsened discretizations; in the aerodynamics case, barely enough panels are used to smoothly represent the geometry. Table 6.3 compares the complexity of the high- and low-fidelity models. In addition, Figure 6-8 shows a picture of the high- and low-fidelity aerodynamic models in the FFD control volume.

		Low-Fidelity	High-Fidelity
Aerodynamic Panels	chordwise	30	70
	spanwise	20	80
	wake	20	60
	total	**1,000**	**10,400**
Structural Elements	chordwise	5	16
	spanwise	30	190
	thickness	4	12
	total (dof)	**5,624**	**57,152**
Time	(cpu $\times$ s)	16	4,800-9,600

Table 6.3: Comparison of the number of degrees of freedom for the high- and low-fidelity models. Run-time includes one function and one gradient evaluation.

6.2.2 Optimization Strategy

To optimize the high-fidelity wing design two approaches are taken. The first is to optimize the wing using a sequential quadratic programming method [45, 86]. The second approach is to use the gradient-based parallel function evaluation algorithm, Algorithm 5.3, with two fidelity levels. To convert this design problem into an equality constrained optimization problem for Algorithm 5.3, the 72 relative thickness constraints and 4 KS function stress constraints are converted to equality constraints through the addition of 76 slack variables. The modified optimization problem has 180 design variables and 77 constraints. Since only the first 5 constraints, lift equals weight and the 4 KS functions, are the results of analysis and the other 72 constraints are geometric, only the 5 expensive constraints are treated with multifidelity methods. The geometric constraints are not approximated. Where $D(\mathbf{x})$ is



(a) Low-Fidelity panel mesh and FFD Control Points.

(b) High-Fidelity panel mesh and FFD Control Points.

Figure 6-8: Free-form deformation control points for the high- and low-fidelity Q400 wings and aerodynamic panels used.

drag, $L(\mathbf{x})$ is lift, $KS(\mathbf{x})$ are the KS functions, and $\mathbf{g}(\mathbf{x}) \leq 0$ are the geometric constraints, the optimization problem is

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^{104},\bar{\mathbf{p}}\in\mathbb{R}^{4},\mathbf{p}\in\mathbb{R}^{72}} \quad & D(\mathbf{x}) \\ \text{s.t.} \quad & L(\mathbf{x}) - w(\mathbf{x}) = 0 \\ & KS(\mathbf{x}) + \bar{\mathbf{p}}^2 = 0 \\ & \mathbf{g}(\mathbf{x}) + \mathbf{p}^2 = 0. \end{aligned} \tag{6.4}$$

The aerostructural analysis solver provides an adjoint solution, so inexpensive gradient estimates are available for both the objective function and the constraints. To successfully exploit the available gradient information the error models used to calibrate the low-fidelity simulation results to the high-fidelity need to be first-order consistent. The Cokriging method, Algorithm 4.2, would be used for this except the dimensionality of this problem is too high. The cost of constructing and repeatedly evaluating a Cokriging error model with 180 design variables would approach the cost of the high-fidelity simulation in this case. So

the error model is constructed using Algorithm 4.2, but only the derivative at the current iterate, $\mathbf{x}_k$, is added to $\mathbf{R}_C$. The derivative of the error at all other calibration points is not used, so the error model is first-order consistent at $\mathbf{x}_k$ and zeroth-order consistent at all other calibration locations. This is similar to the technique used in [44], however, with additional precautions to ensure the error model is accurate. We also fix the correlation lengths, $\xi = 2$, instead of optimizing them, due to the computational expense of that optimization problem.

6.2.3 Results

Table 6.4 presents the number of high-fidelity evaluations required to optimize the wing using SQP and the number of high- and low-fidelity evaluations required to optimize the wing using Algorithm 5.3 from the same initial wing. For the single-fidelity SQP optimization, SNOPT [45] (wrapped with pyOpt [86]) successfully found the optimal design with default settings. As shown in Table 6.4, Algorithm 5.3 successfully reduced the time needed for optimization by over 40% compared with SNOPT. The convergence history of Algorithm 5.3 on this wing design problem is presented as Figure 6-9. It is evident from the convergence history that even though Algorithm 5.3 required 36 high-fidelity evaluations to find the optimal design, it in actuality converged to a nearly optimal feasible design quite rapidly, in approximately 16 iterations or 17 high-fidelity evaluations. So, had Algorithm 5.3 been terminated in half the time, a feasible design that was nearly optimal would have already been found. However, two types of difficulties were encountered with Algorithm 5.3. First, when the initial trust region was too large, the high-fidelity evaluation failed to converge. This is easily remedied by using a smaller initial trust region. Second, the trust-region subproblems in Algorithm 5.3 are solved using SNOPT, and SNOPT frequently fails to maintain feasibility when calculating the tangential step. This issue required tuning parameters of SNOPT. The parameter values used in Algorithm 5.3 and to tune SNOPT are presented in Table 6.5.

Figure 6-10 shows the initial and final FFD control volumes with the optimal unloaded wing inside. Most of the geometric changes to the wing are outboard. The more substantial changes to the wing are in the material thicknesses, since the stresses in the initial design are above the material yield stress. Figure 6-11(a) shows the von Mises stress distribution for

	Low-Fidelity	High-Fidelity	Time (cpu × day)
SQP (SNOPT)	–	137	12.0
Multifidelity (Algorithm 5.3)	16,431	36	6.4

Table 6.4: Comparison of the required number of high- and low-fidelity evaluations to find the optimal wing design. The optimal design from SQP was slightly more accurate than that from the multifidelity method, the objective function value was 1.12×10^{-1} compared with 1.16×10^{-1}, and the constraint violation was 1.8×10^{-5} compared with 8.9×10^{-5}.

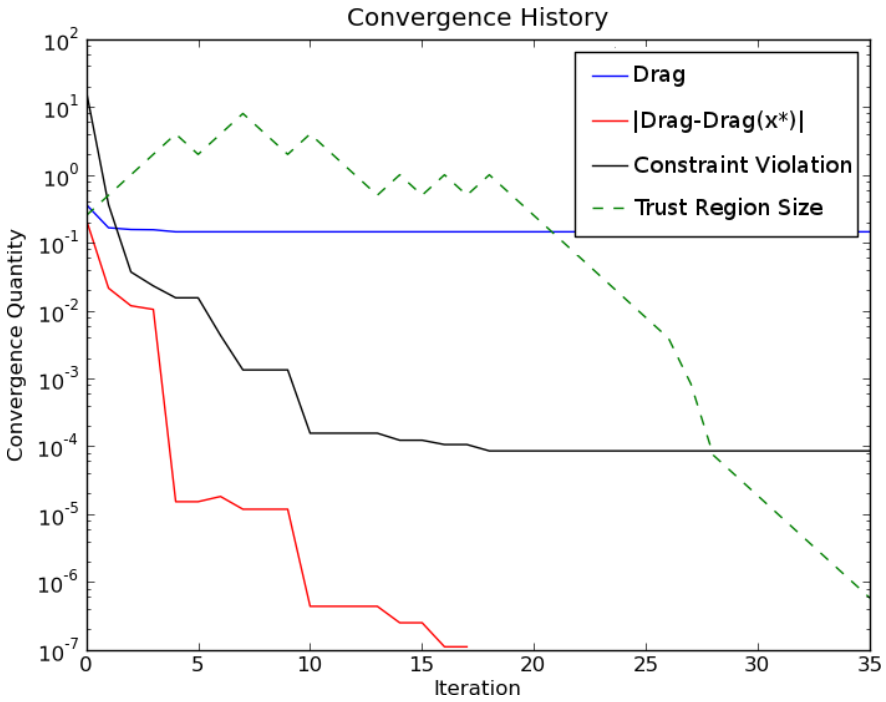


Figure 6-9: Convergence history of Algorithm 5.3 on the gradient-based optimization of a wing.

the low-fidelity analysis of the optimal wing design, and Figure 6-11(b) shows the von Mises stress distribution for the high-fidelity analysis of the optimal wing design. The maximum stress in the wing is approximately half of the material yield stress, so the optimal design is feasible. In addition, Figure 6-12(a) shows the C_p distribution from the low-fidelity analysis of the optimal wing, and Figure 6-12(b) shows the C_p distribution for the high-fidelity analysis of the optimal wing design.

Constant	Description	Value
β	Filter acceptance criteria	0.25
Δ_0	Initial trust region radius	0.25
$\Delta_{\max}$	Maximum trust region size	$100\Delta_0$
ϵ	Termination Tolerance	1×10^{-4}
γ_0	Trust region contraction ratio	0.5
γ_1	Trust region expansion ratio	2
η	Constraint decrease requirement	0.995
ξ	Radial basis function correlation length	2
	Maximum SNOPT iterations, normal step	800
	Maximum SNOPT iterations, tangential step	800
	SNOPT objective value tolerance	$\frac{1\times10^{-2}\epsilon}{(k+1)^3}$
	SNOPT constraint violation tolerance	$\frac{1\times10^{-4}\epsilon}{(k+1)^3}$
	SNOPT initial penalty, tangential step	$\frac{(k+1)^3}{\epsilon}$

Table 6.5: List of parameters used in the gradient-based AAO algorithm. The parameters used in constructing the radial basis function error models are given in Table 2.2.

6.3 Summary

This chapter has demonstrated that the multifidelity framework for multidisciplinary optimization developed in this thesis is capable of optimizing complex, highly-coupled, systems with computationally expensive analyses. The results showed that the framework does not require gradient information to find an optimal design, though it is capable of exploiting gradient information when available to speed optimization. Modifications to the gradient-free algorithm to handle evaluation failures were shown to result in a robust optimization framework that yielded optimal designs when many conventional approaches failed. In addition, the parallelized IDF framework meets or exceeds the performance of serial system design methods, and the MDF framework successfully reduced the time to optimize an aerostructural system by approximately half.

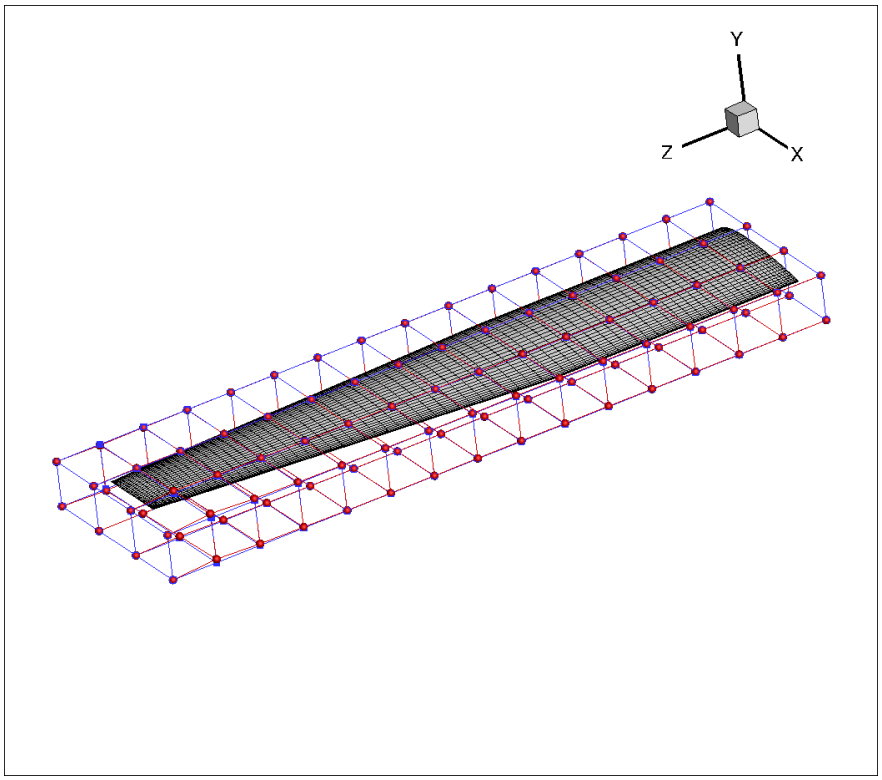
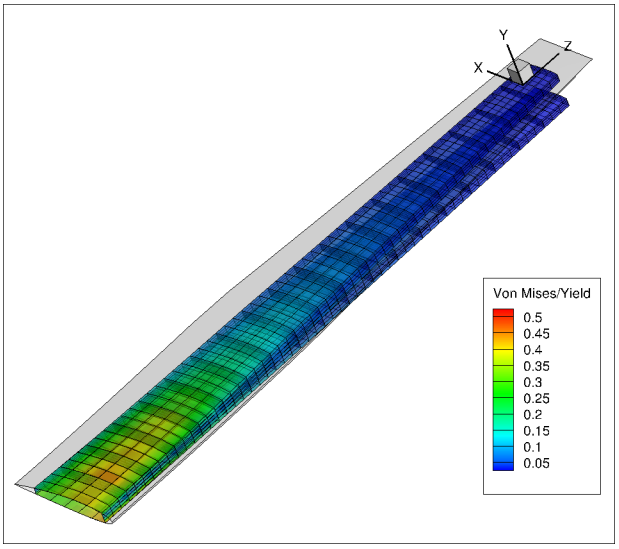
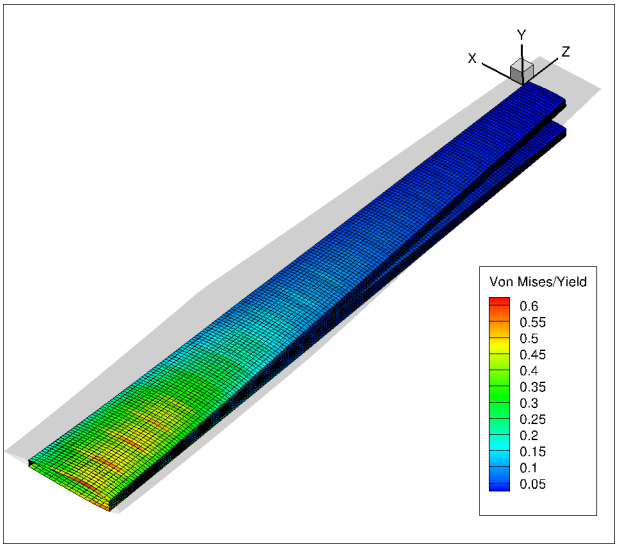


Figure 6-10: Comparison of initial (blue squares) and final (red circles) FFD control point locations and the unloaded final wing configuration.

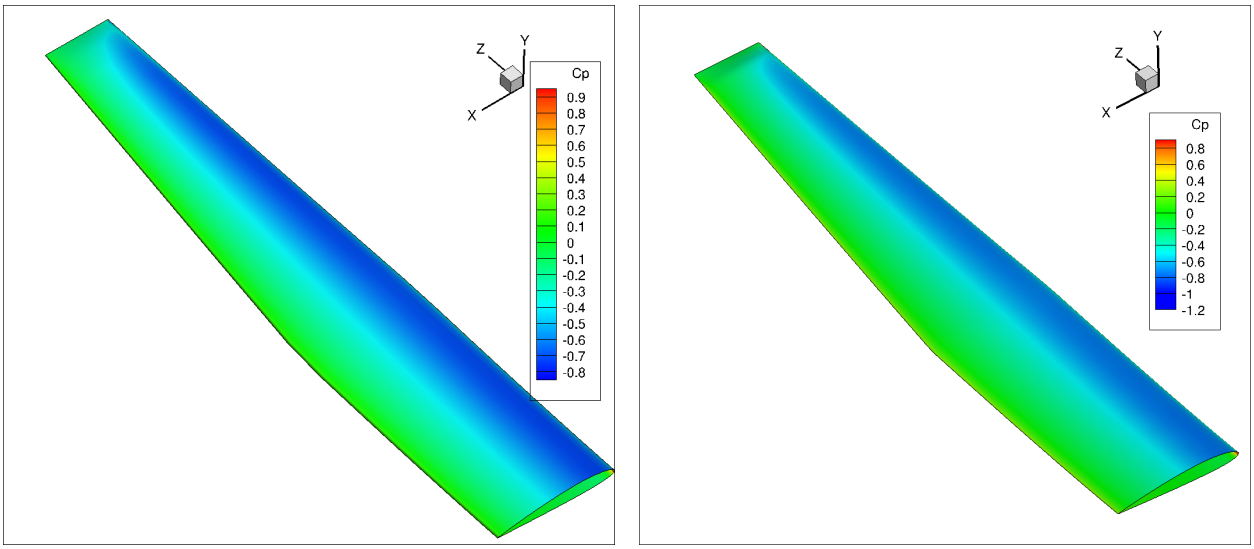


(a) Low-Fidelity von-Mises stress normalized by aluminum 7075 yield stress.



(b) High-Fidelity von-Mises stress normalized by aluminum 7075 yield stress.

Figure 6-11: Von-Mises stress contour plot on the deformed and undeformed Q400 wingbox.



(a) Low-Fidelity C_p distribution.

(b) High-Fidelity C_p distribution.

Figure 6-12: C_p distribution contour plot on the deformed Q400 wing.

Chapter 7

Conclusions and Future Work

The objective of this thesis is to develop methods to aid the design and optimization of multidisciplinary systems that are analyzed with expensive simulations. The challenges addressed are a lack of design sensitivity information, the need to best exploit design sensitivity information when available, and the ability to parallelize the design process. A summary of the work done to meet each objective is given in the following section, which is followed by a specific list of the contributions made, the general conclusions that can be drawn from this research, and finally a discussion of future research that should be considered.

7.1 Thesis Summary

This thesis developed multifidelity methods to speed optimization problems containing costly analyses. Chapter 2 presented a multifidelity optimization method for expensive optimization problems in which sensitivity information is unavailable. In addition, Chapter 2 also presented a method to combine information from multiple fidelity levels to best predict a high-fidelity analysis result. Chapter 3 presented multifidelity optimization methods applicable to non-linearly constrained optimization problems for which sensitivity information is unavailable. Chapter 4 presented a multifidelity optimization method to best exploit high-fidelity sensitivity information when it is available. These multifidelity optimization methods then served as a framework for the multidisciplinary optimization methods devel-

oped in Chapter 5. The two multidisciplinary methods developed speed the optimization of a system comprising multiple interacting subsystems through both multifidelity optimization and parallelization. The multifidelity and multidisciplinary optimization framework developed in Chapters 2 through 5 were demonstrated on two multifidelity and multidisciplinary aerostructural design problems in Chapter 6.

7.2 Contributions

This thesis contains the following contributions:

1. Created a Bayesian model calibration framework that synthesizes multiple sources of lower-fidelity information to accurately estimate high-fidelity performance.

2. Developed a multifidelity optimization methodology that does not require high-fidelity gradient information, is broadly applicable, and sufficiently robust for industry-level design problems.

3. Proposed a parallelized multidisciplinary optimization framework that handles multiple disciplines with multiple fidelity levels, enabling a flexible and efficient system optimization process.

4. Successfully optimized a tightly coupled aerostructural aircraft design problem where other optimization methods failed due to unavailable gradient information, non-smoothness and evaluation failures.

7.3 Conclusions

This thesis has presented a multifidelity framework for optimization of multidisciplinary systems that does not require estimating gradients of high-fidelity functions, enables the use of multiple low-fidelity models, enables optimization of functions with hard constraints, and exhibits robustness over a broad class of optimization problems, even when non-smoothness is present in the objective function and/or constraints. In addition, this thesis developed

two methods to parallelize the design of multidisciplinary systems, a method to optimize individual disciplines in parallel, and a method to evaluate candidate designs in parallel. The complete framework is applicable to a broad class of optimization problems, and for all test cases in this thesis, the presented methods met or exceeded the performance of comparable methods.

Three single-discipline gradient-free multifidelity methods were developed in this thesis, one unconstrained method, and two constrained methods. The unconstrained method is proven to converge to a locally optimal high-fidelity solution, and a theoretical analysis of the two constrained formulations is presented in order to discuss performance limitations and robustness. For airfoil design problems, these approaches have been shown to perform similarly in terms of the number of function evaluations to finite-difference-based multifidelity optimization methods. Moreover, when compared with single-fidelity optimization methods, the single-discipline gradient-free methods provide a 70 to 90% reduction in high-fidelity evaluations. The methods are demonstrated to be robust in the presence of computational noise, and in theory should also be able to solve a broad class of optimization problems such as optimizing analysis results that often fail, are non-smooth, or are experimental. This suggests that the multifidelity derivative-free approaches are a promising alternative for the wide range of problems where finite-difference gradient approximations are unreliable.

This thesis has also shown that a multifidelity optimization method based on a maximum likelihood estimator is an effective way of combining many fidelity levels to optimize a high-fidelity function. The maximum likelihood estimator permits flexible sampling strategies among the low-fidelity models and is robust with respect to poor low-fidelity estimates. In addition, the estimator offers a natural and automated way of selecting among different models that are known to be accurate in different parts of the design space, which is frequently the case in engineering design.

This thesis extended the maximum likelihood estimation multifidelity optimization approach to gradient-exploiting optimization. This method attempts to best use high-fidelity gradient information by constructing gradient-exploiting surrogate models of high-fidelity analyses. The results showed that for both structural design and airfoil shape optimization

the gradient-exploiting multifidelity formulation reduced the number of high-fidelity evaluations compared with single-fidelity methods. In addition, the method met or exceeded the performance of other gradient-based multifidelity methods. However, it was determined that the computational cost of constructing the gradient-exploiting surrogates scales poorly with dimension and the reuse of inaccurate gradient information is detrimental to performance. Therefore, this technique only seems practical for optimization when simulations are quite expensive and the dimension of the design space is small.

The single-disciplinary multifidelity optimization methods developed in this thesis serve as a foundation for two multidisciplinary system design strategies. The methods are novel in that they enable parallelized optimization of systems for which design sensitivity information is not available. For the example problems presented, the parallel IDF formulation is competitive with single-level multidisciplinary optimization formulations with two-disciplines and occasionally exceeds performance of single-level formulations with three disciplines. This is an encouraging result as bilevel multidisciplinary optimization methods typically perform slower than their single-level counterparts. An important attribute of the parallel IDF formulation is that it enables exploitation of aspects of the individual disciplines that may speed the discipline-level optimizations, for example, the ability to use gradient-based optimization, gradient-free optimization, or multifidelity optimization. The parallel AAO approach is applicable to general equality constrained optimization problems. The method does not require gradient information, though it can exploit it when available. For the example problems presented, the performance of the method is competitive with gradient-based methods even when no low-fidelity model is used. In addition, it is demonstrated by optimizing a three-dimensional wing, that these multidisciplinary optimization methods are even capable of optimizing systems with high-bandwidth coupling. Both methods successfully found optimal aerostructural designs. The performance of these methods on all test problems in this thesis suggest they offer a significant opportunity to design multidisciplinary systems.

7.4 Future Work

Many avenues for further research relating to these methods are apparent. The first is to test the multidisciplinary methods on other systems with more than two disciplines. It is an open question if the performance observed on the simple three-discipline test cases and the aerostructural design problem extend to real-world systems with multiple disciplines. Similarly, it is an open question what the most effective processor load balancing is for multidisciplinary systems. The two multidisciplinary methods are parallelizable, but simply distributing smaller optimization problems or function evaluations to available processors may not be the most effect use of the processors. When disciplines include parallelized simulations or when it is possible to both optimize disciplines and evaluate simulations in parallel, it is unknown what the best uses of the computational resources are.

The parallel IDF method developed in this thesis occasionally converged slower than the other multidisciplinary methods tested. Perhaps it is possible to combine an augmented Lagrangian trust region framework [30, 31] with the parallelized structure of the IDF method presented. Since the trust-region augmented Lagrangian method is globally convergent this could add robustness to the IDF method developed. In addition, if it were possible to combine the optimization over the targets and the system design variables, the IDF method may be able to converge faster. It was mentioned that this decreased stability, however, if the stability of this approach is improved, either through a trust-region method or otherwise, then reducing the number of directions being alternated may speed convergence of the IDF method.

In addition, the aerostructural optimization problem demonstrated a reduced-dimensional coupling to speed the system optimization. Current multidisciplinary optimization methods only consider multiple fidelity levels within a discipline or for the entire system model, they do not address multifidelity system couplings. Perhaps there is an opportunity to speed system design using multifidelity couplings or using reduced order models to reduce the dimensionality of the system couplings adaptively. This idea offers a chance to reduce the cost of the discipline-level optimization problems, and the ability to reduce work needed to

satisfy multidisciplinary constraints. Similarly, a design space of increasing fidelity may also speed system design. For example, starting with a simple parameterization of a system is it possible to both optimize the system and add design resolution at the same time? It is simpler to optimize low-dimensional problems, and by starting with few design parameters the initial optimal system designs can be found quickly. Then through an adjoint or other approach detect where more design parameters are necessary and evolve both the design and design space simultaneously.

Appendix A

Details of AAO Trust-Region Algorithm Theory

This appendix presents details of the theory to support the observed convergence behavior of the AAO Trust-Region Algorithm presented in Section 5.3.3. The appendix is separated into four sections, it first demonstrates (i) the constraint violation reduction obtained at each iteration in Section A.1, and then discusses (ii) the interaction between the constraint violation and the trust region size caused by the filter in Section A.2. It then discusses the objective function and (iii) the reduction obtained at each iteration when the constraint violation is small in Section A.3, and concludes by showing (iv) that the filter acceptance criteria ensures that all limit points of Algorithm 5.2 are optimal solutions to (5.24) in Section A.4.

To simplify the notation in this section, we consider $\mathbf{x}$ to have dimension n and $\mathcal{H}(\mathbf{x})$ to have dimension m, and define $\kappa_{\nabla\mathcal{H}} = \sqrt{mn}\max_{i=1,\ldots,m}\{\kappa_g^i\}$, where κ_g is from the definition of a fully linear model Eq. 2.7. From the properties of norms, this enables us to write a fully linear condition for the Jacobian of the surrogate models, $\|\nabla\mathcal{H}(\mathbf{x}) - \nabla\bar{\mathcal{H}}(\mathbf{x})\| \leq \kappa_{\nabla\mathcal{H}}\Delta_k$. Similarly, we define $\kappa_{\mathcal{H}} = \sqrt{m}\max_{i=1,\ldots,m}\{\kappa_f^i\}$ in order to write $\|\mathcal{H}(\mathbf{x}) - \bar{\mathcal{H}}(\mathbf{x})\| \leq \kappa_{\mathcal{H}}\Delta_k^2$.

All norms in this section should be interpreted as the 2-norm unless otherwise stated.[1,2]

A.1 Constraint Violation Decrease Demonstration

In this section we show that for any trust region sufficiently small and under assumptions AAO1 and AAO4 then the fraction of Cauchy decrease conditions in Eqs. 5.27 and 5.30 ensure a decrease in the constraint violation, $\|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\| < \|\mathcal{H}(\mathbf{x}_k)\|$. As the constraint Jacobian has full row rank (Assumption AAO4), then if the constraint violation is non-zero there exists a positive constant, $\varrho(\mathbf{x}_k)$, such that $\|\nabla\mathcal{H}(\mathbf{x}_k)^\top\mathcal{H}(\mathbf{x}_k)\| = \varrho(\mathbf{x}_k) > 0$. This fact enables us to bound the error between the surrogate quantity used in Eqs. 5.27 and 5.30, $\|\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\mathcal{H}(\mathbf{x}_k)\|$, and the high-fidelity counterpart, $\nabla\mathcal{H}(\mathbf{x}_k)^\top\mathcal{H}(\mathbf{x}_k)$. (Note: $\bar{\mathcal{H}}(\mathbf{x}_k) = \mathcal{H}(\mathbf{x}_k)$ by construction.) Using assumptions AAO1, AAO4, and the definition of a fully linear model,

$$\begin{aligned}
\nabla\mathcal{H}(\mathbf{x}_k)^\top\mathcal{H}(\mathbf{x}_k) - \nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\mathcal{H}(\mathbf{x}_k) &= \left[\nabla\mathcal{H}(\mathbf{x}_k) - \nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)\right]^\top\mathcal{H}(\mathbf{x}_k) \\
\|\nabla\mathcal{H}(\mathbf{x}_k)^\top\mathcal{H}(\mathbf{x}_k) - \nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\mathcal{H}(\mathbf{x}_k)\| &= \|\left[\nabla\mathcal{H}(\mathbf{x}_k) - \nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)\right]^\top\mathcal{H}(\mathbf{x}_k)\| \\
&\leq \|\nabla\mathcal{H}(\mathbf{x}_k)^\top - \nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\|\|\mathcal{H}(\mathbf{x}_k)\| \\
&\leq \kappa_{\nabla\mathcal{H}}\Delta_k\|\mathcal{H}(\mathbf{x}_k)\|,
\end{aligned}$$

therefore,

$$\|\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\mathcal{H}(\mathbf{x}_k)\| \geq \varrho(\mathbf{x}_k) - \kappa_{\nabla\mathcal{H}}\Delta_k\|\mathcal{H}(\mathbf{x}_k)\|. \tag{A.1}$$

[1] A commonly used identity in this section is that $\|\mathbf{a} - \mathbf{b}\| \leq c \implies \|\mathbf{a}\| \geq \|\mathbf{b}\| - c$. A proof of this is,

$$\begin{aligned}
\mathbf{b} &= \mathbf{b} - \mathbf{a} + \mathbf{a} \\
\|\mathbf{b}\| &= \|\mathbf{b} - \mathbf{a} + \mathbf{a}\| \\
&\leq \|\mathbf{b} - \mathbf{a}\| + \|\mathbf{a}\| \\
&\leq c + \|\mathbf{a}\|
\end{aligned}$$

[2] Another commonly used identity is that for any finite positive constants a and b and positive sequence $\{\Delta_k\}$ with $\lim_{k\to\infty}\Delta_k = 0$, there exists a Δ_k such that $a\Delta_k - b\Delta_k^2 > \frac{a}{2}\Delta_k$.

We may now use this bound to show that for small Δ_k, the decrease conditions in Eqs. 5.27 and 5.30 ensure

$$\|\mathcal{H}_k(\mathbf{x}_k)\|_2^2 - \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^\perp)\|_2^2 \geq 2\kappa_{fcd}\varrho(\mathbf{x}_k)\Delta_k - 2\kappa_{fcd}\kappa_{\nabla\mathcal{H}}\|\mathcal{H}(\mathbf{x}_k)\|\Delta_k^2. \tag{A.2}$$

We have so far shown for Eq. 5.27,

$$\begin{aligned}\|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^\perp)\|_2^2 \geq \\ 2\kappa_{fcd}\left(\varrho(\mathbf{x}_k) - \kappa_{\nabla\mathcal{H}}\Delta_k\|\mathcal{H}(\mathbf{x}_k)\|\right)\min\left\{\Delta_k, \tfrac{\varrho(\mathbf{x}_k)-\kappa_{\nabla\mathcal{H}}\Delta_k\|\mathcal{H}(\mathbf{x}_k)\|}{\|\nabla\bar{\mathcal{H}}(\mathbf{x}_k)^\top\nabla\bar{\mathcal{H}}(\mathbf{x}_k)+\sum_{i=1}^m \bar{h}_k^i(\mathbf{x}_k)\nabla^2\bar{h}_k^i(\mathbf{x}_k)\|}\right\},\end{aligned} \tag{A.3}$$

and similarly for Eq. 5.30 that,

$$\begin{aligned}\|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}_k^\perp)\|_2^2 \geq \\ 2\kappa_{fcd}\left(\varrho(\mathbf{x}_k) - \kappa_{\nabla\mathcal{H}}\Delta_k\|\mathcal{H}(\mathbf{x}_k)\|\right)\min\left\{\Delta_k, \tfrac{\varrho(\mathbf{x}_k)-\kappa_{\nabla\mathcal{H}}\Delta_k\|\mathcal{H}(\mathbf{x}_k)\|}{\|\nabla\bar{\mathcal{H}}(\mathbf{x}_k)^\top\nabla\bar{\mathcal{H}}(\mathbf{x}_k)\|}\right\}.\end{aligned} \tag{A.4}$$

For Δ_k sufficiently small, Eq. A.3 is equivalent to Eq. A.4, because in both cases the min operator returns Δ_k. It can be shown the alternative to the trust region size in the min operator is bounded from below when the constraint violation is nonzero,

$$\begin{aligned}\frac{\|\nabla\bar{\mathcal{H}}(\mathbf{x}_k)^\top\bar{\mathcal{H}}(\mathbf{x}_k)\|}{\|\nabla\bar{\mathcal{H}}(\mathbf{x}_k)^\top\nabla\bar{\mathcal{H}}(\mathbf{x}_k)\| + m\|\bar{\mathcal{H}}(\mathbf{x}_k)\|\kappa_m} \geq \\ \frac{\varrho(\mathbf{x}_k) - \kappa_{\nabla\mathcal{H}}\Delta_k\|\mathcal{H}(\mathbf{x}_k)\|}{\|\nabla\mathcal{H}(\mathbf{x}_k)\|^2 + 2\kappa_{\nabla\mathcal{H}}\|\nabla\mathcal{H}(\mathbf{x}_k)\|\Delta_k + \kappa_{\nabla\mathcal{H}}^2\Delta_k^2 + m\|\mathcal{H}(\mathbf{x}_k)\|\kappa_m} = \delta_1(\mathbf{x}_k),\end{aligned} \tag{A.5}$$

where, κ_m is the maximum Lipschitz constant for any constraint gradient in $\nabla\bar{\mathcal{H}}(\mathbf{x})$. We can establish that $\delta_1(\mathbf{x}_k) > 0$ by examining this bound for small Δ_k (noting, $\kappa_{\nabla\mathcal{H}}\Delta_k\|\mathcal{H}(\mathbf{x}_k)\| \to 0$), and the definition of κ_τ is from AAO4),

$$\delta_1(\mathbf{x}_k) > \frac{\varrho(\mathbf{x}_k)}{4\|\nabla\mathcal{H}(\mathbf{x}_k)\|^2 + m\|\mathcal{H}(\mathbf{x}_k)\|\kappa_m} > \frac{\kappa_\tau\|\mathcal{H}(\mathbf{x}_k)\|}{4\kappa_d^2 + m\kappa_m\|\mathcal{H}(\mathbf{x}_k)\|}. \tag{A.6}$$

Accordingly, for all Δ_k sufficiently small we have shown that Eq. A.2 holds.

Algorithm 5.2 enables switching from minimizing the 2-norm of the constraint violation

to the Gauss-Newton step when computing the tangential step. A bound for the maximum error between the surrogate model constraint violation, $\frac{1}{2}\|\bar{\mathcal{H}}(\mathbf{x}_k + \mathbf{s}_k^{\perp})\|_2^2$, and the quadratic approximation, $b_k(\mathbf{s}_k^{\perp})$ in Eq. 5.31, within a trust region is $\kappa_m\|\bar{\mathcal{H}}(\mathbf{x}_k)\|\Delta_k^2 + \frac{1}{4}\kappa_m^2\Delta_k^4$ [83, Pages 294-5]. Combining Eq. A.2, the additional error possible when switching to the Gauss-Newton step from the 2-norm step, and the constraint relaxation in the calculation of the tangential step, $b_k(\mathbf{0}) - b_k(\mathbf{s}_k) \geq \eta\left[b_k(\mathbf{0}) - b_k(\mathbf{s}_k^{\perp})\right]$, we have that

$$b_k(\mathbf{0}) - b_k(\mathbf{s}_k) \geq \eta\left[\kappa_{fcd}\varrho(\mathbf{x}_k)\Delta_k - \kappa_{fcd}\kappa_{\nabla\mathcal{H}}\Delta_k\|\mathcal{H}(\mathbf{x}_k)\|\Delta_k^2 - \kappa_m\|\bar{\mathcal{H}}(\mathbf{x}_k)\|\Delta_k^2 - \frac{1}{4}\kappa_m^2\Delta_k^4\right] \quad \text{(A.7)}$$

Now each iteration of Algorithm 5.2 with Δ_k sufficiently small ensures

$$\begin{aligned}\frac{1}{2}\|\mathcal{H}(\mathbf{x}_k)\|^2 - \frac{1}{2}\|\bar{\mathcal{H}}(\mathbf{x}_k + \mathbf{s}_k)\|^2 \geq \\ \eta\left[\kappa_{fcd}\varrho(\mathbf{x}_k)\Delta_k - \kappa_{fcd}\kappa_{\nabla\mathcal{H}}\Delta_k^2\|\mathcal{H}(\mathbf{x}_k)\| - 2\left(\kappa_m\|\mathcal{H}(\mathbf{x}_k)\|\Delta_k^2 + \frac{1}{4}\kappa_m^2\Delta_k^4\right)\right].\end{aligned} \quad \text{(A.8)}$$

In addition, from the definition of a fully linear model, we have that $\|\mathcal{H}(\mathbf{x}) - \bar{\mathcal{H}}(\mathbf{x})\| \leq \kappa_{\mathcal{H}}\Delta_k^2$, from which we can establish at any $\mathbf{x}$ in the trust region,

$$\begin{aligned}\|\mathcal{H}(\mathbf{x})\| &\leq \|\bar{\mathcal{H}}(\mathbf{x})\| + \kappa_{\mathcal{H}}\Delta_k^2 \\ \|\mathcal{H}(\mathbf{x})\|^2 &\leq \|\bar{\mathcal{H}}(\mathbf{x})\|^2 + 2\kappa_{\mathcal{H}}\|\bar{\mathcal{H}}(\mathbf{x})\|\Delta_k^2 + \kappa_{\mathcal{H}}^2\Delta_k^4.\end{aligned}$$

Combining these results with the result in Eq. A.8, a necessary conclusion is

$$\begin{aligned}\|\mathcal{H}(\mathbf{x}_k)\|^2 - \|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\|^2 \geq \\ 2\eta\left[\kappa_{fcd}\varrho(\mathbf{x}_k)\Delta_k - \kappa_{fcd}\kappa_{\nabla\mathcal{H}}\Delta_k^2\|\mathcal{H}(\mathbf{x}_k)\| - 2\left(\kappa_m\|\mathcal{H}(\mathbf{x}_k)\|\Delta_k^2 + \frac{1}{4}\kappa_m^2\Delta_k^4\right)\right] \\ -2\kappa_{\mathcal{H}}\|\bar{\mathcal{H}}(\mathbf{x}_k + \mathbf{s}_k)\|\Delta_k^2 - \kappa_{\mathcal{H}}^2\Delta_k^4.\end{aligned} \quad \text{(A.9)}$$

Therefore, there must exist a $\delta_2(\mathbf{x}_k) > 0$ such that for each $\Delta_k \in (0, \delta_2(\mathbf{x}_k))$ steps computed by Algorithm 5.2 ensure (5.36),

$$\|\mathcal{H}(\mathbf{x}_k)\|^2 - \|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\|^2 \geq \eta\kappa_{fcd}\Delta_k\varrho(\mathbf{x}_k).$$

Which means that the steps generated by Algorithm 5.2 decrease the squared constraint violation proportionally to the size of the trust region size as the trust region gets small. We now demonstrate this decrease must be acceptable by the filter for small Δ_k.

A.2 Constraint Violation and Filter Interaction

In this section we demonstrate two important properties of the filter, first, that the filter acceptance criteria ensures that either the trust region size decreases to zero or the constraint violation decreases to zero, and second, that when the trust region is small, the decrease established in Eq. 5.36 is acceptable to the filter.

To demonstrate that the iterates acceptable by the filter guarantee that either the trust region size decreases to zero or the constraint violation decreases to zero, (5.37), $\lim_{k\to\infty} \|\mathcal{H}(\mathbf{x}_k)\|\Delta_k = 0$, we need assumptions AAO2 and AAO3 and to consider two cases, (i) the number of iterates accepted by the filter is finite and (ii) the number of iterates accepted by the filter is infinite. In the first case, if a finite number of iterates are accepted by the filter, then after a final iterate is accepted, there remains an infinity of iterates that will be rejected by the filter and $\Delta_k \to 0$. In addition, $\|\mathcal{H}(\mathbf{x}_k)\| \leq \|\mathcal{H}(\mathbf{x}_0)\| < \infty$, so this case is complete. In the second case, if an infinite number of iterates are accepted by the filter then the sufficient decrease condition ensures that at least Δ_k or $\|\mathcal{H}(\mathbf{x}_k)\|$ must go to zero. Consider the area enclosed by the Pareto front of constraint violation and objective function values, the upper bound on constraint violation, $\|\mathcal{H}(\mathbf{x}_0)\|$, and the upper bound for the objective function on the domain Ω, $\mathcal{F}(\mathbf{x}_{\max})$,

$$G = \{(\mathcal{F}(\mathbf{x}), \|\mathcal{H}(\mathbf{x})\|) | \forall \mathbf{x} \in \Omega, \mathcal{F}(\mathbf{x}) \leq \mathcal{F}(\mathbf{x}_{\max}) \text{ and } \|\mathcal{H}(\mathbf{x})\| \leq \|\mathcal{H}(\mathbf{x}_0)\|\}\,.$$

(G is shown graphically in Figure A-1, left.) The filter dominates an area,

$$F = \Big\{(\mathcal{F}(\mathbf{x}), \|\mathcal{H}(\mathbf{x})\|) | \forall \mathbf{x} \in \Omega,\ \exists \hat{k} \in [1, \ldots, k] : \mathcal{F}(\mathbf{x}_{\hat{k}}) \leq \mathcal{F}(\mathbf{x}) \text{ and } \|\mathcal{H}(\mathbf{x}_{\hat{k}})\| \leq \|\mathcal{H}(\mathbf{x})\|,$$
$$\text{and } \mathcal{F}(\mathbf{x}) \leq \mathcal{F}(\mathbf{x}_{\max}),\ \text{and } \|\mathcal{H}(\mathbf{x})\| \leq \|\mathcal{H}(\mathbf{x}_0)\|\}\,,$$

that must be a subset of the Pareto area, $F \subseteq G$. (F is shown graphically in Figure A-1, center.) In addition, there is a restricted area in front of the filter which is imposed by the sufficient decrease condition in the filter acceptance criteria,

$$\Big\{(\mathcal{F}(\mathbf{x}), \|\mathcal{H}(\mathbf{x})\|) | \hat{k} \in [1, \ldots, k] : \mathcal{F}(\mathbf{x}_{\hat{k}}) - \beta\hat{\Delta}_k^2 \leq \mathcal{F}(\mathbf{x}) \text{ and } (1 - \beta\hat{\Delta}_k^2)\|\mathcal{H}(\mathbf{x}_{\hat{k}})\| \leq \|\mathcal{H}(\mathbf{x})\|, (\mathcal{F}(\mathbf{x}), \|\mathcal{H}(\mathbf{x})\|) \notin F\Big\}, \tag{A.10}$$

in which no points are considered acceptable to the filter. (The restricted area is shown graphically in Figure A-1, right.)

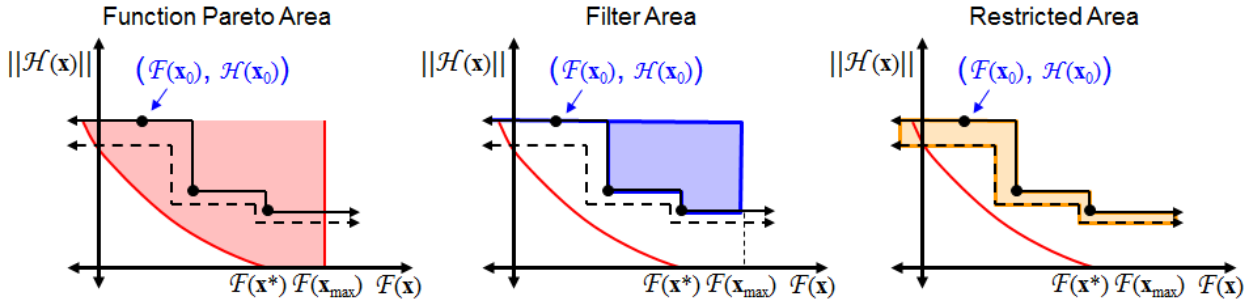


Figure A-1: Graphic function Pareto area, filtered area, and unacceptable but not filtered area.

The size of the area enclosed by the filter, F, increases with every iterate accepted by the filter. Ordering points on the filter by decreasing constraint violation (as on Figure 5-4) the area that must be added to F by accepting a point on the filter between two adjacent points i and $i+1$ is $\beta\Delta_k^2 \times \beta\Delta_k^2\|\mathcal{H}(\mathbf{x}_i)\|$. This holds for all intervals except for the final, between $(\mathcal{F}(\mathbf{x}_l), \|\mathcal{H}(\mathbf{x}_l)\|)$ and $(\mathcal{F}(\mathbf{x}_{\max}), 0)$, where l is the number of points accepted by the filter. So, for every iterate accepted to the filter with $\|\mathcal{H}(\mathbf{x}_k)\| \geq \|\mathcal{H}(\mathbf{x}_l)\|$ the filter area increases by at least $\beta^2\Delta_k^4\|\mathcal{H}(\mathbf{x}_{l-1})\|$. However, because the area enclosed by the filter is contained by the area enclosed by the Pareto front, $F \subseteq G$, the sum of the areas added to the filter must be finite. Therefore if an infinite number of iterates are accepted to the filter with $\|\mathcal{H}(\mathbf{x}_k)\| \geq \|\mathcal{H}(\mathbf{x}_l)\|$, then the conclusion in (5.37) holds. Similarly, if an infinite number of iterates are accepted to the filter with $\|\mathcal{H}(\mathbf{x}_k)\| < \|\mathcal{H}(\mathbf{x}_l)\|$ then for $\mathbf{x}_k$ to be accepted by the filter the constraint violation at $\mathbf{x}_k$ must sufficiently reduce the constraint violation

at $\mathbf{x}_l$, i.e. $\|\mathcal{H}(\mathbf{x}_k)\| \leq (1-\beta\Delta_k^2)\|\mathcal{H}(\mathbf{x}_l)\|$. So, the sum over the points added to the filter with $\|\mathcal{H}(\mathbf{x}_k)\| < \|\mathcal{H}(\mathbf{x}_l)\|$, $\sum_{l=1}^{\infty}\beta\Delta_l^2\|\mathcal{H}(\mathbf{x}_{l-1})\| \leq \|\mathcal{H}(\mathbf{x}_0)\|$, is bounded and the conclusion of (5.37) holds. Therefore, if an infinite number of points are accepted by the filter then $\lim_{k\to\infty}\|\mathcal{H}(\mathbf{x}_k)\|\Delta_k = 0$ and case (ii) is complete.

We now demonstrate that for each $\mathbf{x}_k \in \Omega$ there is a minimum size of the trust region for which the filter accepts a nearby point with lower constraint violation. The filter accepts any step provided that $\|\mathcal{H}(\mathbf{x}_k+\mathbf{s}_k)\| \leq (1-\beta\hat{\Delta}_k^2)\|\mathcal{H}(\mathbf{x}_k)\|$. Meanwhile, Eq. 5.36 demonstrated that for each $\Delta_k \in (0, \delta_2(\mathbf{x}_k))$,

$$\|\mathcal{H}(\mathbf{x}_k+\mathbf{s}_k)\| \leq \|\mathcal{H}(\mathbf{x}_k)\|\sqrt{1-\frac{\eta\kappa_{fcd}\Delta_k\varrho(\mathbf{x}_k)}{\|\mathcal{H}(\mathbf{x}_k)\|^2}}.$$

Therefore, if $\sqrt{1-\frac{\eta\kappa_{fcd}\Delta_k\varrho(\mathbf{x}_k)}{\|\mathcal{H}(\mathbf{x}_k)\|^2}} \leq 1-\beta\hat{\Delta}_k^2$, then $\mathbf{x}_k+\mathbf{s}_k$ is acceptable to the filter, $\mathbf{x}_{k+1} \neq \mathbf{x}_k$ and $\Delta_{k+1} > \Delta_k$. As both the left and right sides of the inequality are less than one, we can establish that $\mathbf{x}_k+\mathbf{s}_k$ is accepted by the filter when $\frac{\eta\kappa_{fcd}\varrho(\mathbf{x}_k)}{\|\mathcal{H}(\mathbf{x}_k)\|^2} \geq 2\beta\Delta_k - \beta^2\Delta_k^3$. As the positive real root(s) of

$$\Delta_k\left(1-\frac{\beta}{2}\Delta_k^2\right) = \frac{\eta\kappa_{fcd}\kappa_\tau}{2\beta\|\mathcal{H}(\mathbf{x}_k)\|}$$

are between $\sqrt{2}$ and $+\infty$, each $\Delta_k \in (0, \min\{\delta_2(\mathbf{x}_k), \sqrt{2}\})$ ensures $\mathbf{x}_k+\mathbf{s}_k$ is acceptable to the filter and that $\Delta_{k+1} > \Delta_k$. The result is that the constraint violation of the sequence of iterates generated by Algorithm 5.2 goes to zero.

A.3 Objective Function Improvement

To demonstrate that Algorithm 5.2 can find an optimal solution it is necessary to demonstrate that Algorithm 5.2 decreases the value of the objective function if the current iterate is not optimal. The challenge is that the normal step can cause the value of the objective function to increase. We therefore must show that the increase in the objective function caused by the normal step must become sufficiently small such that the tangential step can decrease

the value of the objective function. We demonstrate this by considering the trust region step, $\mathbf{s}_k$, projected into two directions, that in the surrogate Jacobian range-space and that in the nullspace. We begin by showing that assumption AAO4, ensures that for each Δ_k in $(0, \kappa_\tau/\kappa_{\nabla\mathcal{H}})$ the surrogate constraint Jacobian, $\nabla\bar{\mathcal{H}}(\mathbf{x}_k)$, has full row rank. For all $\mathbf{p} \neq 0$,

$$\begin{aligned}
\|\nabla\mathcal{H}(\mathbf{x}_k)^\top \mathbf{p}\| &= \left\| \left[\nabla\mathcal{H}(\mathbf{x}_k) + \nabla\bar{\mathcal{H}}(\mathbf{x}_k) - \nabla\bar{\mathcal{H}}(\mathbf{x}_k)\right]^\top \mathbf{p} \right\| \\
&\leq \left\| \left[\nabla\mathcal{H}(\mathbf{x}_k) - \nabla\bar{\mathcal{H}}(\mathbf{x}_k)\right]^\top \mathbf{p} \right\| + \|\nabla\bar{\mathcal{H}}(\mathbf{x}_k)^\top \mathbf{p}\| \\
&\leq \kappa_{\nabla\mathcal{H}}\Delta_k\|\mathbf{p}\| + \|\nabla\bar{\mathcal{H}}(\mathbf{x}_k)^\top \mathbf{p}\|,
\end{aligned}$$

therefore, $\min\{\|\nabla\bar{\mathcal{H}}(\mathbf{x}_k)^\top \mathbf{p}\|/\|\mathbf{p}\|\} > \kappa_\tau - \kappa_{\nabla\mathcal{H}}\Delta_k$, and for each Δ_k in $(0, \kappa_\tau/\kappa_{\nabla\mathcal{H}})$ the surrogate constraint Jacobian has full rank.

From here, we show that when the constraint violation is small the component of the normal step in the range-space of the constraint Jacobian is in the strict interior of the trust region. Specifically, let $\bar{Q}_k \in \mathbb{R}^{n\times n}$ be an orthogonal projection onto the range-space of $\nabla\bar{\mathcal{H}}(\mathbf{x}_k)$, then for a small ϵ and $0 \leq \|\bar{\mathcal{H}}(\mathbf{x}_k)\| < \epsilon$ we have $\|\bar{Q}_k\mathbf{s}_k^\perp\| < \Delta_k$. Consider a quadratic approximation to the squared constraint violation objective function in (5.25). When Δ_k is small we may use the exact solution to the quadratic approximation of (5.25) and the sufficient decrease condition in Eq. 5.26 to establish bounds for $\bar{Q}_k\mathbf{s}_k^\perp$. We then show that as the constraint violation approaches zero the component of $\mathbf{s}_k^\perp$ in the range-space of the constraint Jacobian is in the interior of the trust region, i.e. $\|\bar{Q}_k\mathbf{s}_k^\perp\| < \Delta_k$.

The quadratic approximation of objective function in (5.25) is

$$\begin{aligned}
\frac{1}{2}\|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}^\perp)\|_2^2 \approx & \\
&\frac{1}{2}\left(\mathbf{s}^\perp\right)^\top \left[\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k) + \sum_{i=1}^{m}\bar{h}_k^i(\mathbf{x}_k)\nabla^2\bar{h}_k^i(\mathbf{x}_k)\right]\mathbf{s}^\perp \\
&+ \left(\mathbf{s}^\perp\right)^\top \nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\bar{\mathcal{H}}_k(\mathbf{x}_k) + \frac{1}{2}\bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\bar{\mathcal{H}}_k(\mathbf{x}_k),
\end{aligned} \tag{A.11}$$

and for simplicity we denote the Hessian with the matrix, $\bar{A}_k = \nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k) +$

$\sum_{i=1}^{m} \bar{h}_k^i(\mathbf{x}_k)\nabla^2\bar{h}_k^i(\mathbf{x}_k)$, and the gradient at $\mathbf{s}^\perp = 0$ as, $\bar{B}_k = \nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\bar{\mathcal{H}}_k(\mathbf{x}_k)$,

$$\frac{1}{2}\|\bar{\mathcal{H}}_k(\mathbf{x}_k + \mathbf{s}^\perp)\|_2^2 \approx \frac{1}{2}\left(\mathbf{s}^\perp\right)^\top \bar{A}_k\mathbf{s}^\perp + \mathbf{s}^{\perp T}\bar{B}_k + \frac{1}{2}\bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\bar{\mathcal{H}}_k(\mathbf{x}_k). \tag{A.12}$$

Let $\tilde{\mathbf{s}}_k^{\perp *}$ be such that $\|\bar{\mathcal{H}}_k(\mathbf{x}_k + \tilde{\mathbf{s}}_k^{\perp *})\|_2^2 = 0$. As $\mathbf{y}_k = \mathbf{x}_k + \tilde{\mathbf{s}}_k^{\perp *}$ is a local minimum, then there exists a region with $\|\bar{\mathcal{H}}(\mathbf{y}_k)\| < \epsilon_1$ where $\bar{A}_k$ is positive semi-definite. When $\bar{A}_k$ is positive semi-definite, the minimal norm exact minimizer of this quadratic is $\tilde{\mathbf{s}}_k^{\perp *} = -\bar{A}_k^\dagger\bar{B}_k$, where $(\cdot)^\dagger$ denotes the Moore-Penrose pseudo-inverse. (Note: even if the normal step problem were solved exactly the trust region constraint enforces $\|\mathbf{s}_k^{\perp *}\| \leq \|\tilde{\mathbf{s}}_k^{\perp *}\|$.) For this quadratic approximation, at the exact solution we have $\|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \tilde{\mathbf{s}}_k^{\perp *})\|_2^2 = \left[\bar{A}_k^\dagger\bar{B}_k\right]^\top \bar{B}_k$.

We now consider $\tilde{\mathbf{s}}_k^{\perp *}$ in only the m-dimensional range-space of $\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)$. We use the truncated left singular vectors of $\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)$ as a basis for the range-space. Let

$$\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k) = \underbrace{\bar{U}_k}_{m\times n}\underbrace{\bar{\Sigma}_k}_{n\times n}\underbrace{\bar{V}_k^\top}_{n\times n}, \tag{A.13}$$

then we can represent the component of $\mathbf{s}_k^\perp$ in the range-space of $\nabla\bar{\mathcal{H}}_k(\mathbf{x}_k)$ as the m-dimensional vector $\check{\mathbf{s}}_k$. We then substitute $\mathbf{s}_k^\perp = \bar{U}_k^\top\check{\mathbf{s}}_k$ into Eq. A.12,

$$\|\bar{\mathcal{H}}_k(\mathbf{x}_k + \bar{U}_k^\top\check{\mathbf{s}})\|_2^2 \approx \check{\mathbf{s}}^\top\bar{U}_k\bar{A}_k\bar{U}_k^\top\check{\mathbf{s}} + 2\check{\mathbf{s}}^\top\bar{U}_k\bar{B}_k + \bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\bar{\mathcal{H}}_k(\mathbf{x}_k). \tag{A.14}$$

Furthermore, there exists an $\epsilon_2 \leq \epsilon_1$ such that $\|\mathcal{H}(\mathbf{y}_k)\| < \epsilon_2$ ensures that $\bar{U}_k\bar{A}_k\bar{U}_k$ is positive definite. Let the Cholesky Decomposition of $\bar{U}_k\bar{A}_k\bar{U}_k^\top = \bar{\Gamma}_k^\top\bar{\Gamma}_k$. Now substituting $\check{\mathbf{s}}_k = \bar{\Gamma}_k^{-1}\mathfrak{s}$ into the quadratic approximation, Eq. A.14,

$$\|\bar{\mathcal{H}}_k(\mathbf{x}_k + \bar{U}_k^\top\bar{L}_k^{-1}\mathfrak{s})\|_2^2 \approx \mathfrak{s}^\top\mathfrak{s} + 2\mathfrak{s}^\top\left(\bar{L}_k^{-1}\right)^\top\bar{U}_k\bar{B}_k + \bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\bar{\mathcal{H}}_k(\mathbf{x}_k). \tag{A.15}$$

From the sufficient decrease condition for the normal step, Eq. 5.26, we know the solution to Eq. 5.25 satisfies,

$$\|\bar{\mathcal{H}}_k(\mathbf{x}_k)\|_2^2 - \|\bar{\mathcal{H}}_k(\mathbf{x}_k + \bar{U}_k^\top\bar{L}_k^{-1}\mathfrak{s})\|_2^2 \geq \kappa_{amm-H}\left[\bar{A}_k^\dagger\bar{B}_k\right]^\top\bar{B}_k. \tag{A.16}$$

By completing the square in Eq. A.15 we see that the set of solutions that provide the sufficient decrease in (A.16) satisfy

$$\left[\mathfrak{s} + \left(\bar{L}_k^{-1}\right)^\top \bar{U}_k \bar{B}_k\right]^\top \left[\mathfrak{s} + \left(\bar{L}_k^{-1}\right)^\top \bar{U}_k \bar{B}_k\right] \leq \bar{B}_k^\top \bar{U}_k^\top \bar{L}_k^{-1} \left(\bar{L}_k^{-1}\right)^\top \bar{U}_k \bar{B}_k - \kappa_{amm-H} \left[\bar{A}_k^\dagger \bar{B}_k\right]^\top \bar{B}_k. \tag{A.17}$$

It follows that $\mathfrak{s}$ has bounded norm,

$$\|\mathfrak{s}\|_2 \leq \| \left(\bar{L}_k^{-1}\right)^\top \bar{U}_k \bar{B}_k\|_2 + \sqrt{\bar{B}_k^\top \left[\bar{U}_k^\top \bar{L}_k^{-1} \left(\bar{L}_k^{-1}\right)^\top \bar{U}_k - \kappa_{amm-H} \left(\bar{A}_k^\dagger\right)^\top\right] \bar{B}_k}, \tag{A.18}$$

and because $\bar{B}_k$ is proportional to the constraint violation, we have established that for each Δ_k in $(0, \kappa_\tau/\kappa_{\nabla\mathcal{H}})$ and $\|\mathcal{H}(\mathbf{x}_k)\|$ sufficiently small then $\|\bar{Q}_k \mathbf{s}_k^\perp\| \leq \kappa_{constraint} \|\mathcal{H}(\mathbf{x}_k)\|$ for a positive constant, $\kappa_{constraint}$. Accordingly, for $\|\mathcal{H}(\mathbf{x}_k)\|$ sufficiently small we have that $\|\bar{Q}_k \mathbf{s}_k^\perp\| < \Delta_k$, where we recall that $\bar{Q}_k \in \mathbb{R}^{n\times n}$ is the orthogonal projection onto the range-space of $\nabla\bar{\mathcal{H}}(\mathbf{x}_k)$.

We now demonstrate that, where $\bar{P}_k$, as given in (5.35), is an orthonormal projection matrix into the nullspace of $\nabla\bar{\mathcal{H}}(\mathbf{x}_k)$ there exists an ϵ such that if $\|\mathcal{H}(\mathbf{x}_k)\| < \epsilon$ and Δ_k is sufficiently small, then (5.39),

$$\bar{\mathcal{F}}_k(\mathbf{x}_k) - \bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k) \geq \kappa_{fcd2} \|\bar{P}_k \nabla\bar{\mathcal{F}}_k(\mathbf{x}_k)\| \Delta_k$$

holds. We demonstrate the decrease in (5.39) in two parts, first, we develop a simple optimization problem that will enable us to place an upper bound on the value of the exact solution to the tangential step optimization problem, (5.32). We then combine this upper bound with the sufficient decrease requirement in Eq. 5.33, which enables us to enforce an upper bound for the value of the surrogate objective at $\mathbf{s}_k$. We then show the decrease in (5.39) is ensured for $\|\mathcal{H}(\mathbf{x}_k)\| < \epsilon$ and Δ_k is sufficiently small. We also note that if in lieu of the Gauss-Newton constraint we use $\|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\|^2 \leq \|\mathcal{H}(\mathbf{x}_k)\|^2 - \eta \left[\|\mathcal{H}(\mathbf{x}_k)\|^2 - \|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k^\perp)\|^2\right]$, (5.28), then this set is also convex for $\|\mathcal{H}(\mathbf{x}_k)\| < \epsilon$ and this analysis will hold, though with different

constants.

Let this convex constraint set in (5.32) be denoted E. It is contained between the hyperplane in the nullspace of $\nabla\bar{\mathcal{H}}(\mathbf{x}_k)$ located at $\eta\bar{Q}_k\mathbf{s}_k^{\perp}$ and the trust region,

$$E = \left\{\mathbf{s} | \bar{Q}_k\mathbf{s} > \eta\bar{Q}_k\mathbf{s}_k^{\perp} \text{ and } \|\mathbf{s}\| \leq \Delta_k\right\} \tag{A.19}$$

(see Figure A-2). Because E is convex, from any starting point in E it is possible to reach the minimum of $\bar{\mathcal{F}}(\mathbf{s})$ for all $s \in E$.

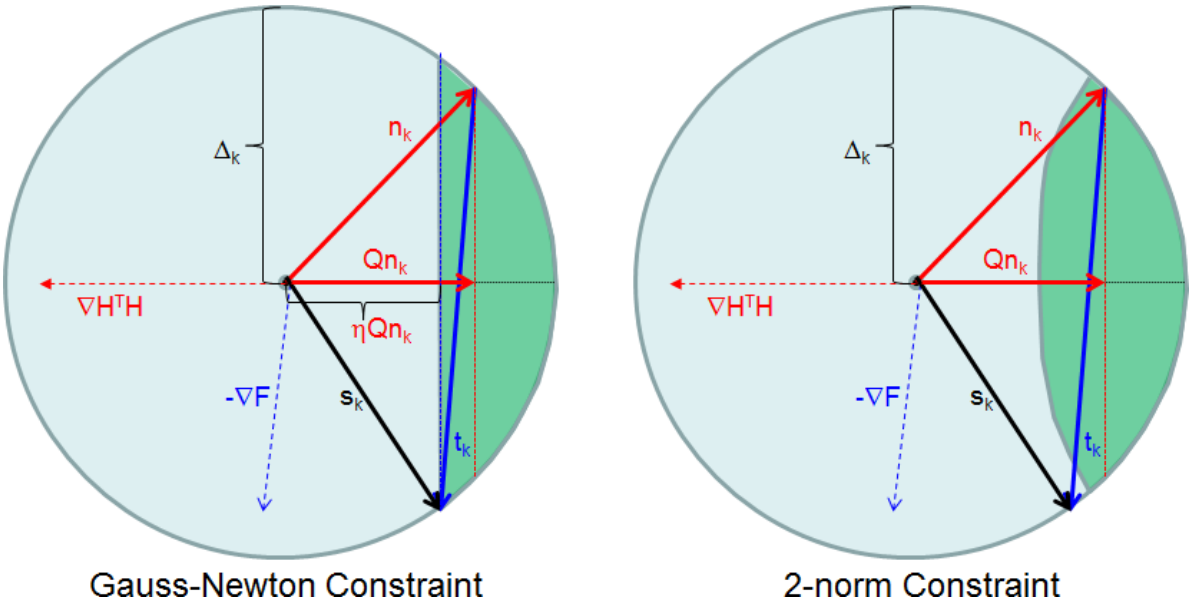


Figure A-2: Schematic of vectors and constraint Jacobian nullspace within a trust region. Also highlighted (in light green) is the convex set formed by the constraints in Eq. 5.32 using the Gauss-Newton constraint (left) and the quadratic approximation to the 2-norm of the constraint violation (right).

Therefore we consider minimizing the surrogate objective function within the nullspace of the surrogate constraint Jacobian and the trust region. This can be written as an optimization problem to find the step $\mathbf{s}_k^N$ minimizing the surrogate objective function at a point

located in the nullspace of $\nabla\bar{\mathcal{H}}(\mathbf{x}_k)$ and with length less than $\sqrt{1-\eta^2}\Delta_k$,

$$\begin{aligned} \min_{\mathbf{s}_k^N} \quad & \bar{\mathcal{F}}_k(\mathbf{x}_k + \eta\bar{Q}_k\mathbf{s}_k^{\perp} + \mathbf{s}_k^N) \\ \text{s.t.} \quad & \|\bar{P}_k\mathbf{s}_k^N\| \leq \sqrt{1-\eta^2}\Delta_k \\ & \|\bar{Q}_k\mathbf{s}_k^N\| = 0. \end{aligned} \tag{A.20}$$

The constraint set in this optimization problem is a subset of E, so the minimum of this optimization problem provides an upper bound for the minimal value of $\bar{\mathcal{F}}(\mathbf{s})$ for all $s \in E$. Moreover, from [31, Lemma 15.4.2], the exact solution to (A.20) satisfies

$$\begin{aligned} &\bar{\mathcal{F}}_k(\mathbf{x}_k + \eta\bar{Q}_k\mathbf{s}_k^{\perp}) - \bar{\mathcal{F}}_k(\mathbf{x}_k + \eta\bar{Q}_k\mathbf{s}_k^{\perp} + \mathbf{s}_k^N) \geq \\ &\qquad \frac{1}{2}\|\bar{P}_k\nabla\bar{\mathcal{F}}_k(\mathbf{x}_k + \eta\bar{Q}_k\mathbf{s}_k^{\perp})\| \min\left\{ \frac{\sqrt{1-\eta^2}\Delta_k}{\|\bar{P}_k\|}, \frac{\|\bar{P}_k\nabla\bar{\mathcal{F}}_k(\mathbf{x}_k + \eta\bar{Q}_k\mathbf{s}_k^{\perp})\|}{\beta_T} \right\}, \end{aligned} \tag{A.21}$$

where β_T is an upper bound for the Hessian of $\bar{\mathcal{F}}(\mathbf{x})$ projected into the nullspace of $\nabla\bar{\mathcal{H}}(\mathbf{x}_k)$ within the trust region. We may now use $\bar{\mathcal{F}}_k(\mathbf{x}_k + \eta\bar{Q}_k\mathbf{s}_k^{\perp} + \mathbf{s}_k^N)$ as an upper bound for $\bar{\mathcal{F}}_k(\mathbf{x}_k + \eta\bar{Q}_k\mathbf{s}_k^{\perp} + \mathbf{s}_k^{\|*})$ which is the exact solution to Eq. 5.32.

When Δ_k is sufficiently small that the min operator in (A.21) returns Δ_k and $\|\mathcal{H}(\mathbf{x}_k)\| < \epsilon$, then the exact solution to Eq. 5.32, $\mathbf{s}_k^{\|*}$ satisfies,

$$\begin{aligned} \bar{\mathcal{F}}_k(\mathbf{x}_k) - \bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\|*}) &= \bar{\mathcal{F}}_k(\mathbf{x}_k) - \bar{\mathcal{F}}_k(\mathbf{x}_k + \eta\bar{Q}_k\mathbf{s}_k^{\perp}) + \bar{\mathcal{F}}_k(\mathbf{x}_k + \eta\bar{Q}_k\mathbf{s}_k^{\perp}) - \bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\|*}) \\ &\geq \bar{\mathcal{F}}_k(\mathbf{x}_k + \eta\bar{Q}_k\mathbf{s}_k^{\perp}) - \bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k^{\perp} + \mathbf{s}_k^{\|*}) - \|\bar{\mathcal{F}}_k(\mathbf{x}_k) - \bar{\mathcal{F}}_k(\mathbf{x}_k + \eta\bar{Q}_k\mathbf{s}_k^{\perp})\| \\ &\geq \frac{1}{2}\|\bar{P}_k\nabla\bar{\mathcal{F}}_k(\mathbf{x}_k + \eta\bar{Q}_k\mathbf{s}_k^{\perp})\|\Delta_k - \eta\kappa_{constraint}\|\mathcal{H}(\mathbf{x}_k)\| \\ &\geq \frac{1}{2}\|\bar{P}_k\left[\nabla\bar{\mathcal{F}}_k(\mathbf{x}_k + \eta\bar{Q}_k\mathbf{s}_k^{\perp}) - \nabla\bar{\mathcal{F}}_k(\mathbf{x}_k) + \nabla\bar{\mathcal{F}}_k(\mathbf{x}_k)\right]\|\Delta_k \\ &\qquad -\eta\kappa_{constraint}\|\mathcal{H}(\mathbf{x}_k)\| \\ &\geq \frac{\sqrt{1-\eta^2}}{2}\|\bar{P}_k\nabla\bar{\mathcal{F}}_k(\mathbf{x}_k)\|\Delta_k - \kappa_f\Delta_k^2 - \eta\kappa_{constraint}\|\mathcal{H}(\mathbf{x}_k)\|. \end{aligned}$$

In addition, a bound for the maximum increase in the surrogate objective between $\mathbf{x}_k$ and

$\mathbf{x}_k + \mathbf{s}_k^\perp$ is

$$\begin{aligned}
|\bar{\mathcal{F}}(\mathbf{x}_k + \mathbf{s}_k^\perp) - \bar{\mathcal{F}}(\mathbf{x}_k)| &= |\nabla\bar{\mathcal{F}}(\mathbf{x}_k)^\top \mathbf{s}_k^\perp + \frac{1}{2}\mathbf{s}_k^{\perp T}\nabla^2\bar{\mathcal{F}}(\mathbf{x}_k + t\mathbf{s}_k^\perp)\mathbf{s}_k^\perp| \\
&\leq |\nabla\bar{\mathcal{F}}(\mathbf{x}_k)^\top \mathbf{s}_k^\perp| + (\kappa_g + \kappa_{fg})\|\mathbf{s}_k^\perp\|^2 \\
&\leq |\bar{Q}_k\nabla\bar{\mathcal{F}}(\mathbf{x}_k)^\top(\bar{Q}_k\mathbf{s}_k^\perp)| + |\bar{P}_k\nabla\bar{\mathcal{F}}(\mathbf{x}_k)^\top(\bar{P}_k\mathbf{s}_k^\perp)| + (\kappa_g + \kappa_{fg})\Delta_k^2 \\
&\leq \|\bar{Q}_k\nabla\bar{\mathcal{F}}(\mathbf{x}_k)\|\|\bar{Q}_k\mathbf{s}_k^\perp\| + \|\bar{P}_k\nabla\bar{\mathcal{F}}(\mathbf{x}_k)\|\Delta_k + (\kappa_g + \kappa_{fg})\Delta_k^2 \\
&\leq \|\bar{Q}_k\nabla\bar{\mathcal{F}}(\mathbf{x}_k)\|\kappa_{constraint}\|\mathcal{H}(\mathbf{x}_k)\| + \|\bar{P}_k\nabla\bar{\mathcal{F}}(\mathbf{x}_k)\|\Delta_k + (\kappa_g + \kappa_{fg})\Delta_k^2.
\end{aligned}$$

For large k, $\|\mathcal{H}(\mathbf{x}_k)\|$ becomes arbitrarily small, and similarly for small Δ_k, Δ_k^2 becomes small compared with Δ_k. Therefore for large k and small Δ_k there exists an arbitrarily small positive constant, κ_1, such that,

$$\|\bar{Q}_k\nabla\bar{\mathcal{F}}(\mathbf{x}_k)\|\kappa_{constraint}\|\mathcal{H}(\mathbf{x}_k)\| + (\kappa_g + \kappa_{fg})\Delta_k^2 \leq \kappa_1\|\bar{P}_k\nabla\bar{\mathcal{F}}(\mathbf{x}_k)\|\Delta_k. \tag{A.22}$$

We have now shown that for large k and small Δ_k that there exists a $\mathbf{s}_k^{\|*}$ which reduces the surrogate objective at least $\bar{\mathcal{F}}_k(\mathbf{x}_k) - \bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k^\perp + \mathbf{s}_k^{\|*}) \geq \left(\frac{\sqrt{1-\eta^2}}{2} - \kappa_1\right)\|\bar{P}_k\nabla\bar{\mathcal{F}}_k(\mathbf{x}_k)\|\Delta_k$. We have also shown that the largest increase in the surrogate objective function caused by the normal step is at most $|\bar{\mathcal{F}}(\mathbf{x}_k + \mathbf{s}_k^\perp) - \bar{\mathcal{F}}(\mathbf{x}_k)| \leq (1 + \kappa_1)\|\bar{P}_k\nabla\bar{\mathcal{F}}(\mathbf{x}_k)\|\Delta_k$. Therefore, if κ_{amm-F} in the sufficient decrease requirement, Eq. 5.33, is greater than $\frac{2+\kappa_1}{2+\sqrt{1-\eta^2}}$ then the surrogate objective function is decreased at least proportionally to the projection of the surrogate objective gradient into the surrogate constraint Jacobian, i.e. such that $\bar{\mathcal{F}}_k(\mathbf{x}_k) - \bar{\mathcal{F}}_k(\mathbf{x}_k + \mathbf{s}_k) \geq \kappa_{fcd2}\|\bar{P}_k\nabla\bar{\mathcal{F}}_k(\mathbf{x}_k)\|\Delta_k$, where $\kappa_{fcd2} \in (0, 1)$.

We now establish that for each trust region sufficiently small, there must be a decrease in the objective function proportional to the actual optimality condition, $\|P_k\nabla\mathcal{F}(\mathbf{x}_k)\|$. By bounding the error $\|P_k\nabla\mathcal{F}(\mathbf{x}_k) - \bar{P}_k\nabla\bar{\mathcal{F}}(\mathbf{x}_k)\|$, we are able to show that for small Δ_k, Eq. 5.39 ensures (5.41), that

$$\mathcal{F}(\mathbf{x}_k) - \mathcal{F}(\mathbf{x}_k + \mathbf{s}_k) \geq \frac{\kappa_{fcd2}}{2}\|P_k\nabla\mathcal{F}(\mathbf{x}_k)\|\Delta_k.$$

We begin with the identity,

$$P_k \nabla \mathcal{F}(\mathbf{x}_k) - \bar{P}_k \nabla \bar{\mathcal{F}}(\mathbf{x}_k) = \left[P_k - \bar{P}_k\right] \nabla \mathcal{F}(\mathbf{x}_k) - \bar{P}_k \left[\nabla \bar{\mathcal{F}}(\mathbf{x}_k) - \nabla \mathcal{F}(\mathbf{x}_k)\right].$$

Since $\nabla \bar{\mathcal{H}}_k(\mathbf{x}_k)$ has full rank for each $\Delta_k \in (0, \frac{1}{2}\kappa_\tau / \kappa_{\nabla \mathcal{H}})$, assumption AAO1 tells us that $\|\bar{P}_k\|_2$ is bounded, and by the definition of a fully linear model,

$$\|P_k \nabla \mathcal{F}(\mathbf{x}_k) - \bar{P}_k \nabla \bar{\mathcal{F}}(\mathbf{x}_k)\| \leq \left[P_k - \bar{P}_k\right] \nabla \mathcal{F}(\mathbf{x}_k) + \kappa_1(\mathbf{x}_k)\Delta_k.$$

We now develop a bound for $\|P_k - \bar{P}_k\|$,

$$\begin{aligned}
&P_k - \bar{P}_k = \\
&\quad \nabla \mathcal{H}(\mathbf{x}_k)^\top \left(\nabla \mathcal{H}(\mathbf{x}_k) \nabla \mathcal{H}(\mathbf{x}_k)^\top\right)^{-1} \nabla \mathcal{H}(\mathbf{x}_k) - \nabla \bar{\mathcal{H}}_k(\mathbf{x}_k)^\top \left(\nabla \bar{\mathcal{H}}_k(\mathbf{x}_k) \nabla \bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\right)^{-1} \nabla \bar{\mathcal{H}}_k(\mathbf{x}_k) \\
&\quad = \left[\nabla \mathcal{H}(\mathbf{x}_k) - \nabla \bar{\mathcal{H}}_k(\mathbf{x}_k)\right]^\top \left(\nabla \bar{\mathcal{H}}_k(\mathbf{x}_k) \nabla \bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\right)^{-1} \nabla \bar{\mathcal{H}}_k(\mathbf{x}_k) \\
&\qquad + \nabla \mathcal{H}(\mathbf{x}_k)^\top \left[\left(\nabla \mathcal{H}(\mathbf{x}_k) \nabla \mathcal{H}(\mathbf{x}_k)^\top\right)^{-1} - \left(\nabla \bar{\mathcal{H}}_k(\mathbf{x}_k) \nabla \bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\right)^{-1}\right] \nabla \bar{\mathcal{H}}_k(\mathbf{x}_k) \\
&\qquad + \nabla \mathcal{H}(\mathbf{x}_k)^\top \left(\nabla \mathcal{H}(\mathbf{x}_k) \nabla \mathcal{H}(\mathbf{x}_k)^\top\right)^{-1} \left[\nabla \mathcal{H}(\mathbf{x}_k) - \nabla \bar{\mathcal{H}}_k(\mathbf{x}_k)\right].
\end{aligned}$$

Using the definition of a fully linear model, the bounds in assumption AAO1, and the full rank assumption for $\nabla \mathcal{H}(\mathbf{x}_k)$ then

$$\begin{aligned}
&\|P_k - \bar{P}_k\| \leq \\
&\quad \kappa_2(\mathbf{x}_k)\Delta_k + \kappa_3(\mathbf{x}_k) \left[\left(\nabla \mathcal{H}(\mathbf{x}_k) \nabla \mathcal{H}(\mathbf{x}_k)^\top\right)^{-1} - \left(\nabla \bar{\mathcal{H}}_k(\mathbf{x}_k) \nabla \bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\right)^{-1}\right] + \kappa_4(\mathbf{x}_k)\Delta_k.
\end{aligned}$$

We use the identity, $A^{-1} + B^{-1} = A^{-1}(A + B)B^{-1}$ [103], in order to establish the bound,

$$\begin{aligned}
&\| \left(\nabla \mathcal{H}(\mathbf{x}_k) \nabla \mathcal{H}(\mathbf{x}_k)^\top\right)^{-1} - \left(\nabla \bar{\mathcal{H}}_k(\mathbf{x}_k) \nabla \bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\right)^{-1} \| \\
&\qquad \leq \frac{1}{(\kappa_\tau - \kappa_{\nabla \mathcal{H}} \Delta_k)^2} \|\nabla \mathcal{H}(\mathbf{x}_k) \nabla \mathcal{H}(\mathbf{x}_k)^\top - \nabla \bar{\mathcal{H}}_k(\mathbf{x}_k) \nabla \bar{\mathcal{H}}_k(\mathbf{x}_k)^\top\| \frac{1}{\kappa_\tau^2} \\
&\qquad \leq \kappa_5(\mathbf{x}_k)\Delta_k.
\end{aligned}$$

As, $\|P_k - \bar{P}_k\| \leq \kappa_6(\mathbf{x}_k)\Delta_k$, and $\|\mathcal{F}(\mathbf{x}_k)\|$ is bounded, we have established that for $\Delta_k <$

$\frac{1}{2}\kappa_\tau/\kappa_{\nabla\mathcal{H}}$,

$$\|P_k\nabla\mathcal{F}(\mathbf{x}_k) - \bar{P}_k\nabla\bar{\mathcal{F}}(\mathbf{x}_k)\| \leq \kappa_7(\mathbf{x}_k)\Delta_k,$$

Furthermore, let the maximum of $\kappa_7(\mathbf{x}_k) \in \Omega$ be κ_8, $\kappa_8 > 0$. Then from Eq. 5.39 and the definition of a fully linear model, Eq. 2.8, we obtain that

$$\mathcal{F}(\mathbf{x}_k) - \mathcal{F}(\mathbf{x}_k + \mathbf{s}_k) \geq \kappa_{fcd2}\|P_k\nabla\mathcal{F}(\mathbf{x}_k)\|\Delta_k - \kappa_\mathcal{F}\Delta_k^2 - \kappa_8\Delta_k^2. \tag{A.23}$$

Accordingly for all Δ_k such that $(\kappa_\mathcal{F} + \kappa_8)\,\Delta_k^2 < \frac{\kappa_{fcd2}}{2}\|P_k\nabla\mathcal{F}(\mathbf{x}_k)\|\Delta_k$, we have established the decrease condition in (5.41).

A.4 Objective Function and Filter Interaction

We have so far shown that the sequence of iterates generated by Algorithm 5.2 decrease the constraint violation to zero, and that when both the constraint violation and trust region are small, then the objective function value decreases proportionally to the trust region size and a measure of optimality. We now consider the case when the constraint violation is small, $\|\mathcal{H}(\mathbf{x}_k)\| < \epsilon$, but $\mathbf{x}_k$ is not optimal, i.e. $\|P_k\nabla\mathcal{F}(\mathbf{x}_k)\| > 0$.

To demonstrate first-order optimality, we will assume for the purpose of contradiction that $\lim_{k\to\infty}\|P_k\nabla\mathcal{F}(\mathbf{x}_k)\| \neq 0$, or, equivalently, that there exists an $\epsilon_3 > 0$ such that $\lim_{k\to\infty}\|P_k\nabla\mathcal{F}(\mathbf{x}_k)\| = \epsilon_3$. We now show that this leads to a lower bound for the size of the trust region, $\Delta_k > \Delta_{\min} > 0$, and a necessary contradiction ensues.

As the current $\mathbf{x}_k$ is always on the current filter, we need to consider two cases, (i) the constraint violation of the current iterate is arbitrarily small but greater than zero, i.e. $\|\mathcal{H}(\mathbf{x}_k)\| \in (0, \epsilon_4)$ and (ii) the iterate exactly satisfies the constraints. (Figure A-3 shows these two potential configurations of the filter, and the index l notes the number of the point on the filter in order of decreasing constraint violation.) To demonstrate Δ_k is bounded from below, we need to show that if $\mathbf{x}_k$ is not first-order optimal then a step computed by Algorithm 5.2, $\mathbf{x}_k + \mathbf{s}_k$, must be accepted by the filter with $\mathcal{F}(\mathbf{x}_k + \mathbf{s}_k) < \mathcal{F}(\mathbf{x}_l)$, where $\mathcal{F}(\mathbf{x}_l)$

is the value of the objective function of the iterate on the filter with the lowest constraint violation.

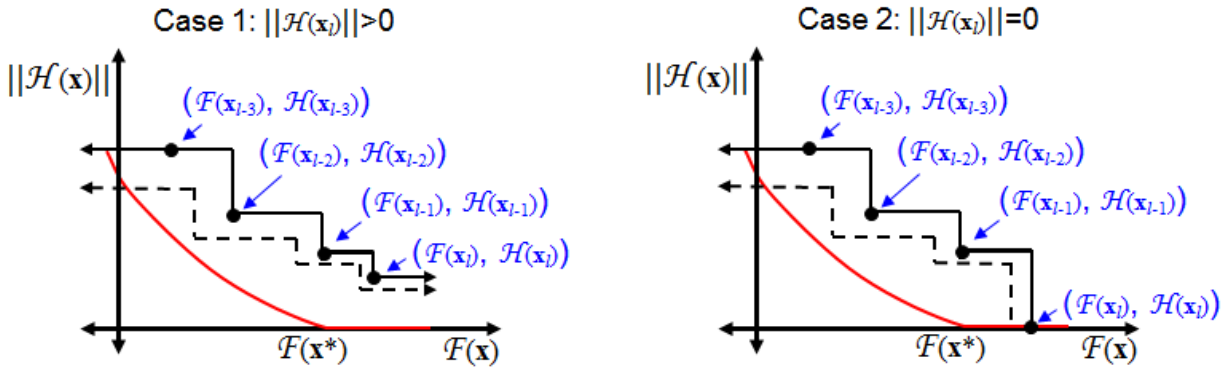


Figure A-3: Possible filter configurations as the constraint violation of $\mathbf{x}_k$ approaches zero.

The filter acceptance criteria requires that any new point is non-dominated, so for both cases (i) and (ii) the acceptance criteria is,

$$\mathcal{F}(\mathbf{x}_k + \mathbf{s}_k) \leq \mathcal{F}(\mathbf{x}_l) - \beta\Delta_k^2 \quad \text{or} \quad \|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\| \leq (1 - \beta\Delta_k^2)\|\mathcal{H}(\mathbf{x}_{l-1})\|. \tag{A.24}$$

Which means in case (i) the accepted point must dominate $\mathbf{x}_l$, both the function value and constraint violation, if only the constraint violation is lower and not also the function value this only shows that the constraint violation at $\mathbf{x}^*$ is zero and not that $\mathbf{x}^*$ is first-order optimal. However, in case (ii) the accepted point must lie on the filter between $\mathbf{x}_{l-1}$ and $\mathbf{x}_l$ or it may dominate $\mathbf{x}_l$.

We have shown for all Δ_k sufficiently small, the objective function decreases proportionally to the optimality metric, $\|P_k\nabla\mathcal{F}(\mathbf{x}_k)\|$, (5.41). Accordingly if Δ_k is small enough that (5.41) holds, then each $\Delta_k < \kappa_{fcd}\|P_k\nabla\mathcal{F}(\mathbf{x}_k)\|/2\beta$ ensures the objective function reduction requirement of the filter is met. However, to show that $\mathbf{x}_k + \mathbf{s}_k$ is non-dominated, we also must show that $\|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\| < (1 - \beta\Delta_k^2)\|\mathcal{H}(\mathbf{x}_{l-1})\|$.

In case (i), $\|\mathcal{H}(\mathbf{x}_k)\| = \|\mathcal{H}(\mathbf{x}_1)\| > 0$, then by (5.36) any $\Delta_k \in (0, \delta_2(\mathbf{x}_k)$ ensures $\|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\|$ is also acceptable to the filter and that $\mathbf{x}_k + \mathbf{s}_k$ is non-dominated. Accordingly, there exists a minimal trust region size for each $\mathbf{x}_k$ in Ω below which the step computed by

Algorithm 5.2 must be non-dominated and acceptable to the filter. Therefore, case (i) is complete.

In case (ii) if $\|\mathcal{H}(\mathbf{x}_k)\| = \|\mathcal{H}(\mathbf{x}_l)\| = 0$, then for $\mathbf{x}_k + \mathbf{s}_k$ to be accepted the filter requires $\|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\| \leq (1 - \beta\Delta_k^2)\|\mathcal{H}(\mathbf{x}_{l-1})\|$. The use of the Gauss-Newton approximation in the tangential step calculation enables the surrogate model constraint violation to be as large as, $\|\bar{\mathcal{H}}(\mathbf{x}_k + \mathbf{s}_k)\| \leq 2\left(\kappa_m\|\mathcal{H}(\mathbf{x}_k)\|\Delta_k^2 + \frac{1}{4}\kappa_m^2\Delta_k^4\right)$. By the definition of a fully linear model, Eq. 2.8, $\|\mathcal{H}(\mathbf{x}_k + \mathbf{s}_k)\| \leq 2\left(\kappa_m\|\mathcal{H}(\mathbf{x}_k)\|\Delta_k^2 + \frac{1}{4}\kappa_m^2\Delta_k^4\right) + \kappa_{\mathcal{H}}\Delta_k^2$. The filter guarantees that $\|\mathcal{H}(\mathbf{x}_{l-1})\| > 0$, therefore a positive Δ_k ensures $2\left(\kappa_m\|\mathcal{H}(\mathbf{x}_k)\|\Delta_k^2 + \frac{1}{4}\kappa_m^2\Delta_k^4\right) + \kappa_{\mathcal{H}}\Delta_k^2 \leq (1 - \beta\Delta_k^2)\|\mathcal{H}(\mathbf{x}_{l-1})\|$. Accordingly, there exists a minimal trust region size for each $\mathbf{x}_k$ in Ω below which the step computed by Algorithm 5.2 must be non-dominated and acceptable to the filter. Therefore, case (ii) is complete.

Assuming $\|P_k\nabla\mathcal{F}(\mathbf{x}_k)\| > 0$, we have now shown that for each $\mathbf{x}_k$ in Ω there exists a minimum trust size, and let $\Delta_{\min}$ be the smallest size of this trust region in Ω. Therefore, for all k sufficiently large, $\mathcal{F}(\mathbf{x}_k) - \mathcal{F}(\mathbf{x}_k + \mathbf{s}_k) \geq \frac{1}{2}\kappa_{fcd2}\epsilon_3\Delta_{\min}$. This is a necessary contradiction as the filter acceptance criteria contains a sufficient decrease condition that requires for $\mathbf{x}_{k+1} \neq \mathbf{x}_k$ then either the constraint violation or the objective function value has been reduced by $\beta\Delta_k^2$. As $\|\mathcal{H}(\mathbf{x})\| = 0$ at all cluster points, then either $\{\Delta_k\} \to 0$ or $\{\mathcal{F}(\mathbf{x}_k)\}$ is a decreasing sequence. However, $\mathcal{F}(\mathbf{x}_k)$ is bounded from below on Ω. Therefore we must also have that $\lim_{k\to\infty}|\mathcal{F}(\mathbf{x}_k) - \mathcal{F}(\mathbf{x}_{k+1})| = 0$. Therefore at any limit point of the sequence of iterates generated by Algorithm 5.2 must have both $\|\mathcal{H}(\mathbf{x}^*)\| = 0$ and $\|P(\mathbf{x}^*)\nabla\mathcal{F}(\mathbf{x}^*)\| = 0$, i.e., that they satisfy the constraints and are first-order optimal. Therefore, the sequence of iterates generated by Algorithm 5.2 asymptotically approaches an optimal solution to (5.24).

Bibliography

[1] M. J. Aftosmis. Solution adaptive cartesian grid methods for aerodynamic flows with complex geometries. In *28th Computational Fluid Dynamics Lecture Series, von Karman Institute for Fluid Dynamics*, Rhode-Saint-Genése, Belgium, March 3-7 1997. Lecture Series 1997-02.

[2] N. M. Alexandrov, J. Dennis, Jr., R. M. Lewis, and V. Torczon. A trust region framework for managing the use of approximation models in optimization. Technical Report CR-201745, NASA, October 1997.

[3] N. M. Alexandrov, R.M. Lewis, C. Gumbert, L. Green, and P. Newman. Optimization with variable-fidelity models applied to wing design. Technical Report CR-209826, NASA, December 1999.

[4] N.M. Alexandrov. Multilevel methods for MDO. In *Multidisciplinary Design Optimization: State of the Art, N.M. Alexandrov and M.Y. Hussaini, editors*, pages 79–89, Philadelphia, PA, January 1997. Society for Industrial and Applied Mathematics.

[5] N.M. Alexandrov, R.M. Lewis, C. Gumbert, L. Green, and P. Newman. Approximation and model management in aerodynamic optimization with variable-fidelity models. *Journal of Aircraft*, 38(6):1093–1101, November-December 2001.

[6] A.C. Antoulas. *Approximation of Large-Scale Dynamical Systems*. SIAM, 2005.

[7] C. Audet and J. E. Dennis, Jr. A pattern search filter method for nonlinear programming without derivatives. *SIAM Journal of Optimization*, 14(4):980–1010, 2004.

[8] C. Audet and J. E. Dennis, Jr. Mesh adaptive direct search algorithms for constrained optimization. *SIAM Journal of Optimization*, 17(1):188–217, 2006.

[9] V. Balabanov, R. Haftka, B. Grossman, W. Mason, and L. Watson. Multi-fidelity response surface model for hsct wing bending material weight. In *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, St. Louis, MO, September 2-4 1998. AIAA 1998-4804.

[10] V. Balabanov and G. Venter. Multi-fidelity optimization with high-fidelity analysis and low-fidelity gradients. In *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Albany, New York, August 30-September 1, 2004. AIAA 2004-4459.

[11] J. F. Bard. *Practical Bilevel Optimization, Algorithms and Applications*. Kluwer Academic Publishers, Dordrecht, 1998.

[12] D. P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Athena Scientific, 1996.

[13] D. P. Bertsekas. *Nonlinear Programming, 2nd ed.* Athena Scientific, 1999.

[14] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

[15] P. T. Boggs and J. E. Dennis, Jr. A stability analysis for perturbed nonlinear iterative methods. *Mathematics of Computation*, 30(134):199–215, April 1976.

[16] A. J. Booker, J. E. Dennis, Jr., P. D. Frank, D. B. Serafini, V. Torczon, and M. W. Trosset. A rigorous framework for optimization of expensive functions by surrogates. *Structural Optimization*, 17(1):1–13, February 1999.

[17] S Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction methods of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[18] R. D. Braun. *Collaborative Optimization: An Architecture for Large-Scale Distributed Design*. PhD thesis, Stanford University, May 1996.

[19] R.G. Carter. On the global convergence of trust region algorithms using inexact gradient information. *SIAM Journal of Numerical Analysis*, 28(1):251–265, 1991.

[20] J. Castro, G. Gray, A. Giunta, and P. Hough. Developing a computationally efficient dynamic multilevel hybrid optimization scheme using multifidelity model interactions. Technical Report SAND2005-7498, Sandia, November 2005.

[21] M. R. Celis, J. E. Dennis, Jr., and R. A. Tapia. A trust region strategy for nonlinear equality constrained optimization. In P. Boggs, R. Byrd, and R. Schnabel, editors, *Numerical Optimization 1984*, pages 71–82. SIAM, 1985.

[22] S. Choi, J. Alonso, and I. Kroo. Multi-fidelity design optimization of low-boom supersonic business jets. In *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Albany, New York, August 30-September 1 2004. AIAA 2004-4371.

[23] H. Chung and J. J. Alonso. Design of a low-boom supersonic business jet using cokriging approximation models. In *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Atlanta, GA, September 2002. AIAA 2002-5598.

[24] H. Chung and J. J. Alonso. Using gradients to construct cokriging approximation models for high-dimensional design optimization problems. In *40th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, NV, January 2002. AIAA 2002-0317.

[25] B. Colson, P. Marcotte, and G. Savard. A trust-region method for nonlinear bilevel programming: Algorithm and computational experience. *Computational Optimization and Applications*, 30:211–227, 2005.

[26] B. Colson, P. Marcotte, and G. Savard. An overview of bilevel optimization. *Annals of Operations Research*, 153:235–256, 2007.

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

[28] A. R. Conn, K. Scheinberg, and L. N. Vicente. *Introduction to Derivative-Free Optimization*. MPS/SIAM Series on Optimization. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2009.

[29] A. R. Conn and L. N. Vicente. Bilevel derivative-free optimization and its application to robust optimization. Technical report, September 2010.

[30] A.R. Conn, N.I. Gould, and Ph.L. Toint. A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds. *Journal of Numerical Analysis*, 28(2):545–572, 1991.

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

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

[33] A.R. Conn, K. Scheinberg, and L. Vicente. Geometry of interpolation sets in derivative free optimization. *Mathematical Programming*, 111(1-2):141–172, 2008.

[34] E. J. Cramer, J. E. Dennis, Jr., P. D. Frank, R. M. Lewis, and G. R Shubin. Problem formulation for multidisciplinary optimization. *SIAM Journal of Optimization*, 4(4):754–776, November 1994.

[35] S. Dempe. *Foundations of Bilevel Programming*. Kluwer Academic Publishers, Dordrecht, 2002.

[36] M. Eldred, A. Giunta, and S. Collis. Second-order corrections for surrogate-based optimization with model hierarchies. 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, 2004. AIAA 2004-4457.

[37] M.S. Eldred and D.M. Dunlavy. Formulations for surrogate-based optimization with data-fit, multifidelity, and reduced-order models. 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Portsmouth, Virginia, 2006. AIAA 2006-7117.

[38] A.V. Fiacco and G.P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. SIAM's Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1990.

[39] R. Fletcher, N. M. Gould, S. Leyffer, Ph. L. Toint, and A. Wächter. Global convergence of a trust-region filter-SQP algorithm for general nonlinear programming. *SIAM Journal of Optimization*, 13(3):635–659, 2002.

[40] R. Fletcher and S. Leyffer. Nonlinear programming without a penalty function. *Mathematical Programming*, 91:239–269, 2002.

[41] R. Fletcher, S. Leyffer, and Ph. L. Toint. On the global convergence of a filter-SQP algorithm. *SIAM Journal of Optimization*, 13(1):44–59, 2002.

[42] A.I.J. Forrester, A. Sóbester, and A.J. Keane. Multi-fidelity optimization via surrogate modelling. *Journal of the Royal Statistical Society*, 463(2):3251–3269, September 2007.

[43] M. Fortin and R. Glowinski. On decomposition-coordination methods using an augmented lagrangian. In *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems, M. Fortin and R. Glowinski (eds.)*, pages 97–146, North-Holland, 1983. Elsevier Science Publishers.

[44] S. E. Gano, J. E. Renaud, and B. Sanders. Hybrid variable fidelity optimization by using a kriging-based scaling function. *AIAA Journal*, 43(11):2422–2430, November 2005.

[45] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: an SQP algorithm for large-scale constrained optimization. *SIAM Journal of Optimization*, 12(4):979–1006, 2002.

[46] R. Glowinski and P. Le Tallec. *Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics*. SIAM, 1989.

[47] J. Golinski. An adaptive optimization system applied to machine synthesis. *Mechanism and Machine Theory*, 8(4):419–436, 1973.

[48] D. Huang, T. Allen, W. Notz, and R. Miller. Sequential Kriging optimization using multiple-fidelity evaluations. *Structural and Multidisciplinary Optimization*, 32:369–382, May 2006.

[49] A. Jameson. Aerodynamic design via control theory. *Journal of Scientific Computing*, 3(3):233–260, September 1988.

[50] D. R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21:345–383, 2001.

[51] D. R. Jones, C. D. Perttunen, and B. E. Stuckmann. Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, October 1993.

[52] D.R. Jones, M. Schonlau, and W.J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13:455–492, 1998.

[53] A. J. Keane. Wing optimization using design of experiment, response surface, and data fusion methods. *Journal of Aircraft*, 40(4):741–750, July-August 2003.

[54] G. J. Kennedy and J. R. R. A. Martins. Parallel solution methods for aerostructural analysis and design optimization. 13th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Fort Worth, TX, September 2010. AIAA 2010-9308.

[55] M. Kennedy and A. O'Hagan. Predicting the output from a complex computer code when fast approximations are available. *Biometrika*, 87(1):1–13, 2000.

[56] M. Kennedy and A. O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society*, 63(3):425–464, 2001.

[57] G. K. W. Kenway, G. J. Kennedy, and J. R. R. A. Martins. A CAD-free approach to high-fidelity aerostructural optimization. 13th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Fort Worth, TX, September 2010. AIAA 2010-9231.

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

[59] T. G. Kolda, R. M. Lewis, and V. Torczon. A generating set direct search augmented lagrangian algorithm for optimization with a combination of general and linear constraints. Technical Report SAND2006-5315, Sandia, August 2006.

[60] I. Kroo and V. Manning. Collaborative optimization: Status and directions. In *8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Long Beach, CA, September 6-8 2000. AIAA 2000-4721.

[61] J. Laurenceau, M. Meaux, and P. Montagnac, M. Sagaut. Comparison of gradient-based and gradient-enhanced response-surface-based optimizers. *AIAA Journal*, 48(5):981–994, May 2010.

[62] S. Leary, A. Bhaskar, and A. Keane. A knowledge-based approach to response surface modelling in multifidelity optimization. *Journal of Global Optimization*, 26:297–319, 2003.

[63] S. Leary, A. Bhaskar, and A. Keane. Global approximation and optimization using adjoint computational fluid dynamics codes. *AIAA Journal*, 42(3):631–641, March 2004.

[64] R. M. Lewis and V. Torczon. A direct search approach to nonlinear programming problems using an augmented lagrangian method with explicit treatment of linear constraints. Technical Report WM-CS-2010-01, College of William and Mary Department of Computer Science, January 2010.

[65] W. Li. *Monotonicity and Sensitivity Analysis in Multilevel Decomposition-based Design Optimization*. PhD thesis, University of Maryland, 1990.

[66] H. W. Liepmann and A. Roshko. *Elements of Gasdynamics*. Dover Publications, Inc., Mineola, NY, 1993.

[67] W. Liu. *Development of Gradient-Enhanced Kriging Approximations for Multidisciplinary Design Optimization*. PhD thesis, University of Notre Dame, July 2003.

[68] G. Liuzzi and S. Lucidi. A derivative-free algorithm for inequality constrained nonlinear programming via smoothing of an l_∞ penalty function. *SIAM Journal of Optimization*, 20(1):1–29, 2009.

[69] S. Lophaven, H. Nielsen, and J. Sondergaard. Aspects of the Matlab toolbox DACE. Technical Report IMM-REP-2002-13, Technical University of Denmark, August 2002.

[70] M. Marazzi and J. Nocedal. Wedge trust region methods for derivative free optimization. *Mathematical Programming*, 91(2):289–305, 2002.

[71] A. March and K. Willcox. A provably convergent multifidelity optimization algorithm not requiring high-fidelity gradients. 6th AIAA Multidisciplinary Design Optimization Specialist Conference, Orlando, FL, 2010. AIAA 2010-2912.

[72] J. R. R. A. Martins, J. J. Alonso, and J. J. Reuther. High-fidelity aerostructural design optimization of a supersonic business jet. *Journal of Aircraft*, 41(3):523–530, 2004.

[73] J. R. R. A. Martins, J. J. Alonso, and J. J. Reuther. A coupled-adjoint sensitivity analysis method for high-fidelity aero-structural design. *Optimization and Engineering*, 6(1):33–62, March 2005.

[74] W. H. Mason. Friction. Technical report, Virginia Tech Aerodynamics and Design Software collection, February 2011. Available online at `http://www.dept.aoe.vt.edu/~mason/Mason_f/FRICTman.pdf`.

[75] B. Matérn. *Spatial Variation*. Lecture Notes in Statistics. Springer-Verlag, Berlin, 1986.

[76] MathWorks, Inc. Constrained nonlinear optimization. Optimization Toolbox User's Guide, V. 5, 2010.

[77] P. S. Maybeck. *Stochastic Models, Estimation, and Control*, volume 1. Academic Press, New York, 1979.

[78] N. Michelena, H. M. Kim, and P. Papalambros. A system partitioning and optimization approach to target cascading. In *12th International Conference on Engineering Design*, Munich, Germany, August 1999.

[79] N. Michelena, H. Park, and P. Y. Papalambros. Convergence properties of analytical target cascading. *AIAA Journal*, 42(5):897–905, May 2003.

[80] J. J. Moré and S. M. Wild. Estimating computational noise. *SIAM Journal of Scientific Computing*, 33(3):1292–1314, 2011.

[81] J. A. Nelder and R. A. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[82] M. Nemec and M. J. Aftosmis. Aerodynamic shape optimization using a Cartesian adjoint method and cad geometry. In *25th AIAA Applied Aerodynamics Conference*, San Francisco, June 2006. AIAA 2006-3456.

[83] J. Nocedal and S.J. Wright. *Numerical Optimization, 2nd ed.* Springer, 2006.

[84] R. Oeuvray. *Trust-Region Methods Based on Radial Basis Functions with Application to Biomedical Imaging*. PhD thesis, Ecole Polytechnique Federale de Lausanne, 2005.

[85] P. Y. Papalambros and D. J. Wilde. *Principles of Optimal Design 2nd ed.* Cambridge University Press, 2000.

[86] R. E. Perez, P. W. Jansen, and Martins J. R. R. A. pyOpt: A Python-based object-oriented framework for nonlinear constrained optimization. *Structures and Multidisciplinary Optimization*, 45(1):101–118, 2012.

[87] N. M. K. Poon and J. R. R. A Martins. An adaptive approach to constraint aggregation using adjoint sensitivity analysis. *Structural and Multidisciplinary Optimization*, 34:61–73, 2007.

[88] M. J. D. Powell. A direct search optimization method that models the objective and constraint functions by linear interpolation. *Advances in Optimization and Numerical Analysis, eds. S. Gomez and J.-P. Hennart (Kluwer Academic: Dordrecht, 1994)*, 7:51–67, 1994.

[89] M. J. D. Powell. Direct search algorithms for optimization calculations. *Acta Numerica*, 7:287–336, 1998.

[90] M. J. D. Powell. The NEWUOA software for unconstrained optimization without derivatives. In *Large Scale Nonlinear Optimization, P. Pardalos, G. Pillo, and M. Roma (eds.)*, volume 83 of *Nonconvex Optimization and Its Applications*, pages 255–297, 2006.

[91] D. Rajnarayan, A. Haas, and I. Kroo. A multifidelity gradient-free optimization method and application to aerodynamic design. In *12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Victoria, British Columbia, September 10-12 2008. AIAA 2008-6020.

[92] D. G. Rajnarayan. *Trading Risk and Performance for Engineering Design Optimization Using Multifidelity Analyses*. PhD thesis, Stanford University, June 2009.

[93] C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[94] R.G. Regis and C.A. Shoemaker. Constrained global optimization of expensive black box functions using radial basis functions. *Journal of Global Optimization*, 31:153–171, 2005.

[95] T. D. Robinson, K. E. Willcox, M. S. Eldred, and R. Haimes. Multifidelity optimization for variable-complexity design. 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Portsmouth, VA, 2006. AIAA 2006-7114.

[96] T.D. Robinson, M.S. Eldred, K.E. Willcox, and R. Haimes. Strategies for multifidelity optimization with variable dimensional hierarchical models. 47th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Newport, RI, 2006. AIAA 2006-1819.

[97] T.D. Robinson, M.S. Eldred, K.E. Willcox, and R. Haimes. Surrogate-based optimization using multifidelity models with variable parameterization and corrected space mapping. *AIAA Journal*, 46(11):2814–2822, November 2008.

[98] V. L. Rvachev. On the analytical description of some geometric objects. Technical Report 4, Reports of Ukrainian Academy of Sciences, 1963. (in Russian).

[99] G. R. Saaris, E. N. Tinoco, J. L. Lee, and P. E. Rubbert. A502I user's manual–pan air technology program for solving problems of potential flow about arbitrary configurations. Technical Report D6-54703, Boeing, February 1992.

[100] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4(4):409–435, 1989.

[101] M. J. Sasena, P. Papalambros, and P. Goovaerts. Exploration of metamodeling sampling criteria for constrained global optimization. *Engineering Optimization*, 34(3):263–278, 2002.

[102] R. B. Schnabel. A view of the limitations, opportunities, and challenges in parallel nonlinear optimization. *Parallel Computing*, 21:875–905, 1995.

[103] S. R Searle. *Matrix Algebra Useful for Statistics*. John Wiley and Sons, 1982.

[104] R. S. Sellar, S. M. Batill, and J. E. Renaud. Response surface based, concurrent subspace optimization for multidisciplinary design. In *34th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, NV, January 1996. AIAA 1996-0714.

[105] T.W. Simpson, J.D. Peplinski, P.N. Koch, and J.K. Allen. Metamodels for computer-based engineering design: Survey and recommendations. *Engineering with Computers*, 17:129–150, 2001.

[106] J. Sobieszczanski-Sobieski. Optimization by decomposition: A step from hierarchic to non-hierarchic systems. Technical Report NASA-TM-101494, NASA, September 1988.

[107] J. Sobieszczanski-Sobieski, J. Agte, and R. Sandusky. Bi-level integrated system synthesis (BLISS). In *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, St. Louis, MO, September 2-4 1998. AIAA 1998-4916.

[108] J. Sobieszczanski-Sobieski, J. Agte, and R. Sandusky. Bilevel integrated system synthesis. *AIAA Journal*, 38(1):164–172, January 2000.

[109] J. Sobieszczanski-Sobieski, T.D. Altus, M. Phillips, and R. Sandusky. Bilevel integrated system synthesis for concurrent and distributed processing. *AIAA Journal*, 41(10):1996–2003, October 2003.

[110] NEi Software. *NX Nastran, Numerical Methods User's Guide*. Siemens Product Lifecycle Management Software, Inc., 2008.

[111] N. P. Tedford and J. R. R. A Martins. Benchmarking multidisciplinary design optimization algorithms. *Optimization and Engineering*, 11:159–183, 2010.

[112] A. Törn and A. Žilinskas. *Global Optimization*, volume 350 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1989.

[113] S. Tosserams, L. F. P. Etman, P. Y. Papalambros, and J. E. Rooda. An augmented lagrangian relaxation for analytical target cascading using the alternating direction method of multipliers. *Structural and Multidisciplinary Optimization*, 31:176–189, 2006.

[114] G. Venter, R. Haftka, and J.H. Starnes. Construction of response surface approximations for design optimization. *AIAA Journal*, 36(12):2242–2249, 1998.

[115] Q. Wang, K. Duraisamy, J. J. Alonso, and G. Iaccarino. Risk assessment of scramjet unstart using adjoint-based sampling methods. In *51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Orlando, FL, April 2010. AIAA 2010-2921.

[116] S.M. Wild. *Derivative-Free Optimization Algorithms for Computationally Expensive Functions*. PhD thesis, Cornell University, January 2009.

[117] S.M. Wild, R.G. Regis, and C.A. Shoemaker. ORBIT: Optimization by radial basis function interpolation in trust-regions. *SIAM Journal of Scientific Computing*, 30(6):3197–3219, 2008.

[118] S.M. Wild and C. A. Shoemaker. Global convergence of radial basis function trust region derivative-free algorithms. *SIAM Journal of Optimization*, 21(3):761–781, 2011.

[119] B. Wujek, J. Renaud, and S. Batill. A concurrent engineering approach for multidisciplinary design in a distributed computing environment. In *Multidisciplinary Design Optimization: State of the Art, N.M. Alexandrov and M.Y. Hussaini, editors*, pages 79–89, Philadelphia, PA, January 1997. Society for Industrial and Applied Mathematics.