AN EFFICIENT APPROACH FOR HIGH-FIDELITY MODELING INCORPORATING CONTOUR-BASED SAMPLING AND UNCERTAINTY

A Dissertation
Presented to
The Academic Faculty

by

Daniel R. Crowley

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Aerospace Engineering

Georgia Institute of Technology
December 2013

AN EFFICIENT APPROACH FOR HIGH-FIDELITY MODELING INCORPORATING CONTOUR-BASED SAMPLING AND UNCERTAINTY

Approved by:

Professor Dimitri Mavris, Advisor
Department of Aerospace Engineering
Georgia Institute of Technology

Professor Brian German
Department of Aerospace Engineering
Georgia Institute of Technology

Dr. Jeffrey Zweber
Aerospace Systems Directorate
Air Force Research Laboratory

Professor Daniel Schrage
Department of Aerospace Engineering
Georgia Institute of Technology

Mr. Barry Hellman
Aerospace Systems Directorate
Air Force Research Laboratory

Date Approved: 23 October 2013

For my parents,

without whose support I would

not be the person I am today.

ACKNOWLEDGEMENTS

I would first and foremost like to thank my parents and siblings, who have been very supportive of me despite indications that I would be a lifelong student. My parents always encouraged me to pursue my interests, even when the road was difficult or uncertain, and that encouragement kept me going when I ran into difficulties. I am very grateful that they are willing to forgive my occasional dumb decisions. I am *also* very grateful for my extended family and the stories they've told, which have helped me to put my dumb decisions in the right context.

While earning my doctorate, I enjoyed the support of my friends and family; without them I probably would have given up long ago. They provided receptive ears and kind words when I needed them most. My girlfriend, Julia Barry, is an amazing woman and a major reason why I was able to finish my research. She listened patiently to one-sided "conversations" about iteration noise and flow separation, and never complained when I worked instead of being social. Given how little attention I paid to nutrition while working on this dissertation, her cooking probably deserves credit for helping me avoid scurvy.

Since I came to Atlanta for graduate school, many people have made this city a home for me: Brian Kestner, Gina Martell, and Shuo-Ju & Rebecca Chou have been amazing friends and I am grateful that they made me a part of their lives (and weddings). Without Lynn Hartley I would have missed out on countless great movie-watching experiences. SPAM Engler and Diana Talley each gave me invaluable advice on matters both professional and personal, and if not for them I would have had a much more difficult experience in graduate school. There are too many others to name them all, but I want to thank the members of the Aerospace Systems Design Laboratory, especially James Arruda, Bjorn Cole, Dane Freeman, Curtis Iwata, Carl Johnson, Jonathan Murphy, and Elizabeth Tang.

I'd also like to express my appreciation for the friends who made sure I had a life outside of work: Emily Colvin, Franklin Espino, Chris Black & Mere Hall, Ryan Hall, Paul Kelley,

Kristin Kelly, Jeff Milheizer, Travis Stout, Scott & Ginger Turner, Judd Weinberg, Beth White, and everyone else who made Tuesday Bar and Happy Hour so enjoyable.

During my time at Georgia Tech I have had some wonderful and unique opportunities. I'd like to thank the members of MDRS crews 69, 79 and 93 for all the fun times, despite the close quarters and limited showers. Even when things went awry, it usually made for a good story to tell. I'm also very appreciative of my time at the Craft Center, which gave me the chance to work with some awesome people, goof off with pottery wheels, and get my mind off my research for a bit.

Certain friends have been with me for many years, and I can't adequately express how grateful I am to have them in my life. Sean Griffin and Jeff & Erin Snow cheered me on in good times and pulled me out of bad ones. Through cross-country visits, football games, and New Years celebrations, they've meant the world to me and my life would've been poorer without them in it. I'm also very glad to have shared so many conversations, whether deep or silly, with Kat Farrell over the occasional bottle of wine. Lastly, I'd like to thank Chris Baggetta, Mandie Miller, Tara Deuso, and Erica Deuso for so many years of friendship - it's been great so far, and I'm looking forward to what the future will bring.

I would like to thank the members of my committee - Barry Hellman and Jeffrey Zweber at the Air Force Research Laboratory and Professors Brian German and Daniel Schrage at Georgia Tech - for their feedback and criticism. I struggled with selecting and scoping my research topic, and without my committee that struggle would have gone on much longer. They pointed out parts of my research plan that were overly-complicated or indirect, and they made suggestions that made my point more clearly while simplifying the work ahead of me. I would especially like to thank Jeff and Barry for sponsoring this research.

Lastly, but certainly not least, I want to thank Dr. Dimitri Mavris, my advisor. He invited me to join ASDL, gave me advice on my work and helped me to plan my future after graduation. His support was unwavering, despite what might politely be called "schedule slip" on my part, and without that support I would not have been able to pursue a Ph.D. I am exceedingly grateful for the opportunities he has given me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

During the design process for an aerospace vehicle, decision-makers must have an accurate understanding of how each choice will affect the vehicle and its performance. This understanding is based on experiments and, increasingly often, computer models. In general, as a computer model captures a greater number of phenomena, its results become more accurate for a broader range of problems. This improved accuracy typically comes at the cost of significantly increased computational expense per analysis.

Although rapid analysis tools have been developed that are sufficient for many design efforts, those tools may not be accurate enough for revolutionary concepts subject to grueling flight conditions such as transonic or supersonic flight and extreme angles of attack. At such conditions, the simplifying assumptions of the rapid tools no longer hold. Accurate analysis of such concepts would require models that do not make those simplifying assumptions, with the corresponding increases in computational effort per analysis. As computational costs rise, exploration of the design space can become exceedingly expensive. If this expense cannot be reduced, decision-makers would be forced to choose between a thorough exploration of the design space using inaccurate models, or the analysis of a sparse set of options using accurate models. This problem is exacerbated as the number of free parameters increases, limiting the number of trades that can be investigated in a given time. In the face of limited resources, it can become critically important that only the most useful experiments be performed, which raises multiple questions: how can the most useful experiments be identified, and how can experimental results be used in the most effective manner?

This research effort focuses on identifying and applying techniques which could address these questions. The demonstration problem for this effort was the modeling of a reusable booster vehicle, which would be subject to a wide range of flight conditions while returning to its launch site after staging. Contour-based sampling, an adaptive sampling technique, seeks

cases that will improve the prediction accuracy of surrogate models for particular ranges of the responses of interest. In the case of the reusable booster, contour-based sampling was used to emphasize configurations with small pitching moments; the broad design space included many configurations which produced uncontrollable aerodynamic moments for at least one flight condition. By emphasizing designs that were likely to trim over the entire trajectory, contour-based sampling improves the predictive accuracy of surrogate models for such designs while minimizing the number of analyses required.

The simplified models mentioned above, although less accurate for extreme flight conditions, can still be useful for analyzing performance at more common flight conditions. The simplified models may also offer insight into trends in the response behavior. Data from these simplified models can be combined with more accurate results to produce useful surrogate models with better accuracy than the simplified models but at less cost than if only expensive analyses were used. Of the data fusion techniques evaluated, Ghoreyshi cokriging was found to be the most effective for the problem at hand.

Lastly, uncertainty present in the data was found to negatively affect predictive accuracy of surrogate models. Most surrogate modeling techniques neglect uncertainty in the data and treat all cases as deterministic. This is plausible, especially for data produced by computer analyses which are assumed to be perfectly repeatable and thus truly deterministic. However, a number of sources of uncertainty, such as solver iteration or surrogate model prediction accuracy, can introduce noise to the data. If these sources of uncertainty could be captured and incorporated when surrogate models are trained, the resulting surrogate models would be less susceptible to that noise and correspondingly have better predictive accuracy. This was accomplished in the present effort by capturing the uncertainty information via nuggets added to the Kriging model.

By combining these techniques, surrogate models could be created which exhibited better predictive accuracy while selecting the most informative experiments possible. This significantly reduced the computational effort expended compared to a more standard approach using space-filling samples and data from a single source. The relative contributions of each technique were identified, and observations were made pertaining to the most effective way

to apply the separate and combined methods.

CHAPTER I

INTRODUCTION

As an aerospace vehicle design project progresses, a variety of decisions must be made which range from the very large-scale - should the vehicle be fixed-wing, rotary-wing, or lighter-than-air? - to the very small-scale - where should each rivet be placed? In order to make these decisions, designers rely on information from a variety of sources such as experience, intuition, direct experimentation, and simulation. All this information serves one ultimate purpose: aiding the decision-maker by revealing the consequences of his or her decisions. Effective decision-making requires accurate knowledge of how each decision will affect the overall vehicle and its performance. Without accurate knowledge of these consequences, a decision-maker risks making bad choices which can lead to poor vehicle performance, program delays, or complete project failure. Care should be taken to ensure that the consequences of a decision are accurately understood before a choice is made.

1.1 Phases of the Design Process

As the design process moves forward, the vehicle in question is progressively defined and refined. In aerospace applications, the stages of the design process are typically referred to as Conceptual, Preliminary, and Detailed Design.[162]

During conceptual design, the widest possible design space is explored. Extremely rapid analysis techniques are used to evaluate as many options as possible. An **analysis** is any approach that quantifies performance given a set of input values, and may be as simple as an equation or as complex as a full-scale flight test. In conceptual design, the analyses will mostly be simplified models that can quickly estimate the performance of possible designs. The objective of this phase is to identify one or more promising vehicle options. The degree of concept definition at the end of this phase may vary from program to program: Li et al.[104] fixed the planform of the wing (i.e. span, taper, sweep) by the end of conceptual design, leaving only the airfoil undecided, while Hutchins et al.[84] carried all of these parameters

into preliminary design to perform further trade studies. During this phase, the geometry of the vehicle is defined using a few parameters, typically less than ten or fifteen.[104] In general, however, conceptual design focuses on broad definitions such as initial vehicle sizing and configuration selection.

During the next stage, preliminary design, the concept or concepts selected during conceptual design are further refined using more accurate analysis methods. If analyses capturing multi-disciplinary effects such as aeroelasticity were not included during the conceptual phase, they must be performed now. This phase emphasizes the use of higher-fidelity models for more accurate estimation of concept performance, with an associated increase in computational cost per analysis. The geometry of the vehicle is defined in greater detail, increasing the number of shape parameters to a few dozen or as many as a few hundred.[104] Most or all of the vehicle's geometric shape is frozen by the end of this phase.[89]

Finally, detailed design emphasizes individual components of the vehicle such as ribs or fuel tanks. Each component is sized to meet its expected requirements and defined at the level of detail required for manufacturing. The final, most highly refined estimates of weight and vehicle performance are made, and prototypes are constructed. This is the longest phase of a design project.

Each phase of the design process is marked by an increase in the available information about the design. Early on, very rough analyses may be performed with only a few details about the concept, such as take-off weight, maximum thrust, and wing area. Conversely, by the end of detailed design there may be thousands of components, each defined via tens or hundreds of parameters, which must be included in order to assess the performance of the vehicle as a whole.

The use of rapid, simplified models in the early phases of the design process introduces some element of risk. If the models do not capture all the phenomena that significantly affect vehicle performance, there is a chance that later on, when more accurate analyses are conducted (such as wind tunnel tests) the selected vehicle will be found to have poor performance. If poor performance is discovered, the design must be modified until performance improves, a process which may cause overruns of schedule or budgetary goals. It is in the

designer's interest to address the risk of deficient performance when making decisions.

This research effort focuses on conceptual & preliminary design, when decisions are made to fix parameters at particular values and a baseline configuration is selected. The potential for mistakes introduced by low-fidelity modeling may be addressed in a number of ways:

- Trade studies to quantify how decisions might affect predicted performance;
- Repeating experiments at one level of fidelity to assess the uncertainty of predictions at that level of fidelity; and,
- Repeating experiments at different levels of fidelity to assess the degree to which the change in fidelity level affects the performance predictions

"**Level of fidelity**" can have different meanings depending on the context, although in general it can be taken to mean the degree to which the analysis matches reality.[8] When describing physical experiments, an experiment that matches flight Mach number is of lower fidelity than one that matches both Mach number and Reynolds number. Computational models have a wider range of fidelity; some models assume that viscous effects are negligible, or that only linear effects are significant.

1.2 Trade Studies & Optimizations

At each stage of the design process, the available design space is explored to determine how decisions might affect vehicle performance. The **design space** is defined by the parameters being investigated and the allowable ranges of those parameters. Exploring the design space may take the form of trade studies or optimizations. Put briefly, optimization progressively (and in some cases, automatically) modifies a design to maximize or minimize some user-selected response function; in contrast, trade studies are used to generate data about various potential designs which will be used to support decision-making.[41]

In both processes, multiple designs are analyzed to evaluate performance. The results are used to learn how the design parameters affect the response(s) of interest. In an **optimization**, the results will be converted into a score using an explicit quantitative objective function, defined at the beginning of the process. One or more new candidates will then be

generated based on the observations and analyzed to determine objective function scores. This objective may be a direct output of the analyses, such as vehicle weight or operating costs, or it may be a combination of multiple responses, each given a particular weight to reflect the importance of improvement in that response. The optimization process will be repeated until an optimum is identified or the allotted resources have been expended. In essence, an optimization may be considered to be a repeated trade study with a known, quantifiable objective.

In a **trade study**, this information may be used to support a decision directly, or used as the foundation for another trade study to investigate any interesting behavior observed. The System Engineering Manual for the Federal Aviation Administration's National Aerospace System states that a trade study is used, "to identify the most balanced technical solutions among a set of proposed viable solutions."[53] The Manual also states that use of trade studies "prevents program/project management from committing too early to a design that may not be cost effective or meets [*sic*] all system requirements." These trade studies are performed at all stages of the design process,[41, 42, 46] and are used to investigate the design space. By investigating the results, the effects of various design parameters on response behavior may be inferred. This allows the decision-maker to identify which parameters significant affect the responses of interest. If a parameter does not significantly affect any responses of interest, it may be set to some reasonable default value and omitted from future trade studies, simplifying the effort.

Decision makers are faced with conflicting motivations when setting up a trade study. There is incentive to include as many parameters as possible in a trade study so that every effect can be investigated, including interactions between design variables. Hutchins et al.[84] cite "the often strong coupling between the variables and the highly multidisciplinary nature of the design" as motivation to include as many parameters as possible in an aero-structural tool for wing trade studies. Furthermore, as the project moves forward some design parameters are frozen;[104] the decision-maker must take care not to freeze a parameter before its effects are known with confidence, or else risk costly backtracking if the frozen value is later found to be detrimental.[84, 166] These factors provide the incentive to

perform large trade studies with many parameters.

On the other hand, a trade study seldom can include *every* variable of interest. These studies must take place within an ongoing design effort, and thus will not have unlimited resources. As the number of design variables in a study increases, the simulation effort required to complete that study increases very rapidly,[72, 99] an effect known as the "curse of dimensionality." As a result, trade studies often must be carefully designed to ensure they can be completed using the experimental resources available, and without taking an excessive amount of time.[89] Careful selection of experiments can ease such constraints to some degree, but they cannot be ignored altogether. Thus, these constraints may limit the analysis tools that may be used in a given trade study.

1.3 Speed & Fidelity of Analyses

> "It can be scarcely be denied that the supreme goal of all theory is to make the irreducible basic elements as simple and as few as possible without having to surrender the adequate representation of a single datum of experience." -Albert Einstein[125]

Every computational analysis tool makes certain simplifying assumptions in its representation of the world, even if only through spatial and/or temporal discretization. Jameson[89] estimated that modeling the airflow around a typical aircraft with discretization fine enough to capture boundary layer behavior over all active length scales would require on the order of ten billion mesh points within the boundary layer alone. Capturing the evolution of that flow field through time would further require roughly fifteen thousand time steps per second. This degree of resolution would put modeling well out of reach at present. Jameson goes on to note, however, that the amount of information produced by such an analysis is far in excess of what is required for typical engineering efforts.

Most engineering analysis focuses on large-scale responses, such as the total drag on a vehicle. If precision is not critical, these responses can be estimated using lower-fidelity methods which make simplifying assumptions. These assumptions usually posit that certain factors have insignificant effects relative to the precision required, and thus may be

neglected. Although these simplified methods are of lower fidelity, they can be very useful if the dominant effects are still captured. For example, the Newtonian theory of gravity produces a very good approximation of orbital motion. Based on observations of the seven known planets, the position and existence of the planet Neptune was predicted in 1846 using Newtonian theory. However, despite astronomers' best efforts, the theory could not account for the observed precession of Mercury's orbit, and for years astronomers searched for a new planet to explain the inconsistencies.[191] When Einstein published his theory of general relativity,[101] its ability to accurately model Mercury's orbit was a major achievement and a strong argument in favor of the new theory. Still, the Newtonian theory of gravity was used to correctly predict the existence and position of Neptune, demonstrating that a model may be useful and informative even when it is not a perfect reproduction of reality.

There exist a variety of levels of modeling fidelity for aerodynamics.[169] Among the most accurate are computational fluid dynamics (CFD) simulations that capture, to varying degrees, viscous effects. Different approaches make different simplifying assumptions to reduce the large computational demands imposed by viscous flow interactions. These viscous models range in complexity from Large Eddy Simulation, which constrains the lower limit of the length scale of the viscous behaviors captured, to Reynolds-Averaged Navier-Stokes (RANS) simulations which only attempt to model time-averaged behavior of the turbulent motion. Euler flow models, the next level of simplification, neglect viscous effects but retain other non-linear effects. Making the additional assumption that rotational flow effects are insignificant yields the potential flow equations, which may be linearized for an even simpler model. Some models intended to analyze fairly slow flight speeds may also neglect the effects of compressible flow.

The modeling approaches in the previous paragraph were developed from analytical descriptions of fluid behavior using a series of assumptions as to which fluid behaviors would significantly affect the desired outputs, such as the resulting lift of the vehicle. The loss of accuracy resulting from each simplification depends on the response being modeled and the problem being analyzed – assuming incompressible flow at hypersonic conditions will introduce more error than the same assumption at Mach 0.1. Incompressible potential

flow models may give a good approximation of lift on a vehicle at slow speeds, but the vehicle drag will be vastly underestimated.

An alternative approach to estimating aerodynamic responses is to draw upon known flight performance of existing aircraft. By its nature, this data captures all relevant fluid behaviors because it is compiled from actual flight test data rather than simulations. With data from enough aircraft, trends and patterns can be identified and correlated with design parameters. Tail volume coefficients are an example of the application of such a trend.[162] Using these trends and some aerodynamic theory, the performance of a new aircraft design can be estimated.[47] This type of model is known as a semi-empirical or handbook method, and it can be very powerful when applied to a concept similar to those used as references for the model. However, the relations in the model are only indirectly based on flow physics, and if the model is applied to a configuration that is *not* similar to those in the underlying data set its predictions may not be accurate.

Most analysis tools will adequately capture simple responses such as normal force, but a complex result like the center of pressure on a body will highlight any inaccuracies of lower-fidelity tools.[129] The center of pressure must be predicted accurately if aerodynamic moments are important. Handbook methods often predict centers of pressure that are not as accurate as those from Euler simulations, which in turn are less precise than viscous predictions. Unfortunately, the superior prediction accuracy of viscous calculations comes at a greatly increased computational cost.

In general, the simpler the analysis, the faster it will execute. OVERFLOW[137] is a RANS model that, depending on the complexity of the analysis and the amount of resources allocated, can complete a viscous analysis of a configuration at one flight condition in perhaps a few hours on a parallel computing cluster. Cart3D,[6] an Euler-level flow solver, can reach an inviscid solution in roughly half an hour on a single computing node of eight cores. The Unified Distributed Panel (UDP) program, the subsonic/low-supersonic portion of the Aerodynamic Preliminary Analysis System (APAS),[181] is a potential flow solver and can estimate the aerodynamic performance of an aircraft at one flight condition in a fraction of a second on a desktop computer.

Furthermore, tools that capture more complex phenomena may require more information before an analysis can be performed. Defining a configuration for APAS typically requires a few dozen cross-sections; defining a configuration for Cart3D requires a full water-tight surface mesh describing the entire vehicle, including joints between components. This requires a more detailed description of the vehicle which may not be available early on in the design process, particularly if the configuration is not defined in a parametric environment and changes have to be made manually. As an added complication, vehicle performance may be affected by geometric details as small as the blending between two wing sections,[65] or the doors covering payload or landing gear bays.[183] Lee demonstrated that small-scale nacelle details can cause shock waves strong enough to affect the entire upper surface of the wing, drowning out many other effects of interest.[103] Using a high-fidelity tool and making unfortunate choices for default settings may make it difficult or impossible to carry out a trade study effectively.

Oberkampf & Roy[138] agree that it may be inappropriate to use the highest possible level of physics modeling for every computational study. A highly detailed simulation of the flow field around an aircraft can be extremely computationally intensive, and the resulting level of solution detail may vastly exceed what is necessary. Such in-depth modeling can produce results with very good confidence but at such high cost that the approach becomes infeasible. Trade studies are usually conducted as part of a larger design effort and therefore must be completed within a budget of allocated resources. These resources may take the form of time, manpower, computational effort, financial budget, or other factors. If the study takes too long to complete it may be ended prematurely, or the rest of the design process may be delayed. The program managers will have to choose between moving forward with insufficient information to support their decisions, or with fewer resources available for later work. To avoid these negative outcomes, trade studies should be carefully planned to balance the worth of the information gained against the cost of providing that information.

One key technique for finding this balance is to pay attention to prediction confidence. The predictions of high-fidelity, high-complexity models are accompanied by tight confidence intervals. Those tight confidence intervals indicate that the true value of the quantity being

simulated, such as lift or drag, is likely to be very close to the predicted value. However, such confidence intervals usually come at the expense of time, manpower, and computational effort. As models are simplified, prediction confidence may be reduced and the confidence intervals grow larger. When a designer defines the prediction confidence required for a study, he or she thus constrains the minimum level of model complexity that can still meet the objectives of the study.

These confidence interval requirements will vary as the design process moves forward. Early studies have relatively broad confidence intervals because they describe the aircraft using only a handful of major design parameters such as wing reference area. Many other aspects of the vehicle, such as the precise shape of the fuselage, are undefined or roughly approximated at that stage. Using the results of such a study, desirable values for this first set of parameters may be identified. Later studies will refine the vehicle concept by incorporating more parameters, such as airfoil selection and control surface sizes. This cycle is repeated until the entire vehicle is defined to a degree that manufacturing may begin. Each trade study must be carefully planned according to its objectives and resources.

1.4 Planning a Trade Study

As described in Section 1.2, the simulation effort for a study increases rapidly as more parameters are included in the study.[72] This provides the motivation to incorporate only the parameters which are likely to significantly affect the results. To quantitatively identify important parameters out of a pool of candidates, screening tests may be performed.[193] Screening tests can be used to identify the parameters that most affect the behavior of the response, and the magnitude of their effect. They may also identify important second-order interactions between parameters, although this typically requires more testing than a screening test for first-order effects only. These screening tests are performed by running the computational model for particular sets of input values and investigating the results with statistical techniques. A screening test requires some investment of effort, but may identify parameters that may safely be omitted from trade studies, reducing the overall effort required without sacrificing the quantity & quality of information produced.

In addition to screening tests, the set of active parameters in a study may be reduced by eliminating variables that have previously been investigated. For example, after a conceptual design study has identified a particular value of wing sweep angle as being most beneficial to vehicle performance, that value may be used as the default for later studies. A design with multiple parameters that have been fixed may be considered the baseline configuration for future studies. This simplifies those future studies by reducing the number of free parameters. When Boeing was designing the 777, the external shape of the vehicle was fixed, or "frozen," during preliminary design. Subsequent studies only investigated trades that would not affect this outer mold line.[89]

Although defaulting and/or screening out variables will reduce the expense of a given study, it will not eliminate the pressure to execute the study within the time and resources allotted by the project schedule. The trade study designer is still responsible for balancing the information gained against the costs of generating that information.[89] The Pegasus booster, developed in the late 1980s, relied entirely on computer simulations using a variety of analysis tools.[131] Due to the limitations of the processing capabilities available at the time, almost all of the analysis was done using potential flow solvers and impact methods. Euler- and RANS-level models were applied only a handful of times, principally for confirmation or correction of the lower-fidelity tools. In particular, the more-accurate models were used to investigate the possibility of plume-induced flow separation and interactions between shock waves and boundary layers, effects that could not be captured using simpler tools. For the most part, full-scale flight testing confirmed the performance predictions.

This example demonstrates more than one important concept. First, flow behaviors that were, or might be, important were identified by the team in advance. Tests were done using tools of appropriate fidelity to determine whether those flow behaviors would significantly affect the vehicle's performance. Secondly, these high-fidelity tools were applied intelligently, with emphasis on conditions which would be the most difficult for lower-fidelity tools to capture accurately, such as high angle-of-attack flight. This aided the team in estimating an upper bound for error in the lower-fidelity performance predictions. Finally, the modeling tools for the primary effort were selected based on the flow phenomena that

were expected to be significant as well as the resources available for the effort. Because complex phenomena were not found to be significant, lower-fidelity tools could be applied safely without introducing much error. These tools executed more quickly, allowing a larger number of analyses to be performed over the flight conditions of interest. Ordinarily, such a study would use wind tunnel testing to substantiate the predictions of computational models. In the case of the Pegasus booster program, no commercial wind tunnel time was available for nearly a year, a delay that was incompatible with the project schedule.[129] The project team instead applied multiple independent analysis tools across the trajectory space of interest in order to estimate the confidence in each prediction.

When setting up a trade study, designers must account for multiple factors: the number of parameters to be investigated, the amount of resources available, and the level of fidelity of the analysis or analyses that will be used. Using simpler analyses may greatly reduce the resource cost per experiment, but designers must take care to ensure that the analyses will capture all relevant effects and behaviors. Using inadequate tools may expose the project to significant risks.

1.5 The Cost of Inadequate Analysis Fidelity

Unless the significant phenomena are identified, it may be impossible to determine the amount of error or uncertainty introduced by simplifying assumptions. A design effort lacking this information is at risk of selecting a concept with deficient performance; this deficiency will then go unrecognized until higher-fidelity tools are applied later in the project, potentially leading to backtracking and/or repetition of previous studies. Dorsey et al.[46] described a sizing trade study for a reusable launch vehicle that captured propellant tank and intertank structure, including detailed tank shape & arrangement parameters. Prior parametric weight estimation tools did not capture these factors, and "[a]s a result major perturbations have been made to the vehicle structure" which led to the "erroneous result of no apparent impact on the vehicle total weight was obtained.". When the tank & intertank parameters were investigated with more accurate modeling tools, it was observed that structural weight was affected quite strongly by design parameters such as vehicle half body

angle and payload carriage location. For example, moving the oxygen tanks from the rear to the front of a vehicle changed the structural loads, causing an 18% increase in the required structural mass; this effect was not captured using lower-fidelity tools, and decisions which would significantly affect the performance of the vehicle might have been made without accurately understanding the consequences.

Other examples of the hazards of insufficient model fidelity may be found in the literature. Aeroelasticity is the complex interaction between aerodynamic loads on a body, its structural response, and the effect of that response on the body's aerodynamics. Capturing this behavior requires knowledge of the local aerodynamic loads on vehicle components as well as a representation of component structures. The detailed information required, along with the feedback loop inherent in the phenomenon, may preclude this analysis until late in the preliminary design phase when much of the proposed vehicle has been defined; this decision carries with it the unstated assumption that aeroelastic effects will not significantly influence the design. Werner-Westphal et al.[192] demonstrated that for rearward-swept wings, neglecting aeroelastic effects may be a *conservative* assumption - that is, aeroelastic effects reduce wingtip loads and by extension wing structural requirements. However, this effect is reversed for vehicles with forward-swept wings: for such vehicles, aeroelastic effects increase wingtip loads, and accounting for the higher loads may drive up wing structural weight of the example aircraft by roughly eight percent.

The incentive for high-fidelity modeling is not only an issue for multi-disciplinary analyses: Collier et al.[35] investigated the degree to which different structural requirements drove wing box structure weight. When damage initiation and damage tolerance (i.e., crack-growth) constraints were taken into account, a design that previously appeared sufficient was found to have negative margin. Modifying the design to satisfy the violated constraints resulted in an increase of the required structural mass of the wing by 12%. Unless these modifications can be identified and executed relatively early in the design process, the designers may be forced to repeat many analyses to understand the effects of these changes on the performance of the overall vehicle. This consequence carries the choice between a dual penalty of lost time *and* increased modeling effort, or a loss of knowledge about the vehicle.

It bears mentioning that the risk introduced by insufficient modeling fidelity is not solely a problem for computer models. The Lockheed C-141 Starlifter was designed in the early 1960s with experimental support from wind tunnel testing. When full-scale flight testing began, the handling characteristics were significantly worse than predicted. Experiments identified the source of the discrepancy: although some shock-induced flow separation was expected, the flow separation at flight Reynolds number differed from what was observed in the wind tunnel experiments by as much as 20% of the local wing chord.[23] Furthermore it was found that, given the knowledge at the time, *only* simulations at the full flight Reynolds number would have adequately predicted the Starlifter's in-flight behavior.[17]

The Space Shuttle development program provides another example: after the first flight test, it was found that hypersonic trim at high angle of attack required a significantly larger deflection of the body flap than was planned. Although the difference in pitching moment coefficient was only 0.03, correcting the discrepancy required 16° of deflection rather than 7°, leaving less than one-third of the expected control margin for maneuvering or controlling dispersions. Later testing indicated that real gas effects, which were not investigated before the flight, accounted for the majority of the discrepancy.[86, 133]

Any study conducted with inadequate fidelity may lead to a concept which will later be revealed as infeasible or deficient. These shortfalls may significantly impact the design effort. Given that such deficiencies would only be identified by higher-fidelity modeling, they will not be captured until later - possibly much later - in the design process. Once discovered, reactionary changes can begin, but the impact of these changes on the state of the design may be very large if more than a few analyses have been performed using the old design. It may be less costly, then, to ensure that the analysis tools are adequate for their purpose in the study; this may be difficult to achieve without a separate investment of effort.

1.6 Determining Sufficient Fidelity

Researchers have identified a variety of techniques for assessing the ability of computational tools to support a particular analysis. This process is known as *validation*. The AIAA Guide for the Verification and Validation of Computational Fluid Dynamics Simulations[8]

defines **validation** as "the process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model." This definition emphasizes that, while validation does compare model predictions to the "real world," it is important to focus on the *intended uses of the model*. Put another way, a previously-validated tool may need to be re-validated if it is to be applied to a substantially different problem, and the similarity of validation tests to the planned application of the tool is critical.

One approach to tool validation, used in the development of the Pegasus booster,[129] is to repeatedly analyze a vehicle at some important flight conditions, such as cruise and landing, using different computational tools. These tools should be independent; any common code shared between tools will make it more difficult to identify inaccuracies in the results. The selection of tools should represent not just various analysis programs, but different levels of fidelity in order to capture the errors introduced by simplifications. The higher the fidelity of the tools included in the effort, the greater the likelihood that the results reflect reality, although this cannot be guaranteed without recourse to physical experiments.

This method must not be applied recklessly. A group of handbook methods, applied to the same unconventional vehicle, may all produce similar results; this should be taken as an indication that there is not a substantial difference in accuracy between those codes, but *not* an indication that the codes all produce accurate results. Accuracy can only be determined by comparing the handbook estimates against higher-fidelity models. When an empirical method is applied to configurations and flight conditions similar to those used to build the tool, it may be highly accurate. This pedigree is lost, however, when the method is applied to unconventional designs or unusual flight conditions. To better capture the prediction uncertainty, multiple levels of fidelity should be used. The tools should be applied to a problem as similar as possible to the topic of the planned trade study. It does little good to verify a code's accuracy when applied to a subsonic large transport if the goal is an accurate performance estimate of a hypersonic missile.

Validation experts[138, 141] prefer to validate analysis tools against experimental data. Data from wind tunnel experiments (or better yet, flight tests) will naturally capture most

or all relevant phenomena. This reduces the risk of neglecting important effects, creating greater confidence when a computational tool matches experimental results well. In a sense, full-scale physical experiments may be considered the highest-fidelity analysis. This fidelity comes at the expense of data availability.

Readily-available validation data may come from institutional memory, or it may be in the public domain: NATO's Advisory Group for Aerospace Research and Development (AGARD) has published sets of data intended for use in validating computational tools.[11, 188] Unfortunately, this data is necessarily of limited scope. The AGARD reports include only one set of experimental data to validate full-vehicle supersonic analyses - a combat aircraft research model with a forward-swept wing and a canard. Unless this matches the expected application of the code, it may be difficult to accurately assess prediction uncertainties. When the objective is to model a vehicle that is similar to those for which data is available, tool validation is relatively straightforward; for revolutionary designs, it may be difficult or impossible to find pre-existing experimental data for tool validation.

The problem becomes more complicated when experimental uncertainty is included. A wide variety of uncertainties are present in any experimental data, such as the precise wind speed, the exact angle of the model, and the accuracy of the sensors used. Behavior such as hysteresis may also affect the measurements at each condition depending on the prior experiments run.[12] This information should be included with the nominal data set for accurate uncertainty estimation, but most published data sets neglect to include some or all of it. The cited AGARD reports are exemplary in the level of uncertainty documentation, but those reports were explicitly designed to be used for validation; it is rare to find that degree of documentation in ordinary experimental reports. For this reason, even if experiments were performed using configurations and conditions similar to those desired, it may still be difficult to accurately estimate tool prediction accuracy without documentation that is unusually thorough.

The final option for tool validation is the use of *ad hoc* experimental data. Wind tunnel or flight test experiments may be designed and executed with the objective of assessing tool prediction accuracy for one or more scenarios. Although this produces high-quality data

that is directly relevant to the problem(s) of interest, it is also the most expensive tactic. Physical models must be built and instrumented, time in wind tunnels reserved, and data taken. Each data point may need to be sampled repeatedly in order to quantify experimental variation and path-dependent effects. Raymer estimates the cost of wind tunnel experiments at "several hundred thousand dollars per model."[161]

To its credit, this option does allow the user to specify which uncertainties should be quantified, and allows the user to guarantee that the configuration and flight conditions are relevant to the intended application of the tools. If time and resources are available, this is the approach that will best characterize the prediction confidence of the modeling codes for the scenarios of interest. Not every program has the resources to perform such experiments. Mendenhall[129] stated that the Pegasus booster program was executed without validation from physical experiments, not by preference, but due to a constrained program schedule and the unavailability of wind tunnels.

Once validation data and all tool predictions are in hand, comparisons can be made. The breadth of the predictions will give the user some idea of the overall prediction uncertainty. If predictions are scattered widely, the simplifying assumptions of the models significantly affect the calculations, and care should be taken to select the appropriate tool. If all predictions agree closely, the choice of tool is not likely to introduce significant error to the results. This procedure must be repeated at all flight conditions of interest until the dependability of each model at every relevant flight condition has been established. This produces an estimate of prediction uncertainty for each tool when applied to each scenario of interest, i.e. each combination of flight condition and vehicle configuration.

The question of how to define "acceptable" prediction uncertainty is an important one, and one which varies with the application. If the pitching moment is of interest, a designer might require a level of uncertainty smaller than the expected control surface authority. More generally, acceptable uncertainty can refer to a level of prediction confidence sufficient to discriminate accurately between the options being considered. This degree of discrimination will change as the design process evolves, becoming progressively more precise. An

example of such an acceptable uncertainty is a rule of thumb, such as: if the pitching moment coefficient for a vehicle with undeflected control surfaces falls within ± 0.1, it is likely that some set of feasible control surface deflections can be found to negate that pitching moment.[24] This rule of thumb allows designers evaluate multiple configurations without having to tweak control surface deflections to trim the vehicle for each flight condition being analyzed.

Modeling in the early stages of design may be somewhat imprecise without causing much concern, because most design parameters have yet to be given values. As the concept is refined, smaller-scale changes are investigated, and more precision is required to support decisions.

1.7 When Requirements Conflict

Note that the requirement for prediction uncertainty is independent of the computational tools being considered. There is no guarantee that any tool will offer the required confidence at a feasible cost. This is especially true if the objective is an investigation of many parameters at once, which can be necessary to capture interactions between parameters. As Raymer says, "[t]he choice of code for a given design problem depends on the nature of the problem and the available budget (and not always in that order!)."[161] What can be done if tool validation reveals that a desired trade study can not be completed without violating either confidence requirements or resource limits?

The designers are then faced with a conflict. One option is to *exceed the resources allocated to the study* or request additional support. This is seldom an attractive option, and may not be a viable choice if the resource limits (whether financial, computational or temporal) are firm limits.

Another option is to *relax the prediction confidence constraints*. This would allow simpler, faster tools to be used in the trade study, and may reduce the resource cost enough that the study could be completed within the given constraints. On the negative side, this introduces risk: with the relaxed confidence requirement, decision makers might select a configuration that is predicted to safely meet performance requirements, but in fact violates them instead.

This deficiency will not be recognized until a more-accurate analysis is done sometime after the current trade study. This is what occurred during the development of a fly-back booster by DLR, the German Aerospace Center: a lower-fidelity aerodynamics tool was used to optimize the design of a canard. Later, when higher-fidelity modeling was performed, it was found that the canard shed vortices that negatively affected wing performance, resulting in deficient performance.[52]

It may be possible to improve the deficient performance using the remaining unfixed variables, although this suggests that the performance of the vehicle could have been even better if the original deficiency were not present. If it is not possible to ameliorate the design problems, the decision makers will have to roll back the design process to address the source of the trouble. Any decisions that were made after the relevant parameters were frozen may have to be revisited. This can be an expensive process with respect to both effort and time. When DLR identified poor performance in their fly-back booster design, high-fidelity analyses – in the form of Euler simulations and wind tunnel experiments – had to be used to identify the source of the error, which was found to stem from canard tip vortices affecting the airflow over the main wing. These analyses were costly, but simpler models demonstrably did not capture the flow phenomenon of interest. Once the canard tip vortices were identified as the source of the discrepancy, the canard was modified to reduce this effect and the results were confirmed using Euler CFD.[176]

If neither the resource limit nor the confidence requirement can be relaxed, the designer's final option is to *change the scope of the trade study.* This may take the form of reducing the ranges of the variables being investigated, eliminating some variables from the study entirely, or both. Reducing variable range simplifies the problem somewhat by limiting the portions of the design space that will be explored. This can be an effective technique to improve efficiency, but there is a risk that interesting portions of the design space might be ignored. Removing variables from consideration can be even more powerful: Section 1.2 described the Curse of Dimensionality, a pithy shorthand for the observation that the computational cost of a study increases extremely rapidly as the number of parameters grows. Correspondingly, the cost will decrease rapidly as parameters are removed.

In many engineering problems, most outputs are significantly affected by only a handful of inputs, a phenomenon known as "effect sparsity" or the **Pareto principle**.[193] Although a computer model may have dozens of input parameters, it is likely that most only have minor effects on a particular output. To determine which parameters are important to a given response, screening tests may be performed. Screening tests use a relatively sparse sampling of the design space to identify major linear and nonlinear effects, including any interactions between variables. Thus, a screening test incurs some cost, but may result in an overall savings of resources by identifying parameters which do not significantly affect the results.

For some problems, this principle does not hold, especially if many responses are to be investigated simultaneously. Although each response depends mostly on a handful of inputs, there is no guarantee that every response will depend on the *same* handful of inputs. The more responses that must be captured, the worse are the chances that any given parameter has negligible effect on *all* responses. Furthermore, the parameters that significantly affect aerodynamic responses will change with the speed and angle of flight: nose shape has a much greater effect at supersonic conditions than it does for subsonic flight. Thus, as the number of important flight conditions for a vehicle increases, the designer can expect that a greater number of geometric parameters to be significant for vehicle aerodynamics. In such situations, designers will not easily be able to omit variables without omitting portions of the relevant design space.

What can be done in this situation? Constraints on resources encourage rapid evaluations or fewer analyses, while constraints on prediction confidence drive the user toward methods with high confidence but slow evaluation speeds. If the study cannot be captured in a small number of analyses, the designer must investigate techniques that can reduce the cost of performing the trade study. An example of such a scenario is investigated in detail in the next chapter.

CHAPTER II

MOTIVATING EXAMPLE

The conflicting objectives of highly accurate data and affordable analysis costs can be illustrated by a recent research effort. The Air Force Research Laboratory (AFRL) sponsored research into modeling the aerodynamics of a **reusable booster system** (RBS). In particular, the AFRL expressed interest in a hybrid system, such that the first stage was reusable while the upper stage or stages would be expendable. For this concept, the first stage would lift the upper stages to a particular altitude, orientation, and speed, and then the first stage would be jettisoned. The upper portions of the launcher would continue toward orbit while the first stage would reverse its course and returned to the original launch site. The maneuver by which the vehicle reversed its course is referred to as a return to launch site (RTLS) maneuver.[19, 71]

Such a system has been the subject of research by various other entities, including NASA[144] and DLR.[176] Hellman[79] gives an overview of some studies on the topic. The research sponsored by the AFRL focused on vehicles which employ the main rocket engines to offset the downrange velocity of the booster after staging, an operational concept known as "**rocketback**." After the engines cease firing, the vehicle executes a gliding return to the launch site.

Previous studies by Masse[118] and Sippel[176] highlight one of the major design challenges for a winged reusable booster: stability and control. The booster must have sufficient control authority to achieve trim along the entire RTLS trajectory. The gliding portion of the RTLS trajectory, which was the main subject of this effort, can include flight conditions from nearly hypersonic reentry speeds down to low-subsonic landing speeds, and angles of attack that stretch from single digits up to 30-40°. Identifying a vehicle that can trim at such a broad variety of flight conditions is a demanding design task.

Further complicating matters is the complexity of the aerodynamics across the various

flight regimes. Some rapid computer models, such as APAS,[181] neglect nonlinear effects which have been shown to be important for reusable boosters by Pamadi et al.[144] In addition, APAS is known to be somewhat inaccurate when predicting pitching moment coefficients for configurations with long fuselages or aft center-of-gravity locations,[143] both of which may be the case for a winged reusable booster vehicle. To increase confidence in the analysis results, Cart3D,[6] an Euler CFD model, was selected as a good balance of fidelity and speed. The results of Pamadi et al. demonstrated that Cart3D could match the available wind tunnel data for the Langley Glide-Back Booster fairly well. Eggers demonstrated that TAU, another Euler solver, showed good agreement with the wind tunnel results for another reusable booster configuration after sting and Reynolds number effects were subtracted.[50]

These results indicate that Cart3D would have sufficient fidelity to capture the dominant effects and flow behaviors. Note that, as an inviscid model, Cart3D would not capture viscous effects like skin friction drag or boundary layers. This was deemed an acceptable loss, as the viscous modeling required to capture those effects would increase computational costs by roughly another order of magnitude. Using 8 processors, Cart3D can typically analyze a vehicle's performance at a given flight condition in 30-60 minutes. Details about the way that Cart3D was applied for this work, including default flow solver settings, are given in detail in Appendix B.

The objective of the RBS research project was to create surrogate models which captured the aerodynamics of rocketback vehicle designs. Those surrogate models could then rapidly evaluate the performance of any configuration within the design space. The surrogate models would be functions of geometric parameters, such as the fuselage radius or wing root chord. A parametric vehicle geometry model was created in the PaceLab Engineering Suite[142, 175] which included 42 geometric variables. Certain design characteristics were considered to be fixed: the forward fuselage was cylindrical, transitioning smoothly to a flat bottom in the rear. The wing would be mounted at or near the rear of the vehicle, low on the fuselage. Rather than a dorsal vertical tail, vertical fins would be located on the wing tips, similar to the X-20 "Dyna-Soar".[90] For a more detailed explanation of the geometry model and the meaning of each parameter, see Garmendia et al.[61]

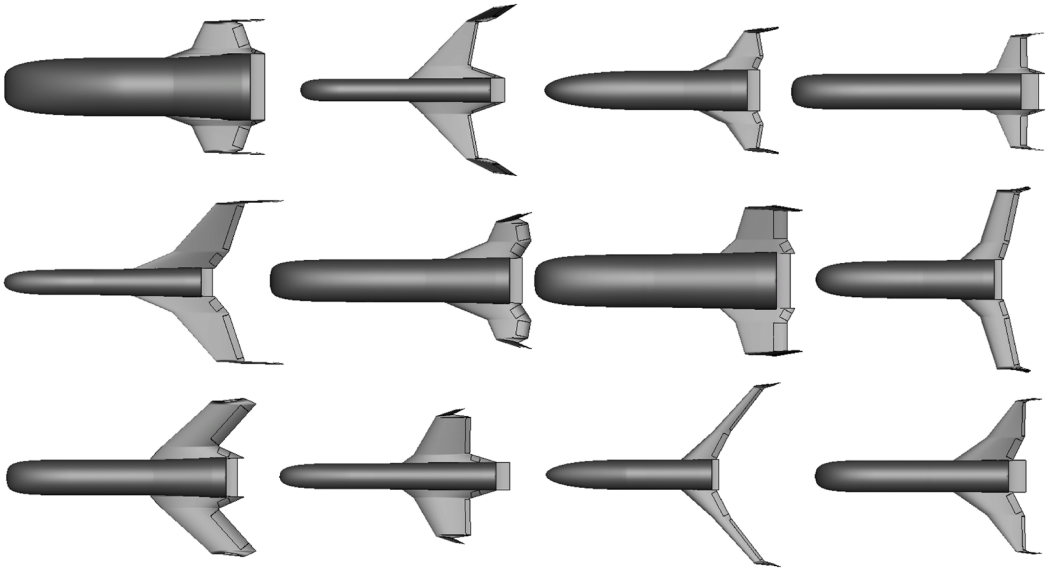

Figure 1: An Illustration of the Breadth of the Design Space

In general, model parameters were selected such that the user could not specify an impossible vehicle. For example, if the independent variables included the leading and trailing edge sweep angles, the root chord, and the half-span of the wing, it would be possible to select a set of incompatible parameter values: if the leading edge sweeps aft while the trailing edge sweeps forward, the two edges might intersect at a half-span distance smaller than what was specified. Thus, if that parameterization were used, individual parameter values which were within the allowed ranges could be combined to define a geometry that was infeasible. To avoid this, the parameters were chosen in such a way that geometrically infeasible vehicles were not possible. One drawback to this approach is that vehicles which would have very poor aerodynamic, structural, or operational characteristics *are* possible within the design space – for example, the vehicle in the third row, third column of Figure 1 might be expected to have structural challenges, and could be at risk of striking the wingtips during landing.

The aerodynamic surrogate models would relate the geometric parameters to the forces and moments acting on the vehicle. Because the dominant flow behaviors would change significantly with flight condition, it was decided that the effort would initially attempt to model the aerodynamics at individual flight conditions separately. If good results were

obtained, further research would seek to expand the models to capture the effects of changing flight conditions, as well.

The vehicle model included 7 control surfaces – a pair of elevons on each wing, a rudder on each vertical fin, and a body flap at the rear of the fuselage. Each control surface could be deflected independently to more accurately capture the control interactions. When the deflection angles for all control surfaces were combined with the rest of the geometric parameters, the resulting design space had forty-nine dimensions. Screening tests were performed to identify any parameters which did not significantly affect the responses of interest.

2.1 Screening Tests

A second-order screening design was used to build 129 configurations for testing. A set of 16 flight conditions was selected as representative of the overall envelope of the vehicle: Mach 0.3, 0.8, 1.2 and 4.0; α 10° and 30°; sideslip 0° and 5°. After simulating all geometries at each combination of flight condition parameter values, a sensitivity study was performed using JMP analysis software.[91]

The study results led to several observations. First, the Pareto Principle, which states that a small number of variables will often account for a large majority of the response behavior,[193] was not the behavior observed in this case. Instead, although individual responses were more strongly affected by some parameters than others, enough responses were strongly affected by different parameters that almost all of the design variables would have to be retained to capture the bulk of the response variation. Secondly, because all aerodynamic forces and moments were considered important to capture, it was observed that every design variable contributed significantly to at least one response at some flight condition. Given these observations, it was decided that all 49 design variables would be included in the subsequent modeling efforts in order to capture the most knowledge about the design space as possible.

2.2 *Identifying Similar Studies*

Previous studies that were identified during initial research offered little guidance when a sampling approach was being selected.

The Computerized Environment for Aircraft Synthesis and Integrated Optimisation Methods (CEASIOM) is a conceptual design tool intended to improve designers' ability to capture stability & control responses during the conceptual phase of design. The aerodynamics tool incorporates high-fidelity modeling, up to Euler or Navier-Stokes, to increase confidence in the results.[63] Multiple levels of analysis fidelity and adaptive sampling are used to identify the minima and maxima of the responses. Note that this tool was intended to be applied to a single, fixed configuration at a time, producing Euler-level aerodynamics for that configuration overnight. Thus, it is not well-suited for the large number of configurations considered during design space exploration.

Scharl & Mavris investigated the use of surrogate models for aerodynamic modeling of a subsonic transport.[171] Surrogate models were trained to reproduce the simplified aerodynamic model HASC, a potential flow model with a semi-empirical vortex lift model, to enable analysis of stability & control. This would allow efficient design of the empennage and control surfaces based on analysis rather than historical analogues. The design space included 21 parameters, such as Mach number, sideslip angle, angle of attack, and altitude. 3,500 random samples were used for training and validation of the surrogate models, which were built with artificial neural networks. Although this effort did incorporate parametric geometry, the sampling techniques used were somewhat simplistic. Additionally, due to the relative simplicity of the aerodynamic model, the results were not necessarily applicable to a study using higher-fidelity tools.

Masse & Wilhite[118] investigated the design of a reusable first-stage booster similar to the concept behind the RBS study. The concept was described using 11 geometric parameters, and APAS[45] was used to model the aerodynamics. The aerodynamic responses were sampled using a 3- or 4-level full factorial scheme, and a quadratic response surface equation was used to represent the results. The relatively low level of model fidelity, the sparse sampling scheme, and a lack of reporting regarding the accuracy of the resulting surrogate

models limit the similarity to the project at hand.

No studies attempting to model high-fidelity aerodynamics as a function of many geometric parameters were identified during the literature search. Lacking such guidance, experiments were designed to assess the effectiveness of different possible sampling plans.

2.3 Preliminary Experiments

2.3.1 Sampling Experiments

First, an **I-Optimal Design of Experiments** of 2,048 cases was constructed. The I-Optimal design distributes samples in such a way that, when a surrogate model is created from those samples, the average predictive variance is minimized.[91] Note that this design does not take any of the sampling results into account, and thus is an *a priori* sampling design. This DOE tends to emphasize the corners and edges of the space, which serves to minimize or eliminate the amount of design space for which the surrogate model would have to extrapolate.

In addition, a 2,500 case **Latin hypercube**[126] was generated. With this sampling approach, the user selects the number of samples to be performed, N, and the range of each design variable is divided into N equal portions, or "bins". Samples are then selected so that no two samples fall into the same bin for any input variable, which effectively creates a uniform sampling over each variable. The process of selecting these samples may be performed in multiple ways, such as *maximin* sampling which maximizes the minimum distance between any two sample points.[132]

To evaluate the accuracy of surrogate models, an additional set of test points were generated. These test points consisted of 2,000 random configurations. Each surrogate model would be used to make predictions for the response values at each point; these predictions would then be compared against the true values to determine which sampling & modeling approaches would be most effective.

Both designs were used to sample the design space at two flight conditions. One condition represented a plausible landing scenario at Mach 0.3, α 10°. The other condition represented an earlier phase in the gliding trajectory at Mach 3.0, α 30°. Both flight conditions had 0°

of sideslip. All 6,000 vehicles were modeled at each flight condition and the results compiled. A variety of surrogate models were created using these data sets in order to identify the best approach.

2.3.2 Surrogate Modeling Experiments

A number of different techniques exist for creating surrogate models. One of the most direct, known as **response surface methodology** (RSM), uses polynomial equations to describe the behavior of the response.[33] RSM can be very effective when applied to engineering problems, as the responses of many physical systems can be captured with second-order models. RSM models are commonly fitted using least squares methods to find the coefficients that best match the observed data.[91] If the response is more complex, higher-order terms may be incorporated, but this may be limited by the quantity of data available: there must be at least as many data points as coefficients to be estimated. Thus, it may be difficult to obtain sufficient data to fit a third- or fourth-order RSM model, especially if there are many design variables. Stepwise regression, in which higher-order terms are added iteratively to the model when they appear to be beneficial to model accuracy, was performed to investigate whether a partial increase in model order would improve the fit of the RSM models. Even when considering terms up to fifth order, model accuracy was still very poor.

Another common modeling technique is **Kriging**. Kriging is a statistical modeling method that represents the response behavior as a combination of some underlying mean function and a stochastic function that has a mean of zero which describes deviations from this mean.[170] Kriging is an exact interpolator, which means that it will exactly reproduce the known response values at the training points.[39] Fitting a Kriging model requires the inversion of a matrix of dimension n, where n is the number of training data points. As a result, fitting the model typically involves a computational burden of order $\mathrm{O}(n^3)$ and a memory burden of order $\mathrm{O}(n^2)$, which can become significant as n becomes large (on the order of a few thousand).[136] The DACE toolbar for Matlab[107] was used in an attempt fit Kriging models, but memory requirements exceeded the available resources when applied to this problem. As a result, Kriging was not used for this portion of the effort.

Artificial neural networks are a third technique for surrogate modeling that have been used for aerospace applications.[171] Neural networks consist of one or more hidden layers and an output layer. Each layer has a number of nodes, known as neurons, and each neuron has an activation function which combines values from the previous layer with weighting terms. Once the specific form of the activation function is chosen, the weighting terms are optimized to best fit the training data.[91] Hornik demonstrated that neural networks with a single hidden layer can act as universal approximators, reproducing any function to an arbitrary degree of accuracy, for sufficiently large networks.[82] This ability to match any function is offset by the problem of fitting the network; for n input values, each neuron in the hidden layer will have $n + 1$ free terms, and for m neurons the output layer will have $m + 1$ free terms. Fitting a network of m nodes to a problem with n input dimensions thus requires the optimization of $1 + m \times (2 + n)$ weights, which can be time-consuming.

Each of these methods was applied to the initial data set to determine the accuracy of the resulting model. One surprising result was that models trained using the I-Optimal points tended to perform worse than those that were not. A rule of thumb is that models trained using larger data sets tend to be more accurate; in this case, models trained using the combination of the Latin hypercube and I-Optimal data sets performed *worse* than models trained using only hypercube points. Recall that the I-Optimal DoE tends to sample the edges and corners of the design space; this result suggests that samples closer to the middle of the space provide better information about the behavior of the responses throughout the space. This opinion was bolstered by using a training set trained to fit points from the hypercube data set and part of the random data set. This *ad hoc* model was compared against a model trained only on hypercube points by applying both models to the unused random points. The *ad hoc* model, with a larger training pool of points distributed more or less evenly throughout the space, demonstrated better accuracy than the model with the smaller training pool.

2.3.3 Preliminary Conclusions

The initial experiments aimed to identify the sampling and modeling approaches that produced the most accurate surrogate models with the fewest samples. Based on the observed results, neural network models trained with space-filling samples, such as those from Latin hypercubes or random distributions, were the best choice. The models created using 2,000–3,000 points were still less accurate than desired, unfortunately, which indicated that more samples would be required to create useful aerodynamic surrogates for individual flight conditions.

2.4 Main Effort

2.4.1 Sampling Approach

Recall that Latin hypercubes are designed *a priori*, with the user specifying the desired number of samples in advance. If, after the experiments are completed, it is discovered that more data is needed to improve surrogate accuracy, it may be difficult to add more samples while still preserving the good space-filling qualities of the samples. Qian[152] has addressed this limitation by proposing a modified Latin hypercube approach known as **nested Latin hypercube** (NLHC) design.

NLHCs are generated by specifying the smallest hypercube size desired, a growth factor, and the number of nested levels to be included. Each level serves a space-filling hypercube. To illustrate the concept, consider a minimum hypercube of 5 points, a growth factor of 2, and 3 levels. The resulting NLHC will have 20 points, and all 20 points form a Latin hypercube. This is depicted in Figure 2c. Unlike most hypercubes, however, the first 5 points will also be a valid hypercube (Figure 2a), and the first 10 points will form a third hypercube (Figure 2b). This is a very useful quality if the user wishes to sample the space with a Latin hypercube but there is uncertainty as to the number of cases that should be analyzed. Due to the infill approach used, the levels of nested Latin hypercubes have geometric growth rates (e.g. the next-larger hypercube will be double, triple, etc. the size of the previous hypercube).

To sample the RBS design space, a NLHC was generated. Surrogate models trained with

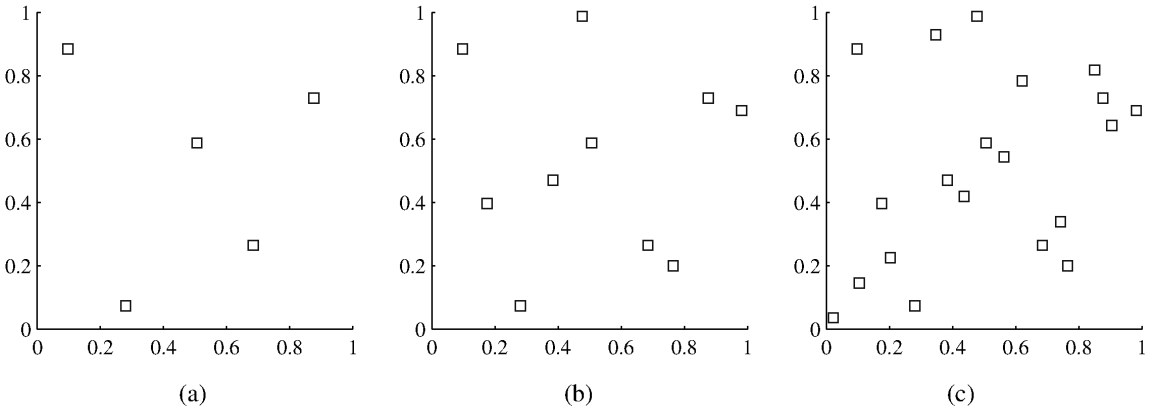

Figure 2: A Nested Latin hypercube with 3 levels: (a) 5, (b) 10 and (c) 20 points

3,000 points had shown moderately good performance, so it was expected that not many more points would be required for truly accurate surrogates. Thus, a NLHC was generated with a base hypercube of 4,000 points and larger hypercubes of 8,000 and 16,000 points. Based on the fit quality for 3,000 points, it was expected that 16,000 points would be far more than sufficient.

An additional 2,000 point hypercube was generated with the wing trailing edge fixed at the rear of the fuselage, as this was expected to a region of the design space where aerodynamic moments were close to zero. This additional sampling would thus enhance model accuracy in this region of the design space. An additional 2,000 random cases were generated to test surrogate model performance throughout the design space. Once all cases were defined, geometry generation with PaceLab began. A small minority of cases, less than 1%, failed to produce viable surface meshes as determined by the Cart3D meshing utility. This was deemed an acceptable degree of loss. Once built, cases could be analyzed with Cart3D.

2.4.2 Flight Conditions

Because the preliminary experiments failed to produce sufficiently-accurate surrogates at either flight condition, the main experiments also treated flight conditions as discrete, rather than continuous, variables. A separate surrogate model would be created for each response at each flight condition. Once adequate surrogate performance was demonstrated at each

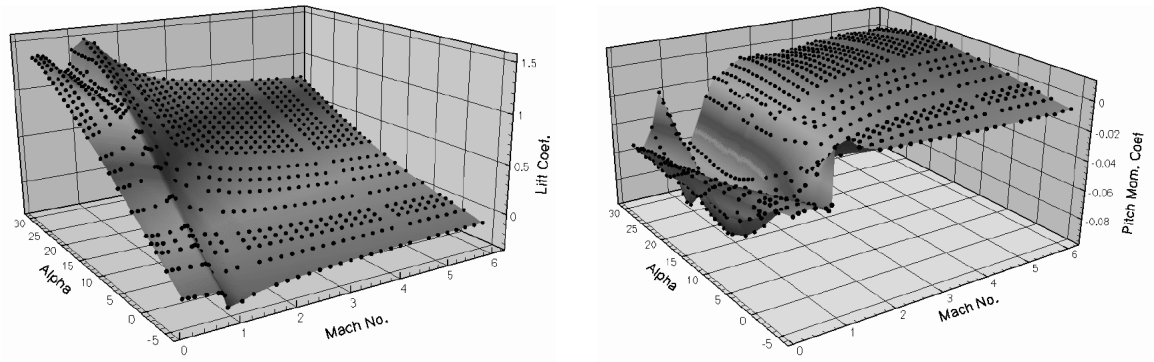

Figure 3: Cart3D Lift and Pitching Coefficients for LGBB (from AIAA 2003–3788)

flight condition, the surrogates would be expanded to interpolate between flight conditions and make predictions over the entire trajectory space.

The three flight condition parameters – Mach number, angle of attack α, and sideslip angle β – were discretized to cover the expected trajectory. The flight conditions that were selected for analysis were based in part on the Cart3D results for the Langley Glide-Back Booster published by Chaderjian et al.[26] These results demonstrated smooth, regular changes in each response for Mach numbers above 2; this contrasted with the relatively large variations observed at slower speeds. These variations with Mach number dwarfed those due to angle of attack. The lift coefficient and pitching moment coefficient for the Langley Glide-Back Booster at zero sideslip are displayed in Figure 3.

As a result, Mach number was sampled finely compared to angle of attack or sideslip, with particular emphasis on speeds below Mach 2. The flight conditions that were analyzed may be found in Table 1. Note that the flight conditions were combinatorial: each possible combination of Mach number, angle of attack, and sideslip angle was used. The lone exception was 15° sideslip, which was only used for Mach numbers 0.3 and 0.5. This produced a set of 48 flight conditions.

Each configuration was analyzed at every flight condition. As a result, even if only the smallest level of the nested Latin hypercube (4,000 cases) were analyzed, the full analysis would still require $4,000 \times 48 = 192,000$ analyses with Cart3D. Despite the relatively sparse sampling of flight conditions, this was a 60-fold increase over the 3,000 or so inviscid analysis

Table 1: Flight Conditions for Main Experiments

Mach number	0.3	0.5	0.8	0.9	1.1	2.5	4.0
Angle of Attack	0	15	40				
Sideslip Angle	0	5	15				

performed by Chaderjian et al. when a single configuration was being analyzed. The large increase in computational effort serves to illustrate the potential expense of design space exploration when higher-fidelity models must be used.

2.4.3 Running Analyses

To support this effort, students participating in this research project were granted access to computing resources at Department of Defense High Performance Computing Centers (HPCCs). These resources could complete a single analysis in thirty to sixty minutes, and could process hundreds or thousands of cases simultaneously.

These modeling efforts were terminated according to schedule constraints in order to leave sufficient time to create & test the surrogates and document the results before the end of the contract. The first 8,000 cases of the nested Latin hypercube were completed for each flight condition, as well as 4,000 additional cases for testing predictive accuracy of the surrogate models. The testing cases varied between flight conditions depending on which cases were available. When modeling ended, nearly 579,000 cases had been submitted for analysis. Some of those cases failed during analysis, or did not converge sufficiently by the end of simulation. After filtering out those unusable results, slightly more than 559,000 cases remained. Every flight condition had at least 10,000 cases, with an average of 11,500 cases per flight condition. This effort took roughly two months and consumed over two and a half million processor-hours of HPCC resources. If a contemporary quad-core desktop had been used for the analysis, it would have taken more than 73 years to complete.

2.4.4 Evaluating the Resulting Surrogates

It was something of a surprise to discover, in light of the number of analyses performed, that the resulting surrogate models still had somewhat poor predictive accuracy. The smallest

95% confidence interval for predicting pitching moment coefficient was ±0.26, and the average 95% confidence interval was ±0.83. These are very broad confidence intervals, given that the Langley Glide-Back Booster pitching moment coefficient never exceeded ±0.1 throughout its entire flight envelope.[26]

A variety of techniques were applied in an attempt to improve model accuracy. The scale of the task – 6 coefficients at each of 48 flight conditions, for a total of 288 responses – precluded the development of a custom approach for each surrogate. Instead, one flight condition (Mach 0.5, α 0°, β 0°) was selected as a test case to identify strategies that might improve all of the surrogate models.

Because the ranges of the design variables could produce aircraft with very large or very small wings, the reference areas by which the forces and moments were normalized could vary greatly across the design space. Models were fit to force and moment values that were only normalized by dynamic pressure to remove the effects of reference area, but no improvement in model accuracy was observed. Transformations of the responses – including squaring, square root, exponential and natural log operators – were applied, without improvement in accuracy. Outliers, identified by Mahalanobis distance calculations[115] in the statistical analysis software JMP,[91] were removed before fitting models, without any increase in accuracy.

One technique was found to be helpful. Surrogate models of pitching moment coefficient were more accurate if predictions for lift and drag coefficients were included in the training data for each point. This did not affect the accuracy of the lateral moment models, but the 95% prediction confidence interval for pitching moment at the test flight condition shrank from ±0.5 to ±0.4. This was considered to be sufficient improvement that it was worth the effort, and was used as the model training method for moments at all flight conditions.

As previously stated, the most accurate surrogate model for pitching moment coefficient had a 95% prediction confidence interval of ±0.26. This model was at a flight condition of Mach 2.5, α 0°, & β 5°. With a prediction confidence interval that large, even if a certain configuration is predicted to have exactly zero net pitching moment, there is so much prediction uncertainty that there is a 44% chance that if the configuration were analyzed

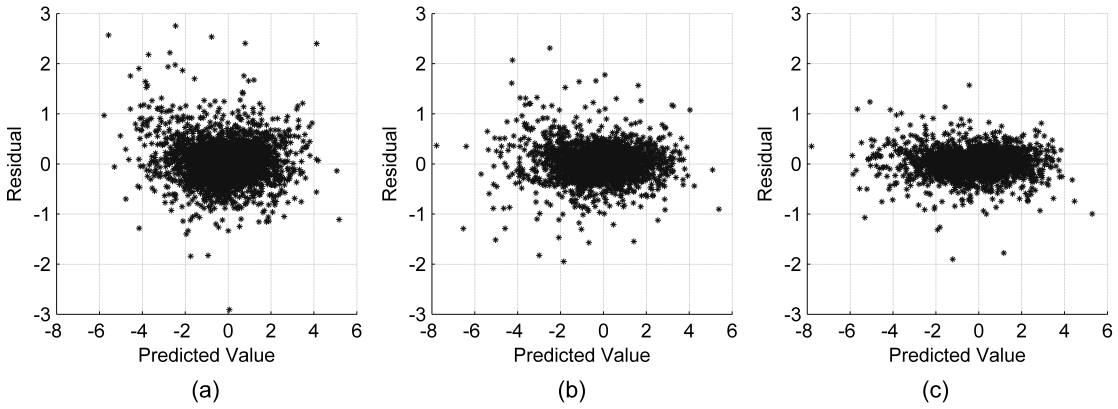

Figure 4: Prediction Error for Models Trained on (a) 4,000 samples, (b) 8,000 samples, & (c) 16,000 samples

with Cart3D it would be found to have a pitching moment coefficient so large it would be uncontrollable. Pitching moment models for other flight conditions had larger uncertainties, which corresponded to lower confidence in predicted performance.

In light of the observed model performances, it was decided to analyze all available configurations at a single flight condition to estimate the number of cases necessary for some particular model confidence. Twenty thousand configurations were analyzed at Mach 0.5, α 0°, β 0°. These configurations included all 16,000 cases in the NLHC, along with the 2,000 random cases and the 2,000 cases with the wing aligned to the rear of the fuselage. Surrogates were trained using each level of the NLHC, i.e. 4,000, 8,000, and 16,000 cases, and the predictive accuracy of those surrogates was tested using the non-NLHC cases. The prediction error distributions of the three surrogates are plotted in Figure 4.

The 95% prediction confidence interval for the 4,000 point model was ± 0.73. For the 8,000 point model, the same interval was ± 0.56, and for 16,000 points the interval was ± 0.42. The residual, which is the discrepancy between surrogate prediction and true response value, grew smaller as the pool of training data was enlarged. However, it shrank relatively slowly; the increase in accuracy between the second and third model was *less* than between the first and second, despite adding *twice* as many samples to the training pool. This suggested that a highly accurate surrogate model would require a very large number of samples.

A very rough estimate was made of the rate of convergence relative to the number of

training points. Linear, exponential, and logarithmic regressions were made of the data points, with the logarithmic regression matching the most closely ($R^2 = 0.997$, which indicated that less than 1% of the variation in the response was not being captured). The results suggested that as a loose approximation, 60,000 samples would be required at this flight condition to achieve a 95% prediction confidence of ± 0.15. At this prediction confidence, if a configuration were predicted to have zero net pitching moment, the chance that Cart3D analysis would reveal a true pitching moment larger than ± 0.1 was less than 1 in 3.

Analyzing that number of cases at each of 48 flight conditions would amount to nearly 2.9 million analyses, i.e. roughly five times the expense invested in the RBS project. Such an effort, using the same high-performance computing resources, would take a year to complete. This would require such an investment of time and effort that it might be considered impractical; on lesser computing systems, it was almost assuredly infeasible.

The research objective – accurate aerodynamic surrogate models for a large, multi-dimensional design space and many flight conditions – appeared to be out of reach unless a superior approach could be developed.

2.5 Research Questions

Eggers[50] and Chaderjian et al.[26] each demonstrated that Euler-level aerodynamic models were moderately successful at capturing the performance of reusable-booster-type vehicles. In addition, Pamadi et al. demonstrated that lower-fidelity analysis tools, such as APAS, had significant difficulty modeling a reusable booster vehicle, in part due to the nonlinear behavior of pitching moment at $\alpha > 15°$ for that vehicle.[144] This was a critical shortcoming, as a substantial portion of the RTLS trajectory may take place at such angles.[79, 98, 144]

This suggested that for vehicles like reusable boosters, where nonlinear effects are likely to be significant, Euler aerodynamics may be the minimum level of fidelity capable of adequately capturing vehicle performance. Simpler, faster models exist, but are known to be deficient for conditions which may constitute a large portion of the return trajectory. As a result, using those faster models to support design decisions introduces the risk that the

configuration(s) selected for further simulation will later be found to have deficient performance, as occurred during the DLR reusable booster effort described in Section 1.7.[52] This creates a strong incentive to use higher-fidelity tools early in the design process, before parameter values are fixed.

However, higher-fidelity modeling may be too expensive for use early in the design process. The execution time required to analyze each concept with a highly accurate model, at minutes or hours per configuration, can be prohibitive. The example of the RBS project illustrates how, although surrogate modeling enables the rapid estimation of the results of said highly accurate model, the training of useful surrogate models may also require excessively large quantities of data. Thus, no matter how desirable it may be to incorporate higher-fidelity tools in the early stages of a design project, the current state of the art makes this unlikely or impossible.

This leads to the primary research question:

Research Question 1 ***How can high-fidelity modeling be feasibly applied earlier in the design process, despite the computational expense?***

The RBS project illustrated some of the critical technical challenges that must be addressed before high-fidelity modeling can be used for design space exploration. The sampling of flight conditions used by the RBS project was relatively sparse. Referring back to the aerodynamic results for the Langley Glide-Back Booster depicted in Figure 3 on page 30, both the lift coefficient and the pitching moment coefficient exhibit variations with Mach number and angle of attack that would not be captured by the sampling used for RBS. This is especially true for the variations in pitching moment at subsonic speeds.

Yet despite the coarse sampling, the computational effort expended during the RBS project was extremely large. Finer sampling would drive the required effort up even further. In addition, the computational work took place on multiple state-of-the-art parallel computing systems simultaneously; improved computational resources for faster processing are not likely for most near-future design programs. If improved resources are not likely to be available, *the cost to generate the data must be reduced.*

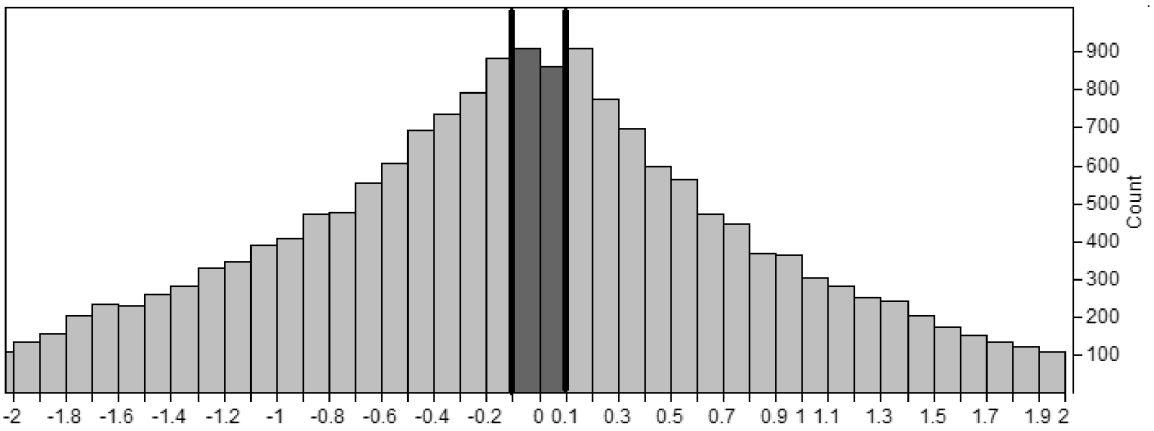

Figure 5: Partial Distribution of Pitching Moments at Mach 0.5, α 0°, β 0°

2.5.1 Emphasizing Useful Regions of the Design Space

A great deal of effort was expended analyzing configurations throughout the design space. If the goal is to reduce computational expense, the question might be raised: are all results of equal value? If some results are more useful than others, the computational expense of the effort might be reduced by increasing the proportion of informative, valuable samples.

Although the performance of every vehicle is different, rules of thumb can be a useful way to capture trends or patterns in behavior common to many vehicles. Carpenter[24] suggested that, for the range of flight conditions relevant to reusable booster systems, configurations with pitching moment coefficients within ± 0.1 should be considered likely to trim given reasonably-sized control surfaces. Vehicles exhibiting larger moments may have difficulty achieving trim. When the data from the RBS study was investigated, it was found that a large majority of the configurations exhibited pitching moment coefficients beyond that range, sometimes by a large margin. The distribution of pitching moment coefficients at Mach 0.5, α 0°, β 0° (the flight condition with the largest number of available results) is shown in Figure 5.

More than 90% of the configurations tested experienced pitching moments so large that they were not likely to be a feasible vehicle. This suggests that much of the design space is of little interest to vehicle designers. Analyzing so many configurations with such poor

performance was directly contrary to the goal of maximizing the amount of useful information gained from each experimental analysis. Furthermore, if it were known that a certain region of the design space had poor performance – i.e., designs in that region were unlikely to trim – that region could be sampled less intensively, which in turn could reduce the overall modeling cost.

Although a surrogate model that is highly accurate throughout the entire design space is intellectually satisfying, the RBS effort demonstrated that it may also be unacceptably costly. A simpler solution may be available: surrogate models are more accurate for cases similar to those used to create the surrogate.[92] By selectively placing samples in promising regions (i.e., those with aerodynamic moments close to zero), models trained to fit the resulting data set could exhibit improved prediction accuracy in those regions. Regions with poor performance need only be sampled enough to identify them as uninteresting.

This sampling strategy would attempt to emphasize configurations with good performance. Surrogate models trained on the resulting data pool would have high accuracy in well-sampled regions, and lower accuracy in sparsely-sampled regions. If the sampling strategy is successful, the result would be a surrogate model that has good prediction accuracy with regard to regions where the moments are close to zero, and sufficient prediction accuracy in the other regions of the design space that those regions could be identified as unattractive. This falls short of a surrogate that is very accurate across the entire design space, but it *does* promise a surrogate that is very accurate in regions of interest to the designer.

The field of adaptive sampling, which chooses the next experiment based on previous results, is well-established, and an overview will be given in the next chapter. Most of the literature focuses on maximizing or minimizing some response value, however. This may be of little use for an effort such as this, when the goal is to find the regions of the design space where the response or responses take values within certain ranges.

This leads to the next research question:

Research Question 2 ***When "good performance' refers to responses within desirable ranges rather than maxima or minima, how can regions of good performance be identified and emphasized during the sampling process?***

Such a sampling process would minimize the number of expensive analyses that must be performed before a surrogate of useful accuracy could be trained, by restricting the regions of useful accuracy to only those likely to contain feasible designs. However, the reduction in modeling expense that would be necessary to bring high-fidelity surrogates into the realm of feasibility is likely to be of an order of magnitude or more. It may be overly optimistic to expect that an improved sampling plan alone would be sufficient.

A large factor in the overall cost is the per-analysis cost of Cart3D. If this cost could be brought down, or the dependence on Cart3D analyses mitigated, the total modeling cost could be reduced by a significant margin.

2.5.2 Reducing Dependence on Expensive Models

Pamadi et al compared the aerodynamic accuracy of linear aerodynamics, Euler CFD, and wind tunnel data for the Langley Glide-Back Booster.[144] Figure 6 displays this comparison near Mach 0.3. Similar results for Mach 1.2 and 4.5 may be found in the source document. Although the linear aerodynamics predictions diverged from the other data for $\alpha > 10^\circ$ or so, the results at smaller α all agreed well. The running time of APAS, the low-fidelity tool used by Pamadi et al., is less than one second on a contemporary quad-core desktop computer. That is orders of magnitude faster than the hour or so to complete a Cart3D analysis on the same number of processors. While APAS may not be sufficiently accurate at high α to act as the *only* source of aerodynamic data over the flight envelope, the significant cost reduction and relative accuracy for APAS makes it attractive as a source of data if the results of different levels of fidelity can be combined.

This raises the second research question:

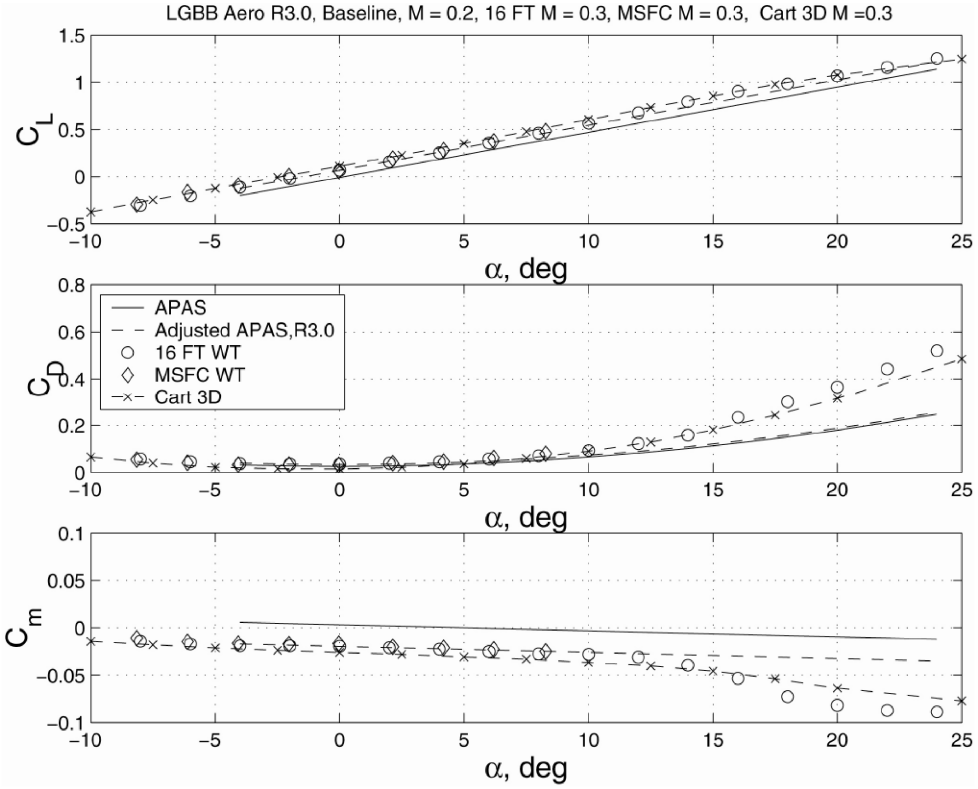

Figure 6: Comparison of APAS, Cart3D, and Wind Tunnel Results for LGBB near Mach 0.3

Research Question 3 ***How can cheaper analyses be integrated with high-fidelity models to reduce the overall cost of design space exploration or exploitation?***

If less-expensive sources of data can be incorporated, cheaper APAS samples can be used to reduce the costs of data generation. Those samples can form the bulk of the training data set. A set of samples will also be analyzed with Cart3D to evaluate the agreement between the two sets of results. At best, the APAS results will agree closely with the Cart3D results, and few Cart3D analyses will be required. It seems prudent to expect that this will not always be the case.

Remember, however, that discrepancies between the two models are unlikely to be random; instead, for the most part the discrepancies are likely to result from the phenomena captured (or neglected) by each model. As a result, it seems plausible that patterns in the discrepancies could also be captured by surrogate models, allowing the correction of results obtained via APAS to values similar to those that would be obtained via Cart3D.

In the worst-case scenario, APAS results will offer no insight into the behavior of the more expensive Cart3D results. In this case, the computational burden would be almost unchanged from what would be expected without the use of multiple analysis tools: the cost per execution of APAS is orders of magnitude smaller than that of Cart3D, so hundreds or thousands of APAS results could be obtained for the amount of effort required to analyze a single extra case with Cart3D.

2.5.3 Uncertain Data

The final observation made from the RBS project results was the poor quality of the surrogates for lateral responses. One of the measures of accuracy for surrogate models is R^2, which quantifies the amount of response variation that is captured by the surrogate; a perfect surrogate would have an R^2 value of 1.0.[134] Many of the surrogates created in the RBS project had R^2 values above 0.95. The average R^2 value for α 40°, β 0° rolling moment surrogates was 0.25, which is extremely low. This suggests that the surrogates are not able to capture very much of the observed behavior of those responses.

Upon closer inspection, the solution history for some Cart3D cases was found to exhibit an oscillatory behavior, even after the solution had essentially converged. This oscillatory behavior is illustrated in Figure 7. This figure plots the aerodynamic moments acting on a notional business jet model, which was included as a test problem with Cart3D. For this demonstration the angle of attack was increased over that of the test problem, from 3° to 40°, and the Mach number was increased from 0.84 to 0.9. This produces a large region of separated flow over the wing similar to what was observed in some sample booster configurations during the RBS study. This is not likely to be an important flight condition for a business jet. Instead, this example mimics an experiment performed late in the RBS study.

A cursory review of the time history of the three moments, depicted in Figure 7a, shows that they have essentially converged long before the simulation is terminated. Closer inspection, however, reveals small variations or "jitters" – the solution becomes mildly oscillatory near iteration 300. This is of particular interest since the response values presented as output by Cart3D are simply the last values calculated for those responses; if the solution is exhibiting noisy behavior rather than being perfectly converged, that noise will be present in the response values that are reported. For the pitching moment coefficient (shown in Figure 7b) the magnitude of the oscillations is roughly 10% of the average value of the response. For yawing moment coefficient (Figure 7c) the variations are of equal or larger order of magnitude as the average response value. Rolling moment exhibited similar behavior to yaw and is omitted for visual clarity. The two lateral responses can be considered comparatively noisy, resulting in poor surrogate model accuracy.

An investigation identified that this oscillatory behavior was the likely culprit for the poor accuracy of the lateral surrogate models. The oscillations appeared to stem from the large-scale flow separation that most or all vehicles experienced at high angles of attack and/or transonic conditions.

There was some concern that the oscillatory solver behavior was being caused by some aspect of the custom PaceLab tool that was used to create the surface meshes of the configurations being analyzed. To determine whether this was the case, an example problem

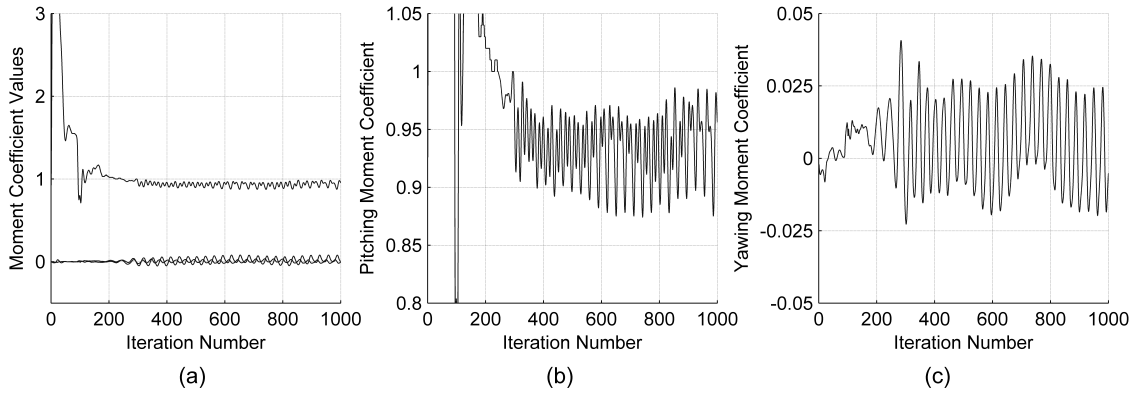

Figure 7: Oscillatory Solution Behavior

included with Cart3D – the business jet model mentioned earlier – was analyzed at one of the conditions with poor surrogate behavior. The oscillatory solution was again observed, suggesting that the oscillations stem from the juxtaposition of an inviscid flow solver and flow conditions that are strongly dependent on viscous effects.

When the reported response is affected by some characteristic of the model used, the effect can take two forms: a bias effect or a random effect. Bias-type effects often result from the simplifying assumptions in a model, and may have a consistent sign or even a predictable magnitude. Many multi-fidelity modeling techniques take advantage of such predictability to "correct" the responses from a simpler model to match the results from a more complex model. For example, by neglecting viscous effects, Cart3D does not capture drag due to skin friction, and so its estimated drag coefficients are likely to be less than what would be measured in flight testing. If designers are aware of this characteristic, however,corrections can be made: by adding a constant C_{D_0} to inviscid Cart3D results, Aftosmis et al. were able to closely match much of the drag polar computed by viscous models for a RAE 2822 wing.[7] Less-accurate results were obtained when the same approach was applied to the DLR–F4 wing-body configuration. Scharl & Mavris observed similar results.[172]

Effects that are more random in character, on the other hand, are harder to account for. This is particularly true for models that must iterate to reach a solution, as the final result may have some degree of oscillatory behavior. When the responses are of different orders of

magnitude, oscillatory behavior that is negligible for one response may significantly affect another. If those oscillations are large enough relative to the magnitude of the response, the reported response value could be overshadowed by the noisy effects of the oscillations. A surrogate model which treats those cases as deterministic – that is, as a precise representation of the response value for that case – may attempt to reproduce the noisy component as well, with negative consequences for its predictive accuracy. It is believed that this is what occurred for some surrogates for lateral responses during the RBS project.

Artificial neural networking, the surrogate modeling approach used in the RBS project, approximates the data rather than exactly interpolating it. Small-scale variations, such as the iteration noise described above, might not affect the resulting model if the magnitude is small. The poor R^2 values observed in the RBS project indicate that for many lateral responses, the variations were large enough to have a negative effect on the predictive accuracy of the models.

Rather than treating all data as deterministic and precise, it may be possible to estimate the amount of uncertainty present in the data. For example, in addition to the force and moment summaries produced by Cart3D after an analysis, the iteration histories of those responses are recorded in a separate file. Using these histories, the amount of uncertainty in the responses can be calculated. However, even if such information were available, it could not be incorporated into the neural network model, leading to the final research question:

Research Question 4 ***How can information about uncertainty in the data be captured and incorporated effectively?***

The scripts used to run Cart3D can parse the force and moment iteration histories and report the average and standard deviation values of each. This information is already used to ensure that any case which did not adequately converge could be discarded. The information is then retained but not used during the modeling, as the neural network tools that were used could not incorporate the uncertainty data. Modeling techniques which *can* incorporate uncertainty data would have a ready source of noise estimates.

2.6 Review

The RBS research project undertaken by the AFRL and ASDL illustrated one of the major reasons why extensive design space exploration is seldom undertaken using expensive models: the computational effort required to explore the space is enormous. While the project was not successful at creating true aerodynamic surrogate models, it did highlight particular aspects of the problem that were particularly troublesome. If these difficult aspects could be addressed, the cost of such an effort might be reduced sufficiently to make the effort feasible. A set of research questions have been formulated to guide the efforts to address those difficulties. These research questions directed the literature search, which is presented in the next chapter.

CHAPTER III

REDUCING THE COST OF INFORMATION

The previous chapter demonstrated how design efforts could benefit if the cost of information could be reduced. A variety of approaches have been proposed toward this end. The simplest and most obvious is the increase in computing power that occurs over time. The tools that had been cutting edge are becoming cheaper to use. APAS, an analysis tool with over 30 years' pedigree, took 45 seconds per analysis on a mainframe in 1981,[45] while today a similar analysis can be performed on a common desktop computer in less than one hundredth of that time. If a desired trade study is too expensive to be performed with the given resources, the simplest response would be simply to wait. After enough time, the infeasible becomes the feasible, and the feasible becomes the commonplace.

While true, this is more of a trivial solution than an effective strategy. Programs have deadlines and opportunities can be missed. What is needed is an efficient approach to obtain trustworthy information, particularly when many design variables must be considered. Such an approach could be particularly valuable if it were general and not bound to a particular tool. Once developed, the approach could be used to investigate the design space and identify important trends and interactions in the response behaviors. That knowledge could then be used in support of future project decisions, such as selecting the appropriate parameters and ranges for a future trade studies.

There are two predominant families of techniques for gaining information about a problem when the cost per function evaluation is not negligible: optimization and surrogate modeling. Both have long pedigrees in engineering analysis, and each has strengths and weaknesses that make it more or less appropriate for certain applications.

3.1 Summary of Optimization & Surrogate Modeling

Optimization is the process of identifying the "best" values for input parameters. The quantification of "goodness" for a set of input values is made using some objective function,

which is often a weighted combination of the analysis outputs. The process may be considered a directed search of the design space; points with poor objective scores are considered to be uninteresting, and knowledge about the entire space is only considered useful to the extent that it helps to identify cases with better scores. This is an iterative process, relying on results from previous analyses to select the next point for evaluation. Techniques for selecting the next case may be as simple as random selection, or may incorporate detailed information such as the local gradient of the objective function.

Gradient-based optimization uses information about how the optimization objective varies with small changes in the input settings. This information identifies the direction of maximum improvement in the objective function, and can be used to very quickly find an optimum solution. An optimum is a point where any small change in the inputs would negatively affect the objective score. Such an optimization scheme can be rapid and efficient if the calculation of local gradient is not expensive.[72] Complex objective functions may have more than one local optimum, though, and gradient-based optimization risks becoming trapped at a point which is better than those around it but *not* the best point possible – a local optimum rather than the global optimum.

This risk may be addressed through the use of repeated optimizations starting from different points throughout the space, or even stochastic methods like genetic algorithms, but there is no guarantee that those techniques will identify the global optimum. Optimizations may still be constrained by resources and running time, and it may be too expensive for an extensive design space search.

No matter what optimization technique is used, the results will depend on the choice of objective function. Crucially, the results of an optimization cannot tell the investigator what the optimum might be for a different objective. Should another response be included or the relative weights of objectives be changed, the point selected by the previous optimization is unlikely to be the optimum for the new objective function. The optimization process will have to be repeated every time the objective is altered, making it difficult to play "what if" analyses unless the optimizations are both rapid and inexpensive. For this reason, optimization methods are seldom used for design space exploration unless the objective is

known and unlikely to change, such as maximizing lift-to-drag ratio.

The other primary approach to acquiring and leveraging knowledge about a problem through experimentation is through the use of **surrogate models**. Surrogate modeling methods are a collection of techniques for interpolating or curve-fitting experimental results in order to estimate the value of responses at other, unsampled points.[156] Surrogate modeling also has a long history of use in engineering analyses, with an increase in the complexity of techniques used over time. A major strength of surrogate models is the execution time: once a model has been developed, predictions can made extremely rapidly. This can lead to large savings when the original model requires substantial time to complete an analysis.

Many surrogate modeling techniques place strong emphasis on reproducing the entire range of the response than on regions deemed to be "good." This allows the model to be applied with equal confidence to any point in the design space. As a result, the data points used to build the surrogate may be selected in advance. Research into the best methods for selecting which data points to run is often collected under the title, "Design of Experiments" or DoE. A shortcoming of surrogate models is that the amount of data required to train them may grow large, especially if the response behavior is complex or there are many input parameters.[72] Many experimental design techniques have been developed in the quest to gain the most knowledge about the response for the least cost.

Note that the two approaches described here, optimization and surrogate modeling, are not mutually exclusive and have been blended in various fashions. One popular technique is to create a surrogate model of the outputs of an analysis tool and, during an optimization, use hat surrogate to estimate objective function values instead of using the analysis tool itself. This can significantly reduce the computational expense of each function call, particularly when evaluation-heavy optimization techniques such as genetic algorithms or gradient evaluation through finite differencing are used.

Other researchers have used optimization techniques to select the sample that would be most informative for training a surrogate model. The process of selecting the next sample based on prior results is known as *adaptive sampling*. The optimization process may aim to identify the points with the maximum prediction uncertainty, points that produce a

maximum or minimum response value, or points that produce a particular response value. This topic will be revisited in greater detail in Section 3.4.1.

Despite all the work that has been done in these fields, designers still encounter problems – like the RBS study – which are constrained by computational costs. Problems involving complex features, such as transonic flow, typically require high-fidelity modeling to accurately capture response behavior. Those high-fidelity models may be very computationally expensive. Secondly, if a vehicle must operate at a wide variety of flight conditions, it may not be possible to eliminate many design variables through screening tests, as was the case in the RBS study; the variables that are unimportant for one response or flight condition may strongly affect another, resulting in a large number of dimensions which must be considered. Problems with many dimensions are typically expensive to address through surrogate modeling and certain optimization techniques. Finally, some investigations cannot adequately be captured by the objective function of an optimizer. An investigation of how trailing edge sweep angle might affect adverse yaw behavior, for example, might be difficult to express as a maximization or minimization of some objective function.

Faced with challenges such as these, it can be difficult for designers to investigate the design space in a cost-efficient manner. Surrogate models can be very expensive to train, especially as the necessary prediction accuracy increases, and while some optimization strategies may be cheaper to execute they offer no indication of how the result might change if the objective function were altered. If designers address the problem inefficiently, they may expend their resources without collecting enough information to support future decisions, and the project would be exposed to the risk of dead ends, backtracking or even failure. This provides the incentive to research techniques that might address some of the factors which make these problems so difficult. These techniques include:

- methods to combine data at multiple levels of fidelity to reduce the need for high-fidelity modeling wherever possible;
- adaptive sampling to reduce sampling in uninteresting regions of the design space;
- techniques to estimate the uncertainty of a response to quantify confidence in each

data point; and,

- techniques to incorporate information about uncertainty present in the data when a surrogate model is trained.

The next section will address the first of these topics, methods to combine data from multiple levels of analysis fidelity.

3.2 Multi-Fidelity Methods

Multi-fidelity methods, also called **data fusion** techniques, are techniques which combine data from multiple separate sources into a unified whole. For the most part, those data sources are models with different levels of complexity, and therefore with different costs per analysis. Although it is usually trivial to extend a data fusion method to handle any number of contributing models, this description will assume that two models will be used; this assumption is made for the sake of descriptive simplicity. Additionally, the description will assume that all analyses are computational models, but it should be noted that the methods are equally effective for experimental data sources such as wind tunnel or flight testing.

Typically, the cheaper data source achieves its cost savings by simplifying or neglecting one or more phenomena important to the scenario at hand, making it too inaccurate for use alone. Surrogate models may also replace the cheaper data source, further reducing the per-analysis cost without affecting accuracy. Conversely, the costlier source captures these phenomena accurately but the per-experiment expense makes it undesirable or impossible to generate all the training data required for an accurate surrogate model. By combining the two, an accurate surrogate may be trained at less expense than if the more accurate source were used alone.

If the execution time of the low-fidelity data source is not trivial, it may be worthwhile to create a surrogate model of the low-fidelity results. This surrogate model, rather than the original source, is then used as the low-fidelity data source thanks to its more rapid execution time.

3.2.1 Additive & Proportional Correctors

The simplest approaches to combining data of different fidelity levels are the bias and proportional corrections. These approaches are most useful when the high-fidelity analysis is so expensive it cannot be performed more than a handful of times. The low-fidelity model is used to explore the entire region of interest and the predictions are recorded. The high-fidelity analysis is then applied to a few of the same cases and the results compared.

If an **additive corrector** (also called a bias corrector) is used, the difference between the two is treated as a bias error in the low-fidelity model, and a "high"-fidelity approximation can be obtained by adding this bias estimate to each low-fidelity result. This method is most effective when the bias is not expected to vary within the design space. For example, many simple aerodynamic models neglect viscous effects and thus cannot capture skin friction drag; because skin friction drag is not strongly affected by angle of attack, a more accurate drag polar can be obtained by calculating the skin friction contribution for one case with a higher-fidelity tool and adding it to the drag polar produced by the low-fidelity analysis.[7, 172]

An alternative approach is to treat the discrepancy as **proportional** – that is, to assert that the low-fidelity result is always some fraction of the high-fidelity result. To correct for this, rather than being augmented by some constant value, the low-fidelity predictions are multiplied by some constant value. These two approaches to correcting the less-accurate predictions encompass a majority of multi-fidelity methods in the literature.

Naturally, these corrections are not restricted to constant values. If multiple high-fidelity analyses are possible, a proportional correction may be calculated. This proportional correction is a linear function of the input variable(s), and helps to capture errors in the lower-fidelity analysis that are not constant. This is helpful when the low-fidelity trend is in the correct direction but of an incorrect magnitude. Higher order corrections, in which the Taylor series approximation is extended to include additional terms, have also been described;[60] these corrections may require large quantities of data and thus become very costly.

The complexity of the correction is limited only by the quantity of data available for comparison. Researchers sometimes create surrogate models of the discrepancies between

the data sources to more accurately estimate the high-fidelity result, treating the low-fidelity contribution as cheap or trivial. This approach is often cheaper than creating a surrogate model of the high-fidelity result directly, given that the low-fidelity predictions will capture some or most of the high-fidelity behavior.

Examples of efforts incorporating one or both of these approaches abound, primarily focusing on two levels of fidelity.[28, 36, 48, 73, 155, 194] Some variations on these methods are of particular interest and shall be described in greater detail.

Huang et al.[83] creates a separate surrogate model for each level of fidelity, capturing the deviations between the current level and the next-simpler level. These models are fitted sequentially rather than simultaneously, simplifying the task of parameter estimation. Kennedy and O'Hagan[94] combine the additive and proportional correctors, building a linear model for the proportional corrector and fitting a Gaussian Process to the remaining discrepancy for use as an additive corrector. They refer to this process as *Bayesian calibration of computer models.* Note that fitting such surrogate models requires the inversion of covariance matrices which incorporate *all* training data points rather than only those at a single level of fidelity. As a result, this method can become computationally arduous for large data sets.

Qian and Wu[154] and Xiong et al.[194] expand on this approach, replacing the simple linear model with another Gaussian Process which allows the proportional factor to vary as a function of input settings. Gano et al.[60] demonstrated a hybrid corrector, constructed as a weighted sum of additive and proportional correctors. The weighting function evaluates nearby data samples to determine the utility of each correction style, eliminating the need for the user to choose one or the other *a priori.*

Although the additive or proportional correction methods are far and away the most common in the literature, they are not the only methods possible. Research indicates that, rather than training models to transform the output of the inexpensive models, useful high-fidelity surrogates may be developed via a scaling of the *inputs* of the models. Robinson et al.[167] proposed an optimization algorithm which utilizes space mapping, a technique for transforming the inputs of the lower-fidelity model in such a way that the model outputs

match the higher-fidelity results to at least the first order.

3.2.2 Cokriging

Other multi-fidelity methods exist as well. The surrogate modeling technique known as Kriging has been expanded to incorporate data from multiple sources. Regular Kriging calculates the covariance between each data point, defined with one covariance equation and one vector of model parameter values. The user may capture the relationships within and between multiple sets of data, by expanding the model to include multiple covariance equations. This family of similar techniques is known as **cokriging**.[182] Due to the extra covariance equations, the model has more parameters which must be estimated, making the fitting of a model a more complicated endeavor.

Additionally, multi-fidelity efforts often incorporate a large quantity of lower-fidelity data due to its relative cheapness. The estimation of optimal values for cokriging model parameters requires the inversion of the covariance matrix, which carries a computational cost of order $\mathrm{O}(N^3)$ where N is the number of data points. As a result, when a model is trained on a large data set, the training process may become very memory- and computationally-intensive.[136] This may become infeasible for data sets larger than a few thousand cases.

To simplify the problem somewhat, some researchers[56, 184] make the assumption that the lower-fidelity data is independent of the higher-fidelity results. A Kriging model is then trained to reproduce the lower-fidelity data. After that, the discrepancy between the higher- and lower-fidelity responses at each high-fidelity sample point is calculated, and another Kriging model is trained to fit the discrepancy behavior. This reduces the number of model parameters being simultaneously optimized, and blurs the distinction between cokriging and the additive/proportional correctors described above. However, note that even after the model has been fit, applying it to predict the response value or prediction variance at some point would require the inversion of a covariance matrix that incorporates all data points at all levels of fidelity, although if the inverted matrix is saved it need not be re-inverted to make subsequent predictions. This is similar to the method proposed by Kennedy & O'Hagan[94] mentioned earlier, and like that method can become computationally expensive when applied

to large data sets.

Ghoreyshi et al.[63] uses a variant of cokriging to combine low- and high-fidelity data to reduce the cost of building an aerodynamic database for a new configuration. In this approach, the Kriging surrogate model is first trained to reproduce the low-fidelity data. Next, that surrogate is used to make predictions for the response value at each of the sites of the high-fidelity data. The low-fidelity predictions are then treated as an extra input dimension, and a Kriging model of the high-fidelity data is fit to the combined input data. This form of cokriging, which will hereafter be referred to as "**Ghoreyshi cokriging**," does not require cross-covariance terms, and has computational expense $\mathrm{O}\left(N_{lowfi}^3 + N_{hifi}^3\right)$ rather than $\mathrm{O}\left((N_{lowfi} + N_{hifi})^3\right)$.

Yamazaki and Mavriplis[195] use an expanded version of cokriging to construct their variable-fidelity model, combining up to three sources of data simultaneously. Gradient information is generated using adjoint calculations or automatic differentiation of the analysis tool and may also be incorporated in the surrogate model through additional covariance terms. This gradient information may be used in a Taylor series approximation to estimate response values at other points close to a known point; these estimated values are treated as lower-fidelity data because the response values are estimated, not known.

3.2.3 Data Harmonization

Whereas cokriging attempts to model responses that are nominally different but correlated, Baume et al.[14] propose an approach which combines multiple sources for a single response which they call "**data harmonization**." Unlike cokriging, which implicitly assumes that the responses being modeled are correlated but not identical, data harmonization aims to combine disparate sources of information about a single response. This philosophy lends itself more easily to the prediction of aerodynamic responses using multiple models of different fidelity.

The motivation for this approach was the creation of a unified model of environmental data across national boundaries, using data sets collected by European nations. After selecting gamma ray dosage as a test data set, clear biases and variations were identifiable between

sensor results from different countries. If these biases and variations were not addressed, predictions for gamma ray dosage from the resulting models would exhibit low prediction confidence.

To account for these factors, Baume et al. introduce the data harmonization approach to modeling. This approach is similar to universal Kriging, in which a polynomial mean function is fit to the data while discrepancies from this mean are captured through a covariance matrix. In the gamma ray example, the effects of altitude were subtracted as a known factor, soil composition was treated as an unknown effect, and a country code was introduced as a bias term, G_β, to capture variations between sets of data provided by the contributing nations. The data harmonization results demonstrated better agreement across national boundaries, all other parameters being equal, as well as increased prediction confidence.

Data harmonization has some commonality with cokriging, in that multiple sources of data are modeled using a Kriging-based approach. Data harmonization is set apart from cokriging by its introduction of bias variables, both known and unknown. A more mathematically-focused description will be given in Section 5.1.5.

3.2.4 Summary of Multi-fidelity Techniques

In general, researchers have found multi-fidelity methods to be useful techniques for reducing computational cost when high-fidelity predictions are desired. Simpler, faster analyses provide overall trends and general behavior, while slower but more accurate tools provide corrections. The utility of a multi-fidelity approach will vary somewhat with the problem. The methods are at their most effective when the lower-fidelity analysis is less expensive than the higher-fidelity analysis. This effect is enhanced if the simpler analysis can be easily reproduced via surrogate models. The degree of agreement between the two analyses is also a factor – the closer the agreement between the two levels of analysis, the more easily a corrective model can be trained. Lastly, the expense of problem formulation should be considered: effort may be required to ensure that both analyses are applied to the same problem, e.g. that vehicle geometry representations match as well as possible. Differences in the input and output data requirements between analyses may require extra preparation

effort.

The data harmonization technique of Baume et al.[14] appears to lend itself directly to the task at hand. Multiple computer models will be applied, but each will be estimating the same response instead of separate responses which are correlated. There is some cause for concern when applying data harmonization to problems with large design spaces; typically such problems require a large number of training points to fit accurately, and as a result the matrix inversion required to fit the model may become exorbitantly computationally expensive. If this proves to be the case, research into sparse methods may be of use.

As for other multi-fidelity techniques, true cokriging in the style described by geostatisticians[93] is beyond the scope of the current effort, as no implementation of cokriging could be identified that could incorporate more than three input dimensions.[145, 164] Other multi-fidelity methods will also be assessed to determine which of them is the most effective for the current problem. These methods were selected on the basis of conceptual simplicity and relative ease of implementation. Additive correction, proportional correction, and Ghoreyshi cokriging will all be assessed.

Unlike data harmonization, all three of the other techniques treat each source of data independently. A surrogate model is trained to match the lowest-fidelity data source, and this model is combined with higher-fidelity data to train a surrogate model for the next data source. These methods need not handle every data point from every source simultaneously, and as a result they will be much less vulnerable to the expense that comes with fitting Kriging models to very large sets of data. Despite this, if the quantity of data in use does grow large, sparse methods may be of use here as well. Sparse methods were therefore the next line of research to be conducted.

3.3 Sparse Methods

The simplest approach for limiting the computational effort required to fit a model would be to select a covariance function that decays to zero quickly, resulting in a very sparse covariance matrix if the data set is spread out throughout the design space.[128] This would produce a model that is heavily dependent on the underlying mean function, diverging from

this mean only in the very close neighborhood of the training data points. Such a decision would be risky without high confidence in the choice of the mean function. Alternatively, many different mean functions could be tested to identify the one or ones that best fit the data. This testing process would have to be repeated for each response being modeled.

The largest objection to this approach is that the region of influence given to a training point by a covariance model is defined by the model parameters, which are estimated from the data when the model is fit. If training points are found to correlate with sites relatively far away, optimal model parameters will reflect this behavior and the covariance matrix will not be sparse. Bounds on allowable model parameter values could be set to enforce this approach to sparsity, but such sparsity might come at the price of surrogate accuracy.

A different approach to minimizing computational effort through sparse methods is the "Subset of Data" (SoD) approach.[157] Rather than utilizing every sample that is available for training, the SoD approach trains a model using a subset of the available data. This reduces the scale of the problem by discarding information. Information loss can be minimized by the intelligent selection of the subset. The best subset of m points, chosen from the N points available, will be that which produces the most accurate surrogate model. The accuracy of a surrogate is tested by assessing its ability to correctly predict response values for the unused training samples. Selecting the best subset of samples can itself become computationally intensive. However, the effort to invert a matrix of dimension m can grow as quickly as $\mathrm{O}(m^3)$,[136] and since $m < N$, a moderate amount of effort may be spent on subset selection before this approach becomes more costly than the brute force approach.

A third technique found in the literature is that of using pseudo-inputs.[179] This technique is conceptually similar to the Subset of Data approach, in that m training points are used to build the model rather than the full N points of the available data pool. The critical distinction is that pseudo-input methods are not restricted to the points in the pool. Instead, the locations of the control points are considered to be extra model parameters that must be estimated. If m control points are desired with each point having dimension d (equal to the number of input parameters being modeled), the number of parameters to be estimated is increased by $m \times d$. As Snelson[179] points out, this may result in an intractable problem

if the dimensionality of the problem is large, and in response they suggest projecting the input space into a lower-dimensional space, asserting that significant dimensional reduction can often be achieved for real problems.

In general, when creating a surrogate model of a large data set, most strategies for reducing the cost of fitting the model emphasize using only the most informative points (or pseudo-points). A large portion of the available samples are therefore used only indirectly (during selection of the best subset or pseudo-set) or not at all. If the model used to produce those data points is expensive to run, this may be a very inefficient use of resources. In order to make the best use of available resources, it may be worthwhile to apply a more iterative sampling process that aims to identify only the most informative data points. With respect to the motivating problem of predicting RBS aerodynamics, if such an iterative sampling method could identify points with near-zero pitching moments it would dovetail nicely with the research objective of emphasizing cases with good performance.

3.4 Selection of Experiments

The second of the four approaches from the list on page 49 was the selection of samples to emphasize interesting regions of the design space. Many approaches exist for choosing interesting cases to analyze. The "best" case will depend on the user's objectives. If prior knowledge of response behavior is not available, it may be necessary to do an initial round of experiments to investigate the response characteristics. This is sometimes referred to as a "warm start". Options for selecting this warm start will be reviewed before true adaptive sampling techniques are discussed.

The process of *a priori* experiment selection, whether for a warm start or not, is typically known as the **Design of Experiments**, or DoE. Some classical designs, such as Full Factorial, sample the design space at regular intervals and explicitly capture the corners of the design space, where input variables take their minimum or maximum values.[33] Including all corner points in a training set for surrogate models eliminates the risk of extrapolation, which can improve confidence in surrogate predictions. As the size and complexity of a problem increase, however, response behavior in the interior of the design space may take

on a larger role in surrogate model uncertainty, and it may be more efficient to perform experiments which fill the space effectively rather than sampling the extremes. This was noted in Section 2.3 as part of the RBS effort with the Latin hypercube & I-Optimal samples acting as the interior and extreme samples, respectively.

Latin hypercube sampling is a common space-filling DoE.[126] Latin hypercubes uniformly sample each design variable, distributing points throughout the space. Unlike the Full Factorial DoE, Latin hypercubes are not guaranteed to sample the corners of the design space, which may lead to extrapolation. Latin hypercubes with more points will fill the space more densely, which will better resolve the space and reduce potential prediction errors due to extrapolation by the surrogate model. This sampling approach has the advantages of accommodating any number of samples for a given design space and resolving response behavior in the central regions of the space, but has the disadvantage of poor resolution near the edges of the space.

There is a trade-off between the number of "warm start" samples and the number of adaptive samples: a larger warm start will result in more information about response behavior, which leads to more accurate identification of interesting regions for later sampling. On the other hand, if the experimental budget is finite, a larger warm start means fewer adaptively-selected samples may be evaluated. The user will have more information, but less opportunity to apply it. Once some knowledge about the response behavior is obtained, adaptive sampling can begin.

3.4.1 Overview of Adaptive Sampling

Adaptive sampling is the process of choosing new samples based on prior observations. Once regions of interest have been identified, these regions are sampled more extensively to improve understanding of nearby response behavior. This process is then repeated until the response is considered to be understood sufficiently accurately, or until the experimental budget is consumed. Selection of the next sample point is based on some algorithm which evaluates points and identifies the one that is "most interesting" according to some criterion. The choice of selection criterion is what distinguishes different adaptive sampling approaches.

Many adaptive sampling approaches make use of Kriging models. Kriging models, as a subset of Gaussian Process models, allow the user to not only predict the response value at any point, but also estimate the uncertainty of that prediction without arduous calculations. This uncertainty typically goes to zero for deterministic (i.e., noiseless) training points and increases for points farther from the training data.

If the objective is to create a surrogate model with the highest possible prediction confidence, the simplest approach would be to place samples where the prediction uncertainty is largest. After this point is sampled, the response value there is known for certain and nearby predictions can be made with more confidence. This is an attempt to maximize the information gained per experiment.[110] Another approach would be to identify the point that best improves the average prediction confidence throughout the design space; this average confidence can be approximated by evaluating how the confidence would change at a large number of test points throughout the space.[18, 34] Kleinjen and van Beers[97] describe a variation on that metric, called Integrated Mean Squared Error, which multiplies the variance at each test point by some weighting function. However, the weighting function is left uniform in that work.

For some applications, it may be worthwhile to focus on particular regions of the design space rather than improving global knowledge. If uninteresting regions can be identified, they may be sampled sparsely so that more promising areas may be investigated more thoroughly. Once infeasible regions are identified, later samples can avoid those regions.[103] Alternatively, if the objective is to optimize a response value, adaptive sampling can be used to identify the sample point that would most improve knowledge of the response near certain values. Such sampling approaches offer improved model accuracy in the attractive regions while minimizing the cost to sample unattractive regions. The question then becomes how to identify these attractive regions.

3.4.2 Adaptive Sampling for Optimization

The most common goal for localized sampling is the optimization of a response, such as minimizing wing weight or maximizing lift-to-drag ratio. The simplest approach would be

to sample the point which is predicted to have the optimum response value, but this may have undesirable consequences: if samples are clustered too closely together, some surrogate modeling techniques such as Kriging may encounter numerical problems. To avoid this, the adaptive sampling algorithm can be designed to encourage a certain degree of design space exploration.

The most popular strategy that combines exploration with exploitation is the Expected Improvement function,[173] which can be interpreted as the likelihood that a given point will have a better response value than the current best observation. This likelihood is calculated for a candidate point based on the predicted response value and uncertainty of the prediction at that point. This prediction uncertainty is typically assumed to be zero at sampled points and to grow for points farther from samples. As a result, the EI function has incentive to select points in poorly-sampled regions. This approach is sometimes called Efficient Global Optimization, or EGO.

Cox and John[38] propose the Sequential Design for Optimization (SDO) method, which also utilizes the prediction and uncertainty at each point. This algorithm also calculates the prediction and confidence interval at each candidate point. The candidate with the best confidence bound is selected as the next sample point. This may be a point with a very desirable predicted value and small uncertainty, or a point with a moderate predicted value and large uncertainty. Xiong et al.[194] expanded this approach, varying the number of standard deviations used for the confidence interval to encourage the algorithm to explore regions with high uncertainty or to emphasize regions expected to have good response behavior.

Huang et al.[83] extend the EI function to allow sampling at different fidelity levels. The function is modified using three multiplicative terms which capture the reduced confidence associated with a lower-fidelity result, the benefit of repeated samplings for noisy functions, and the relative costs of sampling at each fidelity level.

3.4.3 Adaptive Sampling for Other Objectives

Although optimization of some response is the most common application of adaptive sampling, algorithms have been developed which sequentially choose samples based on other criteria as well. Farhang-Mehr and Azarm[55] define a characteristic certainty width (CCW) parameter, which is used to represent the regularity of the response behavior in a local region. Large values suggest the response changes slowly and simply over a broad region, while small values indicate large or rapid shifts in the response over a small distance. Large values thus indicate regions which may not require many samples to capture response behavior, while small values indicate regions where additional sampling may be beneficial.

Mackman and Allen[111] score candidates for sampling according to the predicted local nonlinearity of the response. Nonlinearity is quantified using the diagonal of the Hessian matrix, i.e. the Laplacian. To dissuade the algorithm from clustering points too closely together, a separation function was incorporated that grows with increasing distance from existing sample locations, improving the scores of points far from the current training set and reducing the scores of points too close to previous samples.

It is also possible that the objective might not be to identify points with maximum or minimum response values, but to find points where the response has a particular value. This may occur in reliability modeling, when the objective is to identify whether or not a case exceeds specified constraints; model performance is better served by accurately capturing this failure contour than by seeking the maximum response value.

Ranjan et al.[159] describe a sampling criterion to best improve knowledge about the response behavior near the contour of interest. This criterion includes multiple factors to balance the sampling style between sampling points predicted to fall near the contour and sampling points predicted to be farther from the contour but with enough uncertainty that the contour may be closer than expected. A clustering penalty term is used to dissuade the algorithm from placing samples in regions of low uncertainty, i.e. close to existing samples.

Picheny et al.[149] propose an alternative contour sampling algorithm based on the Integrated Mean Squared Error approach described by Kleijnen et al.[97] The prediction uncertainty at a multitude of test points is combined to quantify the global prediction

confidence as in Kleijnen et al., but in this formulation a non-uniform weighting function is applied. This weighting function is a measure of the likelihood that the response at the current test point is close to the contour of interest. Test points predicted to have responses close to the target will have their uncertainty more heavily weighted. The algorithm will then select the candidate point which most decreases the IMSE value, i.e. the candidate which most improves prediction confidence for points near the threshold.

3.4.4 Summary of Adaptive Sampling Techniques

Many sampling algorithms have been documented that promise to leverage current knowledge about the response(s) of interest when selecting the next experiment to be run. Depending on the user's goals, sampling algorithms exist to improve global model confidence, find global maxima or minima, identify local nonlinear behavior, or accurately capture response behavior near some desirable threshold.

Given the context of aircraft design and the concept of design-for-trim, the contour sampling approaches described by Ranjan et al. and Picheny et al. are the most promising. Contour estimation may be used to focus samples in the region of interest, i.e. configurations that are expected to experience small moments at likely flight conditions. In addition, both selected approaches incorporate factors which dissuade clustering of points. Clustered points are informative when the goal is to find an optimum response value, but when the goal is to fit an useful surrogate model throughout a design space, clustered points are likely redundant and potentially a waste of effort.

Note that the selected sampling techniques, as well as many others, incorporate prediction uncertainty in the evaluation criterion. This uncertainty is not a function of the accuracy of the data points, but rather stems from the estimation of model parameters. Each data point in the training set is considered to be a precise representation of the true response for the appropriate input values. When applying adaptive sampling to data at multiple levels of fidelity, knowledge of the likely accuracy of each data point can affect sampling behavior. For example, linear aerodynamics tools are moderately trustworthy at low angles of attack, but neglect the nonlinear effects that become important at higher angles of

attack. Both low and high angles of attack would have to be sampled repeatedly to identify this pattern, as it may not be possible to exactly quantify the discrepancy in advance.

Having addressed both adaptive sampling techniques and multi-fidelity methods, two of the topics of interest remain: quantification of uncertainty inherent in the data, and means of incorporating that knowledge when creating a surrogate model. These topics will both be addressed in the next section.

3.5 Quantifying & Incorporating Uncertainty

To understand how uncertainty can be quantified and addressed, the most obvious source of information is the way that it has been addressed in the past. Two case studies, the Space Shuttle and the X-33, were used to illustrate how vehicle design programs evaluated and accounted for uncertainty in the data. These case studies will provide an initial understanding of the subject.

3.5.1 Case Study: Space Shuttle

The Space Shuttle was the first large-scale effort to quantify and document pre-flight uncertainty with respect to vehicle performance measures. The extensive public-record documentation of that effort makes the Shuttle program an excellent resource. Unlike previous programs, the Shuttle did not pursue a development program based on a gradual expansion of the vehicle's envelope. Instead, the program leapt from low-speed glide tests to a manned orbital mission and reentry. This challenging program was driven by incentives to minimize testing costs and duration. Instead of flight testing, understanding of the Shuttle's performance relied on one of the most extensive wind tunnel testing regimens in history.[199]

In order to prepare for the initial orbital mission and subsequent reentry, NASA needed to quantify the uncertainty in the aerodynamic data base for the vehicle. Aerodynamic uncertainty, coupled with the proposed vehicle flight control system, could be used to identify conditions at which the vehicle had minimal or negative control margin. When aerodynamic uncertainty resulted in a risk of compromised mission performance, the design team had the choice of either requesting additional ground testing (to reduce prediction uncertainty) or adjusting the reentry trajectory to avoid the troubling flight conditions. Through use of this

process, Shuttle engineers were able to increase confidence in mission performance.

The Shuttle aerodynamic uncertainty was quantified in a number of ways. The first and most direct was the repetition of certain tests using multiple facilities, models, and sets of instrumentation. Such repetition helped the analysts to estimate the variation in responses that was related to the testing equipment and procedures rather than true vehicle behavior. Those variations were between performance results at similar levels of fidelity, and were called **tolerances**. The Shuttle had to be able to cope with prediction errors of this scale without affecting overall vehicle performance.[198]

In addition to determining tolerances, Shuttle aerodynamicists had to estimate the magnitude of the discrepancies between wind tunnel predictions and in-flight behavior; those discrepancies referred to as **variances**. Variances would be known with confidence after flight testing, but early approximations were required to identify risky conditions and plan accordingly.[198] These approximations were drawn from previous flight test experience available in the public literature. No single vehicle matched the Shuttle's configuration or wide range of flight regimes. Instead, estimates of variances were drawn from the published pre-flight and post-flight aerodynamic comparisons for a number of vehicles, selected to match particular aspects of the Shuttle such as a delta wing planform, the use of wing flaps for longitudinal control, the use of elevons for lateral control, a single vertical tail, and a relatively large fuselage.[59] No one vehicle satisfied all similarity characteristics.

Vehicles were chosen based on the degree of similarity and the availability of both pre- and post-flight performance data. These vehicles included commercial aircraft (Concorde), military service aircraft and prototypes (B–58, XB–70, YF–12), and research vehicles (X–15, X–24B, M2–F3, HL–10). Although the X–15 was not thought to share many geometric characteristics with the Orbiter, it was included as a reference due to the paucity of hypersonic data.[190] The comparisons between the aerodynamic performance estimated before flight and the data collected during flight testing and operation of the vehicles provided insight into the likely accuracy of the Shuttle pre-flight database.

Weil[190] uses the M2–F3 lifting body to illustrate the process of estimating variations. At Mach 1.1, the predicted values of C_{n_β} were compared over a range of angles of attack.

The maximum deviation between predicted and flight-measured values was identified after any clear outliers are removed. A trend in deviation behavior was discernible, growing larger as angle of attack increased. Weil cautions that "care must be exercised to limit this variation in regions of rapidly changing characteristics" but does not go into detail how this might be accomplished. This process was repeated for other Mach numbers and for other vehicles. The resulting discrepancies were grouped by Mach number only – the effects of angle of attack or sideslip were aggregated – and then uncertainty limits were defined based on engineering judgment, rather than statistical analysis.

Expert opinion played a large role in the development of the variances: it directed the selection of characteristics that would define useful reference vehicles, it directed decisions as to whether individual vehicles were sufficient similar to serve as references, and it was the foundation for the final variance values in the uncertainty model.

Test flights of the Shuttle demonstrated that even conservative variance estimates did not always encompass the true vehicle performance. After the first flight test, it was found that hypersonic trim at high angle of attack required a significantly larger deflection of the body flap than was planned. Although the discrepancy in pitching moment coefficient was only approximately 0.03, correcting the discrepancy required 16° of deflection rather than 7°, leaving less than one-third of the expected control margin for maneuvers or controlling dispersions. Later testing suggested that real gas effects, which were not extensively simulated pre-flight, were responsible for the majority of the discrepancy.[86, 133] This supports the expectation that gaps in the set of reference vehicles – in this case, a lack of lifting configurations with data at double-digit Mach numbers – may result in an insufficient understanding of likely variances.

3.5.2 Case Study: X–33

A similar approach to variance estimation was used in the modeling of aerodynamic uncertainty of the X–33 vehicle, a proposed single-stage-to-orbit demonstrator.[32] The Shuttle, along with six lifting-body vehicles, were selected as reference vehicles due to similarity of geometry and/or flight regimes. All selected vehicles also offered sufficient documentation of

the comparisons between pre-flight aerodynamic predictions and flight test data. The X–33 aerodynamic uncertainty model focused on variances rather than tolerances, and effort was taken to make the model applicable to any lifting body vehicle. Note that the estimated uncertainty is again a one-dimensional function: the uncertainty associated with other parameters, such as angle of attack, is aggregated into a quantity that is purely a function of Mach number.

Despite the quantity of data available, expert opinion once again played a large role in the final X–33 uncertainty model. First, expert opinion once again drove the process of selecting vehicles to serve as references. In addition to having publicly documented pre- and post-flight aerodynamic databases, the chosen vehicles were believed to experience flow behaviors similar to what the X–33 would encounter, such as nonlinear flow and vortex shedding. Expert knowledge was required to identify flow behaviors that will be relevant to the configuration, as well as selecting existing vehicles which were subjected to those flow phenomena.

Secondly, decisions were made to incorporate some data points from the reference programs, modify others, and exclude some entirely. Because the Shuttle had the benefit of an extensive wind tunnel testing regimen, its pre-flight predictions had less uncertainty than would be likely for a smaller program. To account for this, parts of the X–33 uncertainty envelope were enlarged relative to Shuttle variances. Similarly, the X–33 uncertainty envelope was made to encompass some of the lifting body data points while excluding others, signifying a decision that the excluded data points were in some way inaccurate or unimportant. Both the choice of values to be modified and the degree to which they were modified were decisions made by X–33 aerodynamicists.

3.5.3 General Approach: Estimation of Uncertainty

The uncertainty quantification described in the case studies were based on two considerations. First, the *tolerances* were the estimated uncertainty at a particular level of fidelity – based on CFD results, how accurately can other CFD results be predicted? Secondly, the *variances* were the potential discrepancies between different levels of fidelity – based on

CFD results, how accurately can flight performance be predicted?

For this application, the estimation of aerodynamic tolerances would depend on a number of data sources. First, data that is generated by an iterative solver can be interrogated to determine solution convergence. For this particular problem, the iteration history from Cart3D may be used as a measure of the variation in each response. Rather than defining a response as the value of a certain coefficient after the final iteration, it may be more effective to observe the mean value over some number of iterations, and use the standard deviation over those iterations as a measure of the noise for that case. The scripts currently in use to run Cart3D could be easily modified in this manner. This would provide an individual estimate of uncertainty for each Cart3D data point.

Secondly, for the three multi-fidelity techniques which handle each source of data separately, the low-fidelity data itself will be a source of uncertainty. For this effort, the low-fidelity data will be produced by APAS. APAS runs fairly rapidly, on the order of 1–2 seconds per case, but even that could add hours to the time required to select a sample if many options are considered. Instead, a surrogate model of APAS results will be generated so that thousands of low-fidelity response values can be estimated in the time it would take to run APAS once.

This additional surrogate model, unless it is a perfect representation, will only approximate the low-fidelity results. The **goodness-of-fit** checks which quantify a surrogate model's performance include Model Representation Error (MRE), a measure of how well the surrogate predicts response values for cases that were not part of the training process.[36] MRE can be considered an estimate of how accurate the surrogate will be when used to predict response values at points where the true analysis has not been run. It is typically cited as a mean and a standard deviation of prediction error. The standard deviation can be squared to produce the error variance, which can then be treated as the prediction variance of the low-fidelity surrogate model in the absence of more precise estimates. These two sources of data – the iteration noise from Cart3D analysis and the prediction uncertainty from the surrogate model of APAS results – shall make up the aerodynamic tolerances for the purposes of these experiments.

The process for estimating aerodynamic *variances* – the uncertainty introduced when using results at one level of fidelity to estimate response behavior at some other level of fidelity – is more complex. For both the Space Shuttle and X–33, expert opinion was used to select vehicles which could offer useful comparisons. The Shuttle is an especially good example of the potential difficulties. None of the reference vehicle satisfied all five of the similarity criteria. Additionally, some reference vehicles only had data available for a limited set of flight conditions. This was not unexpected; the Shuttle was significant departure from what had been done before, and thus a comprehensive suite of reference vehicles was not likely to exist.

Although the X–33 uncertainty database was intended to be applied to both wind tunnel and computational model results, it did not specify the type of computational model that is expected. This may be considered a shortcoming, as the prediction accuracy of computational models will vary greatly with model fidelity. This type of knowledge would be useful when designing the experimental plan: if it is known that a simple model will have sufficient prediction accuracy at a particular flight condition, the designer will not need to apply more expensive models for confirmation. Such an approach would rest on the ability to accurately and dependably quantify the accuracy of a given model at various flight conditions.

Recall again from Section 1.6 the calls for treating model validation as an ongoing process, one that is repeated and extended as necessary each time the too is applied to a different application.[8, 138] If each computational model is validated over the relevant ranges of flight conditions, the user will have obtained a pool of data which can be used to estimate each model's likely accuracy across the proposed vehicle's flight regime. This pool of data quantifies the accuracy for each validation case; carefully chosen validation cases will therefore allow the estimation of variances for the configuration of interest.

Note that the Shuttle and X–33 uncertainty databases modeled the variance as a single value per Mach number. This reflects the assumption that aerodynamic data would come from wind tunnel testing and possibly computer modeling at an equivalent level of fidelity. Early in the design process, aerodynamic data may have many sources due to the variety of analysis tools available. Each source may have different prediction confidence at each

flight condition of interest. If each data source is validated against relevant cases, the likely accuracy of each data point may be estimated.

For this particular effort, variances would be less important. The scope of the research did not offer the opportunity to acquire flight test data for a reusable booster system, so CFD analyses would be the highest-fidelity data source available. As a result, it was not considered necessary to estimate how well the CFD results would match physical measurements. Instead, the CFD results would act as "truth" data for these experiments. Furthermore, it was already established that the discrepancies between the lower-fidelity APAS results and the CFD results would be accounted for as part of the tolerances. Thus, variances would not play a significant role in this effort, although they are certainty important to take into account when designing a revolutionary vehicle. Still, variances were investigated to a reasonable degree during the course of this research.

The uncertainty databases as described were intended for use in low-risk trajectory design and the testing of flight control software. If such information were available earlier in the design process, it would reduce the risk of selecting a deficient design by quantifying the confidence of performance predictions. Knowledge of the prediction confidence at each flight condition would also increase computational efficiency by focusing the use of expensive simulations on cases which most benefit from the increased prediction confidence.

3.5.4 Incorporating Uncertainty

The final topic to be addressed is the incorporation of the uncertainty information into the analysis process. Given that the adaptive sampling methods which are best-suited to the problem at hand are Kriging-based, it would seem reasonable to focus the search on techniques to incorporate uncertainty into Kriging models. Recall from Section 3.4.1 that prediction variance goes to zero at the training points if those points are deterministic.

If the training points are not deterministic, i.e. there is some uncertainty as to the exact value of the response at each point, a "**nugget**" parameter may be used to quantify the response uncertainty at that point.[93] The nugget parameter is a scalar or vector of values that are added to the diagonal of the covariance matrix when fitting a Kriging model; the

nugget magnitude controls how closely the Kriging model will reproduce the training data, with larger nuggets corresponding to looser fits of the data. Nuggets were included in some of the earliest Kriging formulations,[119] and were intended to capture measurement error and small-scale variations in the response as part of geostatistical modeling.[93] Even when modeling the output of deterministic (i.e. noiseless) computational tools, some researchers advocate the addition of small nugget terms to covariance matrices to improve numerical stability and reduce the risk of a poorly conditioned covariance matrix.[67, 107, 160, 197]

One consequence of adding a nugget to the covariance matrix is that the resulting Kriging model will no longer be an exact interpolator: it will not exactly reproduce the training data set when used to predict the response at a known point.[39] Very small nuggets, on or close to the order of machine precision, will not affect the interpolation behavior significantly.[107] Gramacy and Lee also argue that nuggets improve the statistical properties of the emulator in cases of sparse data or when modeling assumptions such as the correlation type are incorrect.[68] The larger the nugget values, the more closely the model will follow the estimated mean function.[182]

Overall, the ability of a vector of nuggets to capture any uncertainty that varies between samples, as well as the compatibility of nuggets with the Kriging formulations that support many adaptive sampling approaches in the literature, made the use of nuggets very attractive as a means of incorporating uncertainty into the surrogate modeling process.

3.6 Review of Research Questions & Formulation of Hypotheses

As a result of the literature search that was driven by the research questions, multiple techniques have been identified that may improve designers' ability to apply high-fidelity modeling earlier in the preliminary design phase. The major factor which stymied earlier attempts was the significant increase in modeling expense when more complex data sources are used. A research effort was undertaken to identify alternative techniques and methods which would help to reduce this expense and bring such modeling into the realm of feasibility.

The first focused research question was:

Research Question: **When "good performance" refers to responses within desirable ranges rather than maxima or minima, how can regions of good performance be identified and emphasized during the sampling process?**

In light of the described adaptive sampling approaches, the contour-based sampling approaches described by Picheny et al.[149] and Ranjan et al.[159] should allow the preferential selection of cases with near-zero moments. This is the basis for the first hypothesis:

Hypothesis 1 ***Contour-based sampling will balance the selection of cases with good performance and the reduction of prediction uncertainty in promising regions, identifying samples that efficiently improve surrogate accuracy for configurations with small aerodynamic moments.***

The savings in computational effort from contour-based sampling alone is not expected to reduce modeling costs sufficiently to enable large scale application of expensive models. For further reduction in cost, low-fidelity data will be used as a source of cheaper data, providing estimates for the responses of interest that will be corrected by more accurate models.

The next focused research question spurred the investigation of techniques that would allow the low-fidelity estimates to be blended with data from the more trusted models:

Research Question: **How can cheaper analyses be integrated with high-fidelity models to reduce the overall cost of design space exploration or exploitation?**

After a review of the available techniques, a number of possible methods were identified that might improve predictive performance and reduce dependence on expensive data sources. Multi-fidelity modeling is abundant in the literature, but it remains to be seen which of the methods identified, if any, will be effective for the problem at hand. Rather than selecting one now, the choice will be deferred until comparisons between the methods can be made for one or more representative problems. Thus, the second supporting

hypothesis is couched in more general terms:

Hypothesis 2 ***Data fusion techniques will allow results from high-fidelity analyses to be augmented with cheaper sources of data to produce surrogate models that are more accurate yet require less computationally-expensive data.***

The last focused research question highlighted uncertainty in the data and sought to determine how that knowledge might be incorporated:

Research Question: How can information about uncertainty in the data be captured effectively?

It was noted that one of the multi-fidelity techniques that had been identified, data harmonization, serendipitously also included an explicit mechanism to incorporate uncertainty: the Kriging nugget. If Kriging models are used to implement the other data fusion techniques as well, nuggets can be used to extend each of those methods in turn so that each one can integrate data from multiple sources *and* capture any uncertainty in those data points. This leads to the next hypothesis:

Hypothesis 3 ***When creating a Kriging model, the use of nuggets will capture uncertainty in the data, improving predictive accuracy for noisy responses.***

Supporting this goal are the various techniques which may be used to quantify the uncertainty in data points. These techniques include interrogation of response history for iterative models, the use of validation experiments to assess expected model accuracy, and targeted validation experiments to quantify the likely discrepancies between different sources of data.

Together, the selected techniques should serve to produce more accurate surrogate models while reducing the cost necessary to generate the required data, addressing the primary research question:

Research Question: How can high-fidelity modeling be feasibly applied earlier in the design process, despite the computational expense?

The three previous hypotheses come together to address this, the overarching goal of the research, with a final hypothesis:

Hypothesis 4 ***By placing samples intelligently, reducing dependence on the expensive models, and accounting for any uncertainty in the data, the selected methods will enable improved surrogate model accuracy with significantly reduced data requirements, such that high-fidelity modeling becomes a feasible option earlier in the design process.***

This hypothesis represents a proposed procedure for sample selection and surrogate model creation. The first three hypotheses will provide guidance as how certain steps of the process might best be carried out. The final hypothesis asserts that, by approaching each step of the method with appropriate techniques, the result – surrogate models for cases of interest – will have quantifiably better predictive accuracy than what is currently possible with standard techniques. In essence, the final hypothesis corresponds to a proposed approach to the problem.

The approach being proposed is intended for problems for which standard sampling and surrogate modeling techniques would result in excessive costs, particularly in the form of execution time and/or computational effort. In particular, the proposed approach will improve efficiency by maximizing the information gained from each expensive analysis while minimizing the number of such analyses that will be required.

3.6.1 Steps in the Method

The method being proposed will assume that the user has already set up the problem. That process entails a number of activities, such as identifying the independent and dependent variables that will be included, selecting appropriate data sources, and validating each data

source against "truth data" to quantify its expected accuracy. This description will also assume that two data sources will be used, one being more accurate but having much higher per-analysis costs, and the other being less costly but lacking the necessary accuracy for use as the only data source. Depending on this cheaper data source's speed of execution, the user may wish to replace it with a surrogate model. Given these stipulations, a general description of the procedure for creating surrogate models is as follows:

Step 1: Generate an initial set of samples to be analyzed.

Prior knowledge about the behavior of the responses to be modeled may indicate that certain regions of the response space are of greater interest than others. The initial samples may then be distributed to emphasize those regions. Barring such prior knowledge, space-filling sample distributions such as Latin hypercubes or Sobol sequences are often appropriate.

Step 2: Analyze the samples using the appropriate data sources.

Step 3: Train Kriging surrogate models using the resulting data.

It should be noted that the effectiveness of a data fusion technique will depend on the problem being addressed. Some of the experimental effort in this research will be dedicated to identifying which of the selected data fusion techniques is best suited for the intended application of modeling reusable booster aerodynamics.

Step 4: Evaluate the resulting surrogate models to quantify the predictive accuracy.

There are two primary ways to quantify the predictive accuracy of a surrogate model. The first method is to set aside a number of cases.[88] These cases are *not* used to train surrogate models. Instead, the surrogate models are used to predict the response values for those cases, and then the predicted values are compared against the observed values. The discrepancy between the predicted and actual values is then used to assess the prediction error.

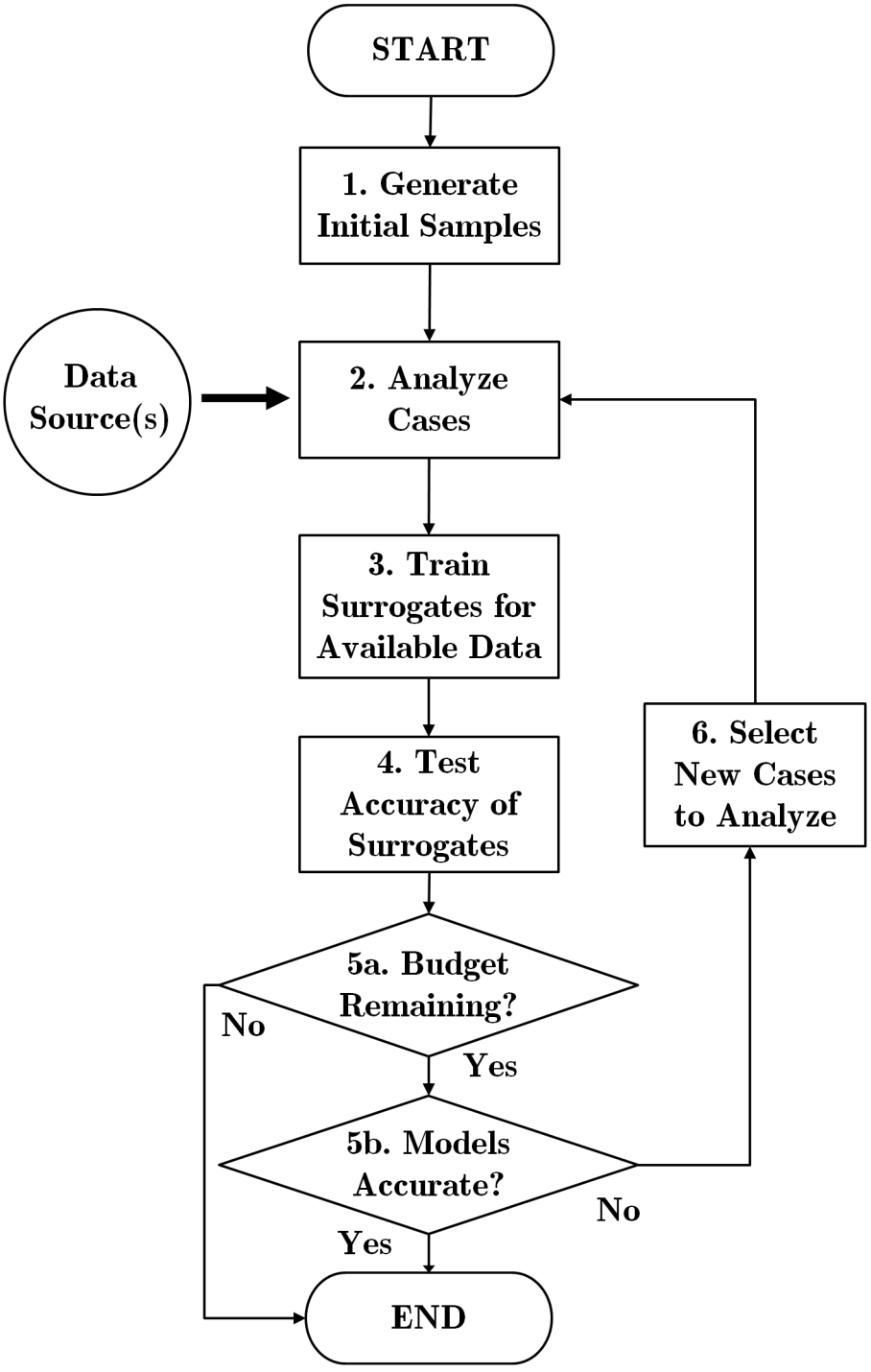

Figure 8: Baseline Process for Sample Selection & Surrogate Model Creation

The other alternative is to use cross validation.[148] When performing cross validation, all of the available data is used to create the main surrogate models. The training data is then split up into subsets and a number of temporary surrogates are trained, each of which is trained using all but one of the subsets. Each temporary surrogate is then used to predict the response values for the subset of data that was not used in its training. Again, the discrepancy between the predicted and actual values is used to assess prediction error. Depending on the amount of data available and the level of effort required to train a surrogate model, the subsets may be as large as 20% of the data set or as small as one data point.[58, 75, 100] This approach has the benefit of using *all* available data when training the final surrogate models, but may under-estimate the predictive accuracy of those surrogates.

Steps 5a & 5b: If the surrogate models are sufficiently accurate or the project resources have been consumed, terminate the process.

Step 6: Otherwise, select new samples for analysis and go to Step 2.

The process of selecting new samples can be thought of as an optimization: the samples that are selected will be those that are the most useful. The exact definition of "most useful," and thus the behavior of the sample-selection algorithm, will depend on the choice of objective function. Hypothesis 1 asserts that contour-based sampling will be effective for the problem at hand; this hypothesis must be tested before being accepted.

For the purposes of this work, it will be assumed that the cost of obtaining the low-fidelity response at any point is negligible. The sample selection process will therefore only be concerned with the selection of high-fidelity samples. If the per-analysis cost of the low-fidelity data source is *not* negligible, it is recommended that the low-fidelity data source be replaced with a separate surrogate model, which would allow the low-fidelity response to be predicted without the associated analysis costs. If that is impossible or too labor-intensive, the reader is referred to the work of Huang et al.,[83] who describe an adaptive sampling approach that also selects the best data source to use for each new sample based on the relative costs and benefits of each data source.

With respect to the overall focus of this research – i.e., the creation of surrogate models & the selection of subsequent samples to improve surrogate accuracy – the generic process is laid out graphically in Figure 8. The first three hypotheses of this research effort pertain to how specific steps in the process are performed. Hypothesis 1 addresses Step 6, asserting that a particular adaptive sampling approach will yield better results than spending the full set of resources on space-filling samples. Hypotheses 2 and 3 address Step 4, emphasizing how the data (including validation results for each data source) will be used to create surrogates. The final and most important hypothesis, Hypothesis 4, asserts that when the process is carried out using the recommended techniques, the performance of the resulting surrogate models will be better than the surrogates produced by the baseline approach of single-fidelity modeling and space-filling samples.

Although each hypothesis was constructed based on known research and evidence, further experiments must be performed to determine whether the evidence supports or undermines these hypotheses. Experimental results which support the first three hypotheses, due to the hierarchical style of construction, also support the fourth hypothesis. The experimental plan was designed to test each hypothesis in turn.

CHAPTER IV

EVALUATING CONTOUR-BASED SAMPLING

The expense of each Cart3D analysis is not negligible, so efforts were made to maximize the re-use of results. To this end, experiments were designed so that the cases for one experiment would be useful for another. This is particularly true of the space-filling sample designs that exemplify the baseline approach. Points selected by adaptive sampling were expected to vary from experiment to experiment, with little chance of commonality. Examples of this will be highlighted in the description of the experiments as appropriate.

The first hypothesis to be evaluated asserts that contour-based sampling would allow accurate surrogate models to be trained using fewer samples. The null hypothesis in this case took the form of a space-filling sampling design. Using the results from the RBS study, three flight conditions were selected as having unusually poor fits for pitching moment relative to the other models. These conditions were: Mach 0.3, α 15°, β 0°; Mach 0.8, α 0°, β 0°; and Mach 2.5, α 0°, β 0°. This trio of conditions posed the most difficulty for the space-filling sampling approach, and thus these models were most in need of improvement.

The overall objective is the generation of surrogate models with good predictive accuracy for vehicles with desirable performance, i.e. small aerodynamic moments. If only one flight condition were being considered, the problem would be simple, as only three responses must be addressed. For a more general problem, it is possible that each of the three responses might exhibit very different behavior: a vehicle with near-zero pitching moment at a small angle of attack might exhibit a large pitching moment at larger angles of attack. However, the problem at hand only included cases with no sideslip, and so the two lateral responses were expected to be close to zero. It was therefore expected that a concept's ability to trim for these conditions would be dominated by its pitching moment, and so the primary focus of the experiment was to demonstrate accurate & efficient modeling of vehicle pitching moment.

4.1 Conceptual Description of Sampling Algorithm

The sampling algorithm was derived from the one proposed by Picheny et al.[149] in 2010. That algorithm was intended for problems with one response. As part of this research effort, the method was expanded to handle multiple responses. The expanded algorithm is as follows:

1. Create Kriging models of the current data points and the response values at those points.
2. Generate a set of candidate points.
3. Generate a set of test points.
4. Use the current Kriging model to estimate the response values and the prediction variances for the candidate and test points.
5. For each candidate point, calculate how adding that point to the training data set would change the Kriging prediction variance at each test point. Combine the resulting test point prediction variances in a weighted sum. The value of the weighting function will vary based on the likelihood that the test point has a response near the threshold of interest.
6. Repeat step 5 for each response.
7. Normalize the weighted sums for each response. The candidate point's "score" is the average of its normalized weighted sum results.
8. Select the candidate with the lowest score as the next case to be sampled.

Steps 1–8 are repeated until the Kriging model is sufficiently accurate or all resources have been consumed. The specifics of this algorithm rely on the mathematical details of the Kriging technique, which shall be reviewed briefly. More detailed information may be found in Journel & Huijbregts,[93] Sacks et al.[170], or Lophaven et al.[108]

4.2 Review of Kriging Mathematics

This work was based on the form of Kriging known as "universal Kriging," which models as a combination of a set of basis functions and a zero-mean Gaussian process. The basis functions describe the overall behavior of the response in a manner similar to response surface equations,[134] while the zero-mean Gaussian process captures any departures from the large-scale behavior captured by the basis functions. The best linear unbiased predictor for the response $y(x)$ at some unsampled point x, given a set of n other observations Y defined by p input parameters, can be calculated as:

$$m_n(x) = f(x)^T\hat{\beta} + c(x)^T C^{-1}\left(Y - F\hat{\beta}\right) \tag{1}$$

Here, $f(x)$ is a $(p+1)\times 1$ vector of basis functions, $\hat{\beta}$ is a $(p+1)\times 1$ vector of estimated coefficients, $c(x)$ is an $n\times 1$ vector of covariance, C is an $n\times n$ covariance matrix, and F is an $n\times(p+1)$ experimental matrix. The vector of estimated coefficients $\hat{\beta}$ is calculated by a generalized least-squares estimate:

$$\hat{\beta} = \left(F^T C^{-1} F\right)^{-1} F^T C^{-1} Y \tag{2}$$

The second set of terms to the right of the equality in Equation 1 is a function which estimates how much deviation from the underlying behavior can be expected at a particular point. This deviation is calculated based on the estimated correlation between the point in question and nearby "known" data points. A number of different correlation functions may be used as part of Kriging models, with the Gaussian function seemingly the most common selection.[107, 170] The Gaussian correlation function is given as:

$$k(u,v) = \prod_{i=1}^{n} exp\left[-\left(\frac{|u_i - v_i|}{\theta_i}\right)^2\right] \tag{3}$$

Here, i refers to the dimension of the vectors u and v. If distance in each dimension is weighted equally, every θ_i has the same value and this becomes the isotropic Gaussian correlation function. The more general case of an anisotropic function, where each dimension is weighted independently, will be assumed for the following work.

Using the correlation function, the Kriging prediction variance at any point x may be calculated as

$$\begin{aligned} s_n^2(x) = &\quad k\left(x,x\right) \quad - c(x)^T C^{-1} c(x) \\ &+ \left[\begin{array}{cc} \left(f(x)^T - c(x)C^{-1}F\right) & \left(F^T C^{-1} F\right)^{-1} \left(f(x)^T - c(x)^T C^{-1} F\right)^T \end{array} \right] \end{aligned} \tag{4}$$

In this equation, $k\left(x,x\right)$ is σ^2, the process variance; $c(x)$ is a vector of covariances between the point x and the points used to build the model; C^{-1} is the inverse of the matrix of covariances between points used to build the model; $f(x)$ is a vector of basis functions describing the point x; and F is the experimental matrix of basis functions which describe the points used to build the model. The covariance function is equal to the correlation function multiplied by the process variance. Basis functions refer to the variables chosen to describe the relevant model factors. For example, for a typical two-dimensional linear model, the basis functions would be 1, x_1, and x_2 to capture a constant mean and the linear effect due to each input dimension.

The process variance is a measure of how well the underlying trend, $f(x)^T\hat{\beta}$, accounts for the observed data. When the trend is a good representation of the response behavior, the process variance will be small because the observed responses are close to the expected trend. Once the trend coefficients $\hat{\beta}$ have been calculated, the process variance can be estimated using the observed data points[117]:

$$\sigma^2 = \frac{1}{n}\left(Y - F\hat{\beta}\right)^T R^{-1}\left(Y - F\hat{\beta}\right) \tag{5}$$

Note that the second set of terms to the right of the equality in Equation 4 captures the degree to which any nearby "known" data points decrease the prediction variance at the desired point x. This term will be maximized (and prediction variance minimized) when the covariance vector $c(x)$ is maximized. Looking back at the Gaussian correlation function in Equation 3, $c(x)$ will be large when the point x is very close to the known points used to build the model. Essentially, Kriging prediction variance will be smallest when x is close to those used to build the model and larger for x farther away. This feature of Kriging forms the foundation of the contour-based sampling algorithm.

4.3 Mathematical Formulation of Sampling Algorithm

The sampling algorithm seeks to identify the point that will most reduce the prediction variance of the Kriging model in regions where the response is near some value of interest. For the planned application of this method, the value of interest is a pitching moment coefficient of zero and with a range of interest of $\pm$ 0.1. Configurations outside this range are not expected to be able to trim, and are thus likely to be infeasible from a vehicle control standpoint.

The algorithm proceeds in the following manner: *first*, a Kriging model is fit to each response of interest. There will be at least one response of interest (pitching moment coefficient) for every flight condition that will be evaluated. The *second* and *third steps* are to generate a set of candidate points and a set of test points. These points may be generated using a space-filling method such as a Latin hypercube,[126] or may be distributed according to the preferences of the user. The space-filling distribution, being the more general approach, was assumed throughout this work. One of the candidate points must be chosen as the next point to be sampled. As the number of candidate points increases, the algorithm will have more options to choose from; as the number of test points increases, the algorithm can more accurately assess each candidate. Enlarging either set comes at a cost of increased analysis time.

The *fifth step* is to calculate the change in prediction variance due to sampling a particular candidate point using a technique known as *weighted integrated mean squared error*, or wIMSE.[97, 149] This technique leverages the ability of a Kriging model to estimate the expected response value and prediction variance at each test point. Once the fifth step has been repeated for each response of interest, the *final step* is to normalize the scores of all candidates for every response and select the candidate which promises the best average improvement over all responses. This fifth step is clearly the heart of the algorithm, and must be explained in more detail.

4.3.1 Quantifying How Candidate Points Affect Prediction Variance

First, the n data points making up the original model are temporarily augmented with one of the candidate points. The $n \times n$ symmetric covariance matrix C is thus expanded to size $(n+1) \times (n+1)$ to capture the covariance of this candidate point with each of the n existing samples.

Recall from Equation 4 that the Kriging prediction variance depends on the inverse of the covariance matrix. Inverting the full covariance matrix once for each candidate point is possible but inefficient: because only the $n+1^{th}$ row and column of C are changing, most of the matrix to be inverted will remain constant no matter which candidate is considered:

$$C_{n+1} = \begin{bmatrix} \sigma^2 & c_{new}^T \\ \\ c_{new} & C_n \end{bmatrix} \tag{6}$$

Picheny et al. recommend that the inverse of the enlarged matrix be calculated using Schur's complement formula,[201] resulting in the following equation:

$$C_{n+1}^{-1} = \begin{bmatrix} 1 & 0 \\ \\ -C_n^{-1}c_{new} & I_n \end{bmatrix} \begin{bmatrix} \frac{1}{\sigma^2 - c_{new}^T C_n^{-1} c_{new}} & 0 \\ \\ 0 & C_n^{-1} \end{bmatrix} \begin{bmatrix} 1 & -c_{new}^T C_n^{-1} \\ \\ 0 & I_n \end{bmatrix} \tag{7}$$

Because C_n^{-1} can be computed once and re-used for every candidate evaluation, this approach replaces a matrix inversion operation with a series of matrix multiplication operations, reducing the computational cost significantly. To quantify the cost reduction, notional data was generated. This data had forty-nine input dimensions, similar to the problem of interest. Using progressively larger sets of notional data, the contour-based sampling algorithm was used to evaluate 10 candidates using 300 test points. For a large parameter space such as this, many more test points would be required to accurately evaluate a given candidate; given that this test was purely to assess numerical speed, it was sufficient that the number of candidate and test points be consistent.

The first algorithm used Schur's complement, as recommended in Picheny et al., while the second algorithm explicitly fit a new Kriging model every time a new candidate was

evaluated. The time to evaluate all candidates and select the best option was recorded. This was repeated 10 times for each data set. The average time required for the two methods to select a sample is given in Table 2. For models of the size and complexity expected in this effort, the use of Schur's complement was found to reduce the sample selection time by approximately 90%, or a full order of magnitude.

Table 2: Average Sample Selection Speed With & Without Schur's Complement (in seconds)

	200 Cases	400 Cases	600 Cases	800 Cases	1,000 Cases
Using Schur's Complement	22	66	133	229	342
Direct Matrix Inversion	196	674	1,418	2,405	3,592
Ratio	9.03	10.3	10.7	10.5	10.5

Using this approach, the inverse covariance matrix C_{n+1}^{-1} can be obtained just as if the Kriging model had been re-trained to include the candidate point. The covariance vector $c_{n+1}(x)$ for a test point x captures the covariance of x with each of the samples as well as with the current candidate point.

The final step in assessing the effect of a candidate on the surrogate is to append the basis function representation of the candidate point to the experimental matrix F. Selection of basis functions is left up to the user. This effort used a linear model, with each input parameter forming one basis function and an extra column to account for the mean value of the response. For a problem with normalized input parameters x_1 and x_2, the linear basis function values for sample k would be $[1\ x_1(k)\ x_2(k)]$.

The new C_{n+1}^{-1} and F_{n+1} matrices are then plugged into Equation 4, which in turn allows the calculation of prediction variance as if the current candidate point had been sampled and added to the model. This updated variance equation is then used to calculate the prediction variance at each test point.

Before these variances are combined, however, a weighting factor is applied. This weighting factor is large for test points with responses close to the target value and small for test points with responses far from the target value. Thus – and this is the heart of the algorithm – the candidate with the smallest weighted sum of variances (i.e. prediction uncertainty) can be considered the one that most reduces prediction variance for cases with responses close to the target value.

4.3.2 Weighting Function Calculations

Picheny et al. suggest two alternatives for the weighting function, one using an indicator function and one using a Gaussian density function. Because the problem at hand features a response region of interest with clearly-defined bounds (e.g., 0 ± 0.1), the indicator function was selected. This weighting function is calculated via:

$$W(x) = \Phi\left(\frac{T + \epsilon - m_n(x)}{s_n(x)}\right) - \Phi\left(\frac{T - \epsilon - m_n(x)}{s_n(x)}\right) \tag{8}$$

Here, Φ is the **cumulative distribution function** (CDF) of the standard normal distribution; T is the target response value (0); ϵ is the half-width of the range of interest (0.1); $m_n(x)$ is the predicted response value at point x; and $s_n(x)$ is the prediction variance at point x. This prediction variance is calculated using only the n known samples; candidate points are not included. In effect, this weighting function is equal to the probability that the response at point x has a value between $T - \epsilon$ and $T + \epsilon$.

Because Kriging prediction variance is assumed to have a Gaussian distribution, the CDF may be calculated analytically. Zelen & Severo provide a method for calculating the CDF for $x > 0$:[1]

$$\Phi(x) = 1 - \phi(Z)\left(b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 + b_5 t^5\right) + \epsilon(Z), \quad t = \frac{1}{1 + b_0 Z} \tag{9}$$

Here, $\phi(Z)$ is the probability density function (PDF) of the standard normal distribution, and $b_0 = 0.2316419$, $b_1 = 0.319381530$, $b_2 = -0.356563782$, $b_3 = 1.781477937$, $b_4 = -1.821255978$, and $b_5 = 1.330274429$. $\epsilon(x)$ is the discrepancy between this approximation and the true CDF value, with $\|\epsilon(x)\| < 7.5 \times 10^{-8}$. The PDF of a normal distribution

is given as:

$$\phi(y) = \frac{1}{\sqrt{2\pi\sigma^2}} exp\left[-\frac{(y-\mu)^2}{2\sigma^2}\right] \tag{10}$$

The standard normal is a special case of the Gaussian, or normal, distribution for which μ, the mean, is equal to zero and σ^2, the variance, is equal to one. This is commonly written as $\mathcal{N}(0,1)$. Any normal distribution $\mathcal{N}(\mu,\sigma^2)$ can be transformed into a standard normal distribution using the following equation:

$$Z = \left(\frac{y-\mu}{\sqrt{\sigma^2}}\right) \tag{11}$$

Using this transformation, Equation 10 becomes:

$$\phi(Z) = \frac{1}{\sqrt{2\pi(1)^2}} exp\left[-\frac{(Z-0)^2}{2(1)^2}\right] \tag{12}$$

In order to calculate the probability that the response at point x is less than the upper bound of the response target region (0.1 for pitching moment), y in Equation 11 would be replaced with $T+\epsilon$. Likewise, $m_n(x)$ would replace μ and $s_n(x)$ would replace σ. The resulting Z is plugged into Equations 9 & 12 to calculate the CDF of the normal distribution. The CDF value which results is the probability on a zero-to-one scale that the true response at point x is less than $T+\epsilon$, i.e. $\Phi\left(\frac{T+\epsilon-m_n(x)}{s_n(x)}\right)$.[1]Once the calculation is repeated with $T-\epsilon$ instead of $T+\epsilon$, the results can be substituted into Equation 8 to obtain the weighting function value for the test point x.

4.3.3 Application of Mathematical Framework

The necessary mathematical tools to evaluate a candidate point have now been collected. Equation 2 captures how the candidate will affect the inverse covariance matrix and thus the prediction variance (via Equation 4) at each test point. This prediction variance is weighted by Equation 8, which emphasizes test cases where the response value is expected to be close to the value of interest. The weighted prediction variances are then summed to produce a

[1]Remember that Equation 9 is only valid for $Z > 0$, i.e. if $(T+\epsilon) > m_n(x)$. If the response at point x is expected to be *less* than $T+\epsilon$, for example, and thus $Z < 0$, Equation 9 will produce nonsensical results. Whenever $Z < 0$, Z should be replaced by $-Z$ to stay within the applicable range for Equation 9. This transforms $\Phi(x)$ from the likelihood that $Z > 0$ ($P(Z > 0)$) into the likelihood that $Z < 0$ ($P(Z < 0)$). This can be easily accounted for by use of the identity $P(Z > 0) = 1 - P(Z < 0)$

weighted Integrated Mean Squared Error (wIMSE) score. This wIMSE score quantifies the amount of prediction variance, or uncertainty, that would be present if the candidate point were added to the model.

If the problem only features one response, the algorithm simplifies to the form proposed by Picheny et al., and the candidate with the lowest wIMSE score would be selected as the next sample because that candidate is expected to produce the largest reduction in prediction variance for that response.

If the problem features multiple responses, further steps are required before the best candidate can be identified. First, because the prediction variance of each response may differ by multiple orders of magnitude, the wIMSE scores for each response must be normalized. This is accomplished using the mean and standard deviation of the wIMSE scores for that response, μ_{wIMSE} and σ_{wIMSE}. The normalized wIMSE score for the i^{th} candidate is thus:

$$wIMSE_{i,norm} = \frac{wIMSE_i - \mu_{wIMSE}}{\sigma_{wIMSE}} \tag{13}$$

Once wIMSE scores have been normalized for all responses, the average wIMSE score for each candidate is calculated based on its wIMSE score for each response. The candidate with the smallest average wIMSE score is chosen as the next point to be sampled.

4.4 Use of Alternative Surrogate Modeling Methods

It should be noted that Kriging is not necessarily required for this algorithm. Strictly speaking, the algorithm requires only (a) a method of predicting the response value at some new point, and (b) a way of estimating the prediction variance or uncertainty at that point. Any surrogate modeling technique should allow the prediction of response values, and cross-validation offers a way to estimate prediction uncertainty at any given point.[148] However, cross-validation requires the generation of many additional surrogate models, which may become very demanding in terms of time and computational effort. The attractiveness of Kriging for this effort stems from the fact that with a Kriging model, prediction variance may be calculated directly with relatively low computational effort, and the prediction variance is known to have a Gaussian distribution. If Kriging were to become excessively expensive, such as if a very large data set became necessary, cross-validation and an alternative surrogate

modeling method should still allow this sampling algorithm to be applied with minimal modifications – albeit at what is expected to be significantly increased computational effort & time.

4.5 Verification of Sampling Algorithm

Before any implementation of an algorithm may be applied, it should first be tested to verify that the algorithm as coded performs as intended. This serves to confirm that the implementation accurately reproduces the intended algorithm, and that the results produced by the implementation adequately reflect the performance of the algorithm.[8, 139] Once the sample-selection code is verified to be performing as intended, its performance could be assessed with greater confidence.

Toward this end, four verification experiments were conducted. The first three verification experiments would feature two input parameters and one output parameter, similar to a demonstration given by Picheny et al.,[149] to assess the basic behavior of the algorithm and its subroutines. The final verification experiment would feature two input parameters and three output parameters to verify algorithm behavior for multiple responses.

Cart3D data would be used for the first and last verification experiments so that the algorithm could be demonstrated for a response representative of the intended application. To select the 2 free parameters from the 49 parameters used in the RBS effort, a sensitivity study was performed to quantify the relative effect of each free parameter excluding control deflections. This study indicated that for the three flight conditions analyzed, the wing root chord fraction and the fuselage radius fraction had the largest average impact on vehicle pitching moment.

Using these two parameters as independent variables, the first verification experiment would apply the contour-based sampling algorithm to one flight condition. This would allow the comparison of the behavior of the implemented algorithm against the behavior described by Picheny et al. Two other verification tests were developed based on standard test functions used in the optimization field due to the similarity between adaptive sampling algorithms and optimization algorithms.

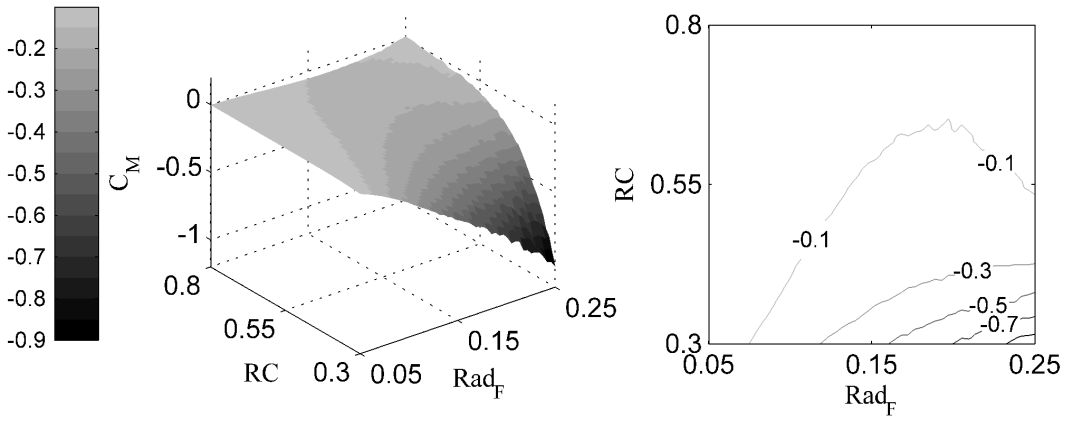

Figure 9: Pitching Moment Coefficient at Mach 2.5, α 0°

The final verification experiment would assess the performance of the algorithm when applied to three flight conditions simultaneously. This experiment would investigate how the algorithm would behave when applied to a problem with multiple responses. This represents an extension beyond what has been described in the literature for this algorithm.

Before these experiments could commence, default values for the other 47 geometric parameters had to be selected. An iterative approach was used to identify a set of default values such that the two-dimensional design space would include configurations with small pitching moments at all three flight conditions. The values that were selected have been documented in Appendix B. Department of Defense High Performance Computing Center resources allowed this approach to be performed in a rapid, massively-parallel fashion, significantly reducing the time required.

4.6 First Sampling Verification Experiment: Two Inputs, One Response

This verification experiment served to confirm that the algorithm proposed by Picheny et al. had been implemented correctly. It was expected that the test problem was simple enough that any aberrant behavior could be identified relatively easily. The Mach 2.5, α 0° flight condition was chosen for this test because at this condition, the test problem exhibits fairly simple response behavior, as seen in Figure 9.

A large fraction of the design space produced pitching moment coefficients (C_M) within

the region of interest, i.e. less than |0.1|, with the exception of cases with a large fuselage radius fraction (Rad_F) *and* a small root chord fraction (RC). Such cases exhibited a pitching moment that was more negative than desired. It was expected that, once this trend was identified, the algorithm would avoid placing samples in the undesirable region and instead emphasize the region where the pitching moment is close to zero.

4.6.1 Contributing Analyses: Prediction Variance

Before verifying the overall behavior of the algorithm, a number of intermediate checks were performed to build confidence in the underlying calculations. Because the algorithm depends on the accuracy of the estimated variances, the prediction variance values produced by the implemented algorithm were evaluated first.

From a conceptual standpoint, the prediction variance was expected to be small near existing samples where the response value was known exactly and grow larger for points farther from the existing samples.[182] Applying these calculations to a two-dimensional problem allowed the output to be evaluated visually. This evaluation would serve as the first check on the calculated values: if this behavior was not observed, there was likely a problem with the implementation.

The DACE toolbox by Lophaven et al.[107] for Matlab[123] has become a popular utility for the creation of Kriging models.[54, 62, 187, 203] In addition, the toolbox offers the option to estimate prediction variance at any point. These estimates would be compared to those produced by the algorithm; this comparison would serve as the second check on the calculated values. If good agreement was found, this would be taken as a sign that the variance estimation portion of the algorithm was working as intended.

A 50 $\times$ 50 grid of configurations was generated, spanning the ranges of the wing root chord fraction and the fuselage radius fraction, and these configurations were analyzed with Cart3D at Mach 2.5, α 0 °. A five-point Latin hypercube was generated and the pitching moment for each point was estimated by interpolating the grid results. These cases and the respective pitching moment for each were then used to train a Kriging model via the DACE toolbox. That Kriging model was the source of the baseline inverse covariance matrix

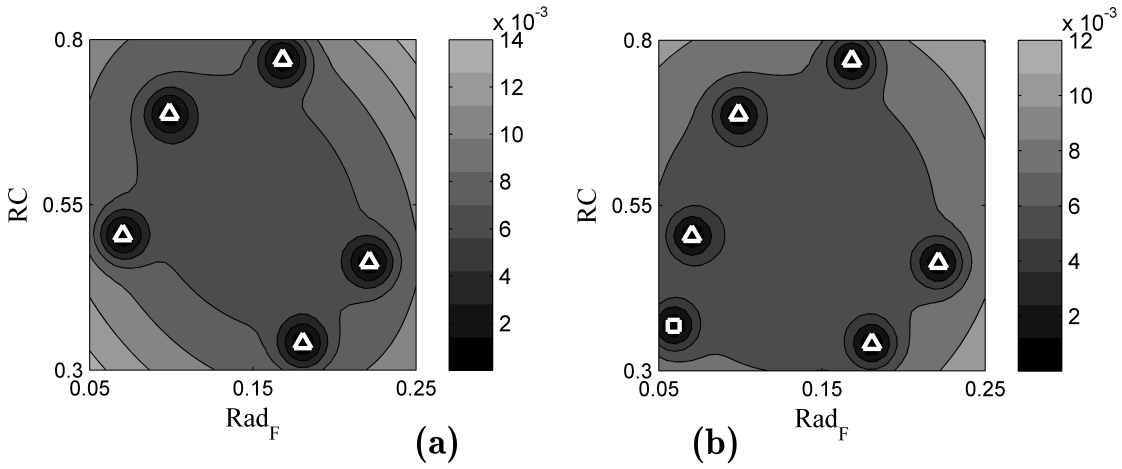

Figure 10: Comparison of Prediction Variance Estimates Produced by (a) DACE and (b) the Implemented Algorithm

(C^{-1} in Equation 7) which was then used to calculate the augmented covariance matrix if a particular candidate point were added to the model. The results of this test may be seen in Figure 10.

In this figure, the triangles represent the five points used to create the Kriging model. In Figure 10b, the square in the lower left represents the candidate point that is being added to the model (assuming that the response value at that point does not significantly affect the estimated model weights).

Note that in both Figures 10a and 10b, the prediction variance is smallest in the neighborhood immediately around each point. Additionally, the prediction variance is relatively low in the center of the space where all samples are somewhat nearby, and grows to large values in the corners of the space, which are the farthest from the sample points.

In Figure 10b, the new sample point clearly reduces the nearby variance values. The close agreement between the two images (both visually and numerically) as well as with the expected behavior indicates that this aspect of the algorithm is functioning correctly.

4.6.2 Contributing Analyses: Weighting Function

After prediction variance, the next set of calculations to be verified were those supporting the weighting function. The weighting function was used to identify regions where additional

experiments would best improve the Kriging model; errors in the calculation of the weighting function would be harmful to the accuracy and efficiency of the method.

The weighting function can be thought of as the probability that the value of the response at a given test point falls within the region of interest. Equation 8 stated that the weighting function is equal to the probability that the response will be less than the upper threshold, minus the probability that the response will be less than the lower threshold. Using Equations 9, 11, and 12, those probabilities can be calculated using only values for the response and variance at that point, both of which can be calculated by the Kriging surrogate model.

Conceptually, it was expected that the weighting function might exhibit a variety of behaviors. If at some point the estimated response was between the two cutoff values and the prediction variance was small, the weighting function at that point should be close to 1, which would indicate that there was a very high likelihood that the true response at that point fell within the range of interest. As the variance grew larger (i.e., the confidence in the estimate decreased) there was an increasing chance that the actual response at that point was outside the range of interest, and thus the weight would decrease. Alternatively, as the predicted response moved farther from the region of interest the weight would decrease as it became less likely that the actual response still fell within the region of interest.

Figure 11a depicts the predicted responses throughout the space based on the initial space-filling cases. In this example, the Kriging model uses a linear underlying model. Recall from Equation 1 that the predicted response is a combination of the underlying model, which captures the general trends of the response, and the covariance matrix which accounts for deviations from that trend. In this case there are relatively few samples, and the response predictions are dominated by the linear model. The covariance effects may be seen only in the close vicinity of the sample points. The prediction variance for this example may be seen in Figure 10a. In essence, the variance was small near the samples and large near the edges of the space.

The calculated weights throughout the space are plotted in Figure 11b. The weights are primarily driven by the predicted response: high weights are observed where the response

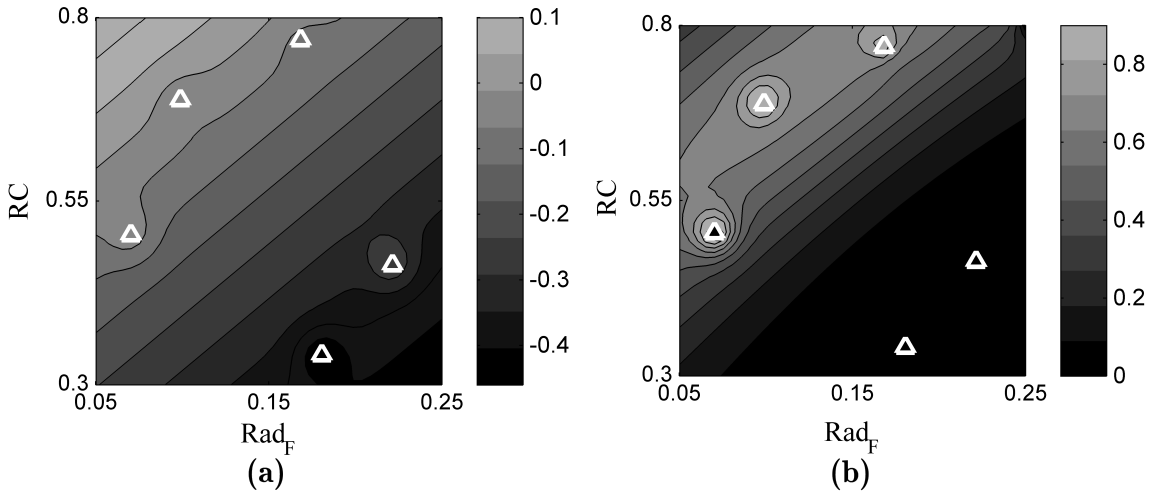

Figure 11: Examples of Predicted C_M & Weighting

is expected to fall within the range of interest, and low weights where the response is far from the range of interest. The weight value smoothly tapers between those regions, indicating points which are unlikely to have good response values, but with enough prediction uncertainty that they cannot be ruled out entirely.

It is critical to note that the weighting function depends on the *estimated* response and variance at each point. Comparing the actual response in Figure 9 and the predicted response in Figure 11, it is clear that with only five samples, the initial Kriging model achieves only a rough approximation of the response behavior. As a result, the algorithm may at first select sub-optimal candidates due to imperfect information. The model is re-generated after each sample, however, and will progressively improve itself as more information becomes available.

Based on this qualitative assessment, the implementation of the weighting function appears to be functioning as intended.

4.6.3 Contributing Analyses: wIMSE Calculation

Having verified the performance of various components of the algorithm, the behavior of the algorithm as a whole could be assessed. The intent was to evaluate candidate points and identify the one which most reduced prediction uncertainty (i.e. variance) in regions where

the response value was within the desired range. It had been qualitatively demonstrated that the algorithm correctly modeled how a new sample would affect nearby prediction variance. It had also been qualitatively demonstrated that the algorithm could assign proper weights to cases based on estimated response and variance values. It only remains to combine the two features into a single scoring metric.

Weighted Integrated Mean Squared Error, or wIMSE, was the metric used to rank candidates. The unweighted form, IMSE, was used by Kleijnen and Van Beers[97] as an approximation of the overall prediction uncertainty in the design space. Integrating the variance analytically would be difficult at best; Kleijnen and Van Beers use numerical integration, by way of adding up the variance at each point in a representative set of samples. The candidate point that most reduced the unweighted IMSE could be thought of as the point which best reduced overall prediction uncertainty.

The use of a weighting factor allowed certain regions to be emphasized and others downplayed. In this case, the weighting factor was large in regions with desirable response performance – for example, regions where the vehicle pitching moment was close to zero – and small in regions with poor response performance. Thus, the candidate with the smallest wIMSE value was the case expected to most reduce prediction variance for the region or regions of interest.

If the algorithm was functioning as intended, it should select candidates which promised the largest reduction for prediction variance in regions of interest. It should, therefore, select candidates in regions of large variance and/or in regions where the weighting function, as a proxy for level of interest, was also large.

The algorithm was presented with a 23×23 grid of candidates and a 40×40 grid of test points. Pitching moment coefficient values at each of those points were calculated by interpolating the 50×50 grid of Cart3D results. Grid sampling could quickly become uneconomical for large real-world problems,[193] but for the problem at hand it offered the significant advantage of repeatability.

Given those candidates and test points, the variation of wIMSE score throughout the design space could be calculated and depicted visually. Figure 12 shows wIMSE scores

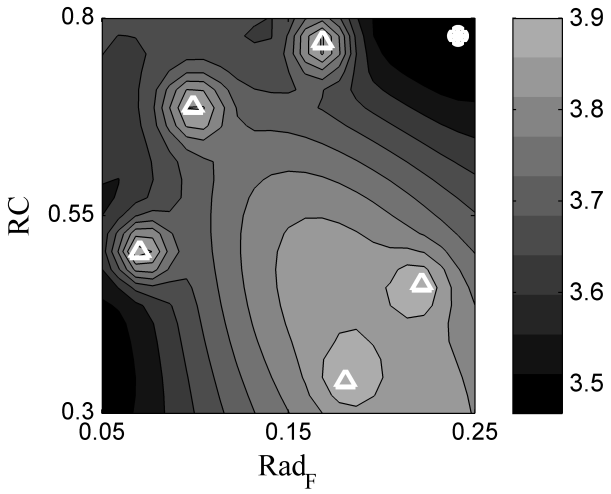

Figure 12: wIMSE Demonstration

for the grid of candidates. The most attractive candidate was the one that produced the largest reduction in weighted variance, which corresponded to the smallest wIMSE score. In Figure 12 this candidate is marked as a white solid circle.

Comparing this image with the estimated variance values in Figure 10a and the weights in Figure 11b, a few observations can be made. The wIMSE values indicated samples in the top-right and bottom-left corners were attractive, which matched the relatively high prediction variances in those regions. Candidates close to existing samples were rated as unattractive, which agreed with the observation that prediction variance in those regions was already low; there was more room for improvement elsewhere. Finally, the upper-left corner was considered to be significantly more desirable than the lower-right corner even though both had roughly equal levels of variance. This showed the influence of the weighting factor, which favored candidates in the upper-left based on the high probability that those cases had desirable response values.

Qualitatively, the algorithm appeared to be functioning as expected. The final test in the set was to evaluate the performance of the algorithm quantitatively.

4.6.4 Evaluation of Accuracy

Given the initial set of five space-filling data points, the algorithm was used to select fifteen adaptive samples. Kriging surrogates were created after each sample selection and used to predict the pitching moment at candidate and test points throughout the space, once again using the 23×23 grid of candidates and 40×40 grid of test points. In each round, after a candidate was selected as the next sample, the actual pitching moment for that point was determined by interpolating the 50×50 grid of Cart3D analyses.

The distribution of the samples appears in Figure 13a & 13b. These images show the initial five space-filling cases (triangles), and fifteen samples selected by contour-based sampling (circles). Figure 13b denotes the order of the samples chosen, laying them over a full contour plot of the estimated response. The next point to be sampled is marked by a filled circle in Figure 13a and by the point labeled 16 in Figure 13b.

Based on the response values for the cases that had been sampled, the response behavior throughout the space was estimated. Figure 13a shows the estimated region of interest, based on the Kriging model, which is bounded by the smooth solid contour line marked "-0.1". The actual region of interest is bounded by the somewhat more erratic dotted line. There is fairly good agreement between the two.

The next notable observation was the overall distribution of the samples selected by the algorithm. Samples were clustered tightly together in the region of interest. The lower-right area, where cases are highly unlikely to be of interest, was untouched except for the initial space-filling cases. If the samples were evenly distributed throughout the design space, more samples would have been placed in this region in the lower-right. Instead, the algorithm correctly identified the region as being of low interest and did not place samples there.

Another observation deserving emphasis was the order of the samples. Figure 11 on page 93 shows the initial prediction of response behavior; the lower-left corner was also expected to be of little interest. As seen in Figure 13b, the exploratory sampling which identified this region as interesting did not occur until the eighth or ninth round. Prior to that round, the algorithm determined that interior samples would improve prediction variance more than exploration of the edges of the estimated region of interest.

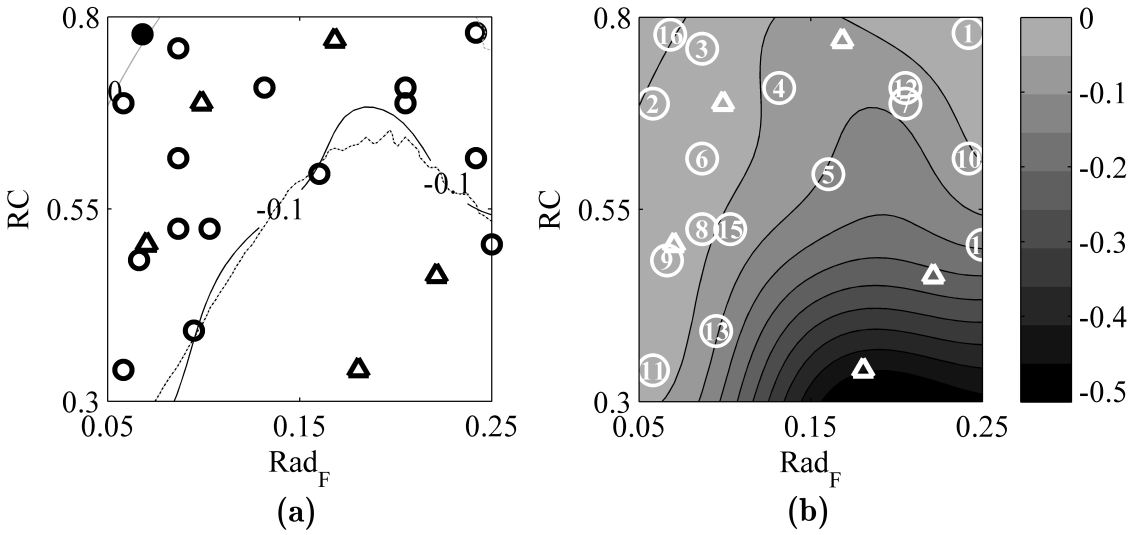

Figure 13: Distribution of Samples & Order of Sample Selection

This is reflected in Figure 14, which shows how the 2.5% and 97.5% prediction error quantiles for the model changed as each new sample was selected and incorporated. Error here was defined as $Y_{predicted} - Y_{actual}$ so that negative errors indicated too-low predictions and positive errors indicated too-high predictions. The 2.5% and 97.5% quantiles represented some of the most-negative and most-positive prediction errors made by the model. Note that the 2.5% and 97.5% quantiles encompassed 95% of the prediction errors. The average prediction error was between the two values.

For comparison, for each number of samples, equivalent-sized Latin hypercubes were generated and used to build models. To minimize the chance that lucky or unlucky sample distributions might skew the comparison, many hypercubes were generated for each sample size. The number of hypercubes was increased until the average metric value displayed relatively smooth trend behavior rather than erratic noisy behavior. To produce the results illustrated below, three hundred[2]Latin hypercubes were generated for each sample size.

Note that after the fifteenth sample (i.e., the tenth adaptive sample plus the five initial cases), the lower quantile for adaptive sampling improved markedly. This corresponded to

[2]The goal of hypercube sampling was to estimate the performance of an average space-filling sample design. Three hundred repetitions at each hypercube size gave fairly smooth average results without requiring excessive analysis time.

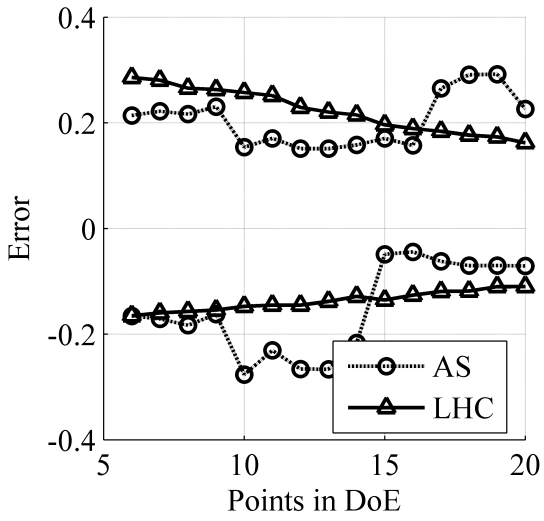

Figure 14: 2.5% & 97.5% Prediction Error Quantiles for Adaptive & Space-Filling Sampling: Entire Space

a shift from predictions which were previously too low and were now closer to the correct value. Prior to the tenth sample being placed there, the response in that region was expected to be more negative than it truly was; after the tenth sample, the algorithm appeared to have a firm grasp on the interesting regions of the problem.

From Figure 14, the reader might conclude that the sampling algorithm and the average space-filling sample distribution are roughly evenly matched for this problem. Before drawing that conclusion, however, bear in mind that the main objective of this algorithm was to maximize prediction confidence and accuracy *only* for cases where the response value is within some range of interest. To this end, the prediction error quantiles were again calculated, but this time only for points where the response fell within the specified range of $|C_M| \leq 0.1$. Those quantiles, calculated for both the sampling algorithm and the Latin hypercubes, are plotted in Figure 15.

When only the cases of interest are evaluated, the sampling algorithm was found to be significantly more accurate than the average Latin hypercube. After the tenth sample, the sampling algorithm had successfully identified and sampled all regions of interest within the design space. The prediction error quantiles then became tightly grouped around the line of zero error, in contrast to the quantiles for space-filling samples.

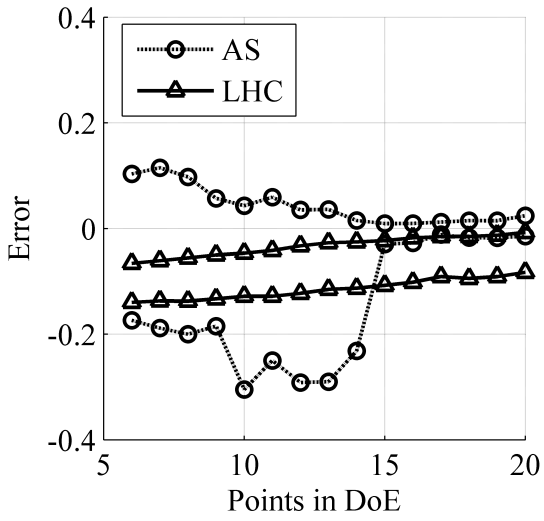

Figure 15: 2.5% & 97.5% Prediction Error Quantiles for Adaptive & Space-Filling Sampling: Region of Interest Only

Unlike the previous image, here the quantiles for the average hypercube-based model did *not* bracket zero, but rather both were below zero. Because the average error commonly lies between the two quantiles, this result suggested a bias in the predictions of hypercube-based models: the responses at interesting cases were being consistently under-predicted, a behavior that was not observed in the adaptive sampling results.

4.7 Second Sampling Verification Experiment: Perm Function

The **Perm function** is a test function used to evaluate the performance of optimization algorithms.[196] This function can be generated with n dimensions where n is any integer; for this application only two dimensions were used. The equation for the function is:

$$f(x) = \sum_{j=1}^{n} \left\{ \sum_{i=1}^{n} \left(i^j + \beta\right) \left[\left(\frac{x_i}{i}\right)^j - 1 \right] \right\}^2 \tag{14}$$

In this function, β can be varied which affects how closely the local minima approximate the global minimum. The function is evaluated over the range $-n \leq x_i \leq n$ and the global minimum is $f(x) = 0$ at $x_i = i$ where $i = 1 \cdots n$. For this experiment, β was set to 0.5 on the recommendation of Hedar.[77] For this value of β, the function produces values from 0 to slightly over 100.

The behavior of the function is plotted in Figure 16a. The function generally increases toward the low end of each variable, and a hump is present in the center of the space. Due to the interplay of these two behaviors, many of the points in the lower-left quadrant have similar $f(x)$ values. Selecting 60 ± 5 as the "range of interest" for this function results corresponds to the region plotted in Figure 16b. This area has nonlinear edges and is mildly concave.

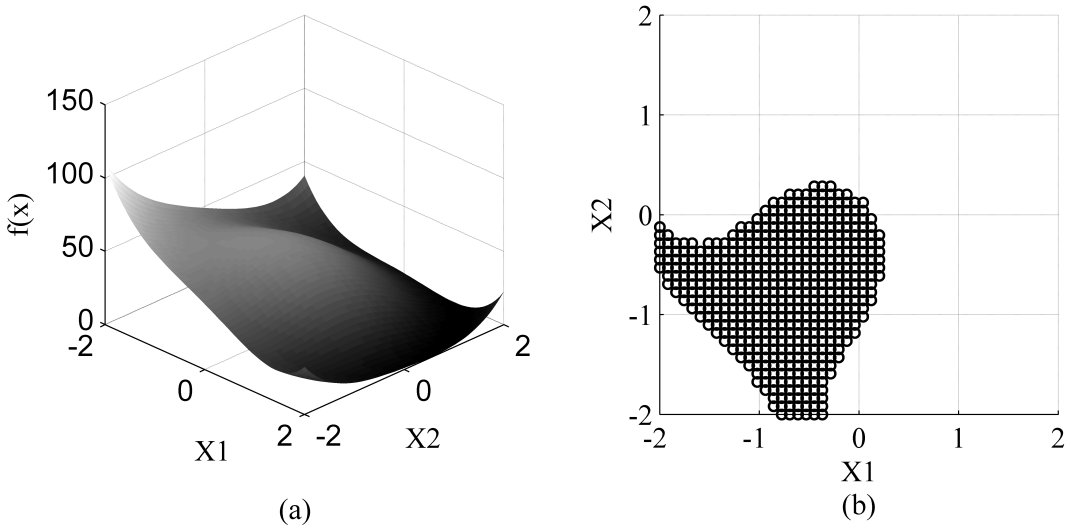

Figure 16: Perm Function & Region of Interest

The sampling algorithm was applied to this test problem for the stated range of interest. A 5-point Latin hypercube was used to initialize the algorithm, after which 15 samples were selected. After each selection, a new Kriging model was trained and used to predict the response values for the region shown in Figure 16b. The predictions were compared to the actual response values for those points, and the prediction error and **Root Mean Squared Error** (RMSE) for the predictions were calculated.[85]

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} \left(Y_{pred,i} - Y_{actual,i}\right)^2}{N}} \tag{15}$$

Additionally, 1,000 Latin hypercubes were generated for each sample size (e.g., 6, 7, ...). A new Kriging model was trained for each hypercube and the model was used to predict response values for the cases of interest. The average prediction RMSE for each sample size

was recorded. The RMSE values produced by Latin hypercubes and contour-based sampling are compared in Figure 17.

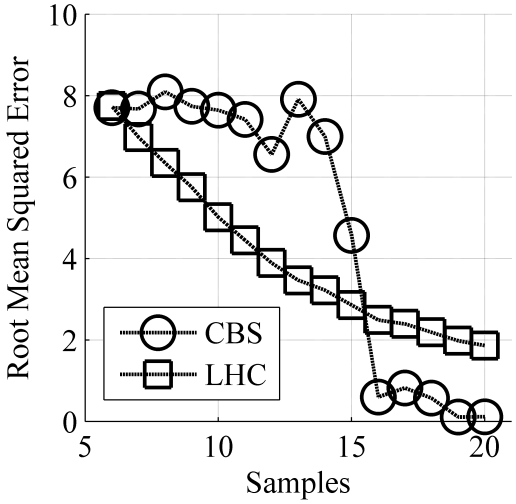

Figure 17: Comparing Predictive Accuracy for Perm Function Region of Interest

The model that used contour-based sampling started out with worse performance, and at first it did not improve very rapidly compared to space-filling samples. After 15 samples, the model based on adaptive sampling had identified the region of interest and had reduced its prediction error by an order of magnitude compared to the average hypercube performance with an RMSE of 0.160 versus 1.86 for hypercubes.

4.8 Third Sampling Verification Experiment: Sphere Function

The **Sphere function** is another test function from the field of numerical optimization.[146] It is a fairly simple function:

$$f(x) = \sum_{i=1}^{n} x_i^2 \tag{16}$$

This function can also take any number of dimensions; once again two dimensions will be used for the sake of simplicity. The search domain for this function was set to $-5.12 \leq x_i \leq 5.12$ based on the recommendation of Hedar.[78] This function has a global minimum of 0 when $x_i = 0$ for all i. The behavior of the function is shown in Figure 18a.

It was decided to make this evaluation more challenging than the previous tests. Rather than choosing a convex region centered about the minimum value, the range of interest

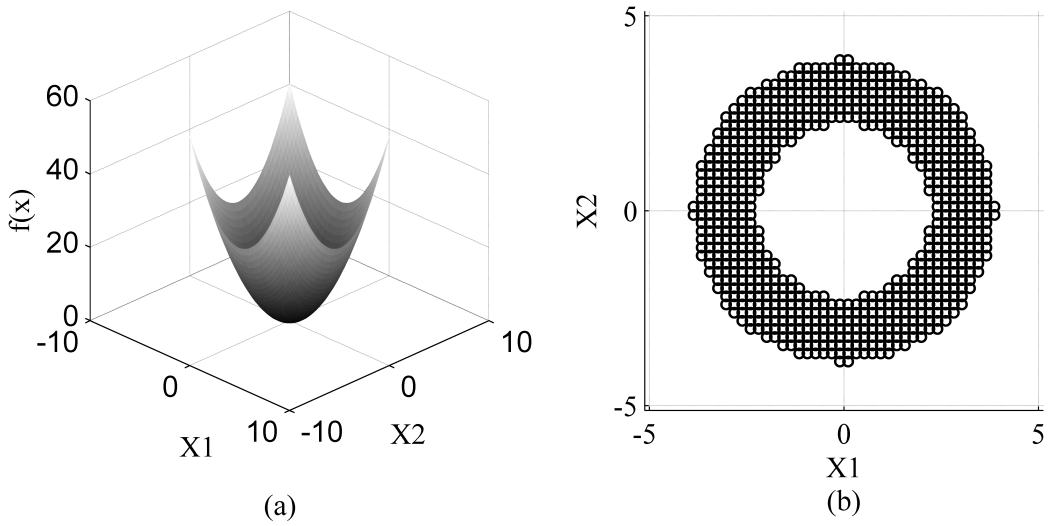

Figure 18: Sphere Function & Region of Interest

was defined to be $5 \leq f(x) \leq 15$. Cases which met this criterion are marked with small circular icons in Figure 18b. These cases were spread across a large portion of the design space and the grouping was non-convex. Furthermore, the linear underlying trend of the Kriging models was not a good match for this function, which meant that the estimated response values used for the adaptive sampling algorithm might be significantly inaccurate. Once again, the predictive accuracy of the model which used contour-based sampling was compared against the average prediction RMSE of 1,000 Latin hypercubes at each sample size. The results of this comparison can be seen in Figure 19.

Models using the space-filling samples once again showed smooth improvement, and after 15 samples reached diminishing returns as the response behavior was fairly well understood. The model based on adaptive sampling, on the other hand, could at best be said to struggle with this problem. The first 4 samples produced significant *decreases* in predictive accuracy. Subsequent samples showed a gradual improvement at a rate slower than the rate exhibited by hypercube sampling. Only in the final 2 samples did the model based on adaptive sampling identify the true form of the response and react accordingly.

The large vertical scale required to include all the data points precluded a detailed visual comparison of the two methods at the end of sampling. It was found that after 20 samples,

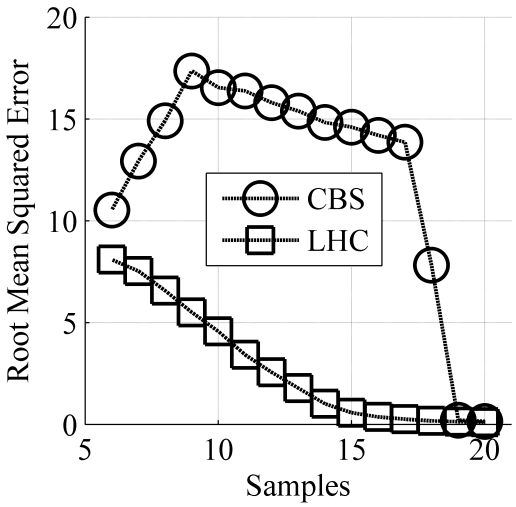

Figure 19: Comparing Predictive Accuracy for Sphere Function Region of Interest

the average space-filling model could claim a prediction RMSE of 0.109 while the model based on adaptive sampling produced an RMSE of 0.151. Although the adaptive model showed substantial improvement, that does not excuse the poor performance which was demonstrated for the bulk of the experiment.

The contour-based sampling algorithm was shown to perform well for problems which have simple regions of interest, but the algorithm could be applied to other problems with some accompanying loss of effectiveness. In addition, because a surrogate model is used when evaluating candidate points, the accuracy of that surrogate model can significantly affect the effectiveness and efficiency of the sample selection process. Even when the Kriging model is a poor representation of the response behavior, continued sampling can eventually correct this shortcoming. Unfortunately, there was no obvious way to know how many more samples will be required before the algorithm would correct itself.

These experiments served to verify that the sample selection algorithm performed as expected. Encouragingly, the algorithm had in many cases been shown to provide better predictive capability for cases of interest compared to hypercube-based results. On the strength of these results, the algorithm was extended to handle multiple responses simultaneously.

4.9 Fourth Sampling Verification Experiment: Two Inputs, Three Responses

The evidence so far, both as part of this effort and in the literature, had only demonstrated contour-based sampling for a single response. With regard to the problem at hand, the aerodynamic moments on vehicles must be controllable at *all* expected flight conditions, not one. The algorithm would thus be modified to identify cases which were beneficial for multiple responses (i.e., cases which improved prediction accuracy for small pitching moments at multiple flight conditions) simultaneously.

This was accomplished by creating an overall sampling criterion which incorporated the performance of a given candidate at all flight conditions. For each candidate, the wIMSE score is calculated at every flight condition as detailed above. Once all wIMSE scores have been calculated, the scores for each flight condition are normalized (as given in Equation 13 on page 87) and then the average normalized wIMSE score is calculated for each candidate. The combined score shall henceforth be referred to as joint wIMSE. This synthesizes the wIMSE information and allows easy ranking of candidates. Before this approach can be adopted, however, it must be demonstrated as effective.

To minimize data generation requirements, the data pool from the first sampling verification experiment was retained. Additional data was generated for the same design space at the other two flight conditions, Mach 0.3, α 15°, β 0° and Mach 0.8, α 0°, β 0°. The response behavior for these two flight conditions may be seen in Figure 20. Note the sharp variations observed at Mach 0.8. Although the pitching moment was converged[3] for all cases or interpolated from nearby converged results, these sharp variations in response were still observed. At the present time it is believed that they result from the transonic flight condition, a condition which the Euler CFD tool may be hard pressed to model.

Using the 50×50 grid of samples, cases of interest for each flight condition are highlighted in Figures 21a, 21b, and 21c. Figure 21d illustrates the cases which have $|C_M| \leq 0.1$ at all three flight conditions simultaneously. The jaggedness of the transonic response behavior

[3]Convergence for these efforts was defined as a standard deviation of less than 0.05 *and* less than 5% of the average response value when evaluated over the final twenty iterations of the flow solver.

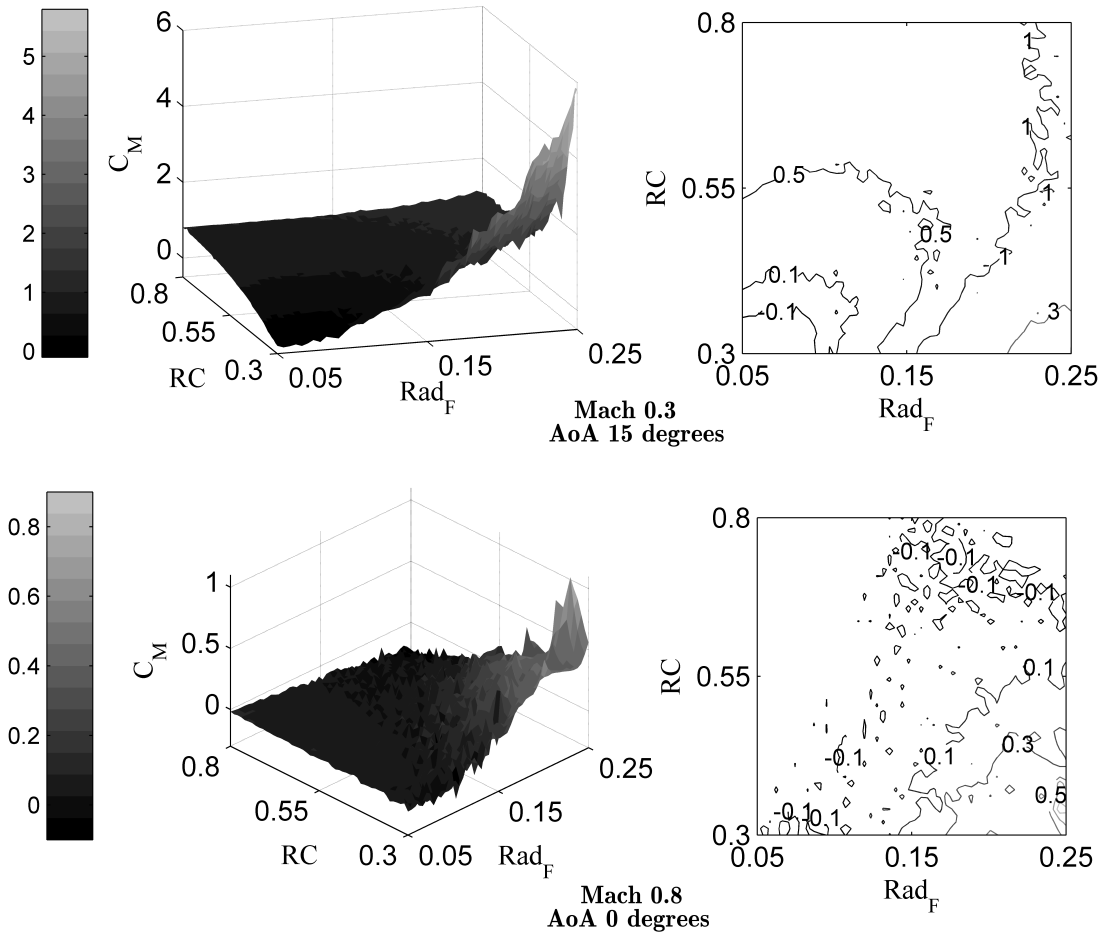

Figure 20: Pitching Moment Coefficient at Mach 0.3, α 15° and Mach 0.8, α 0°

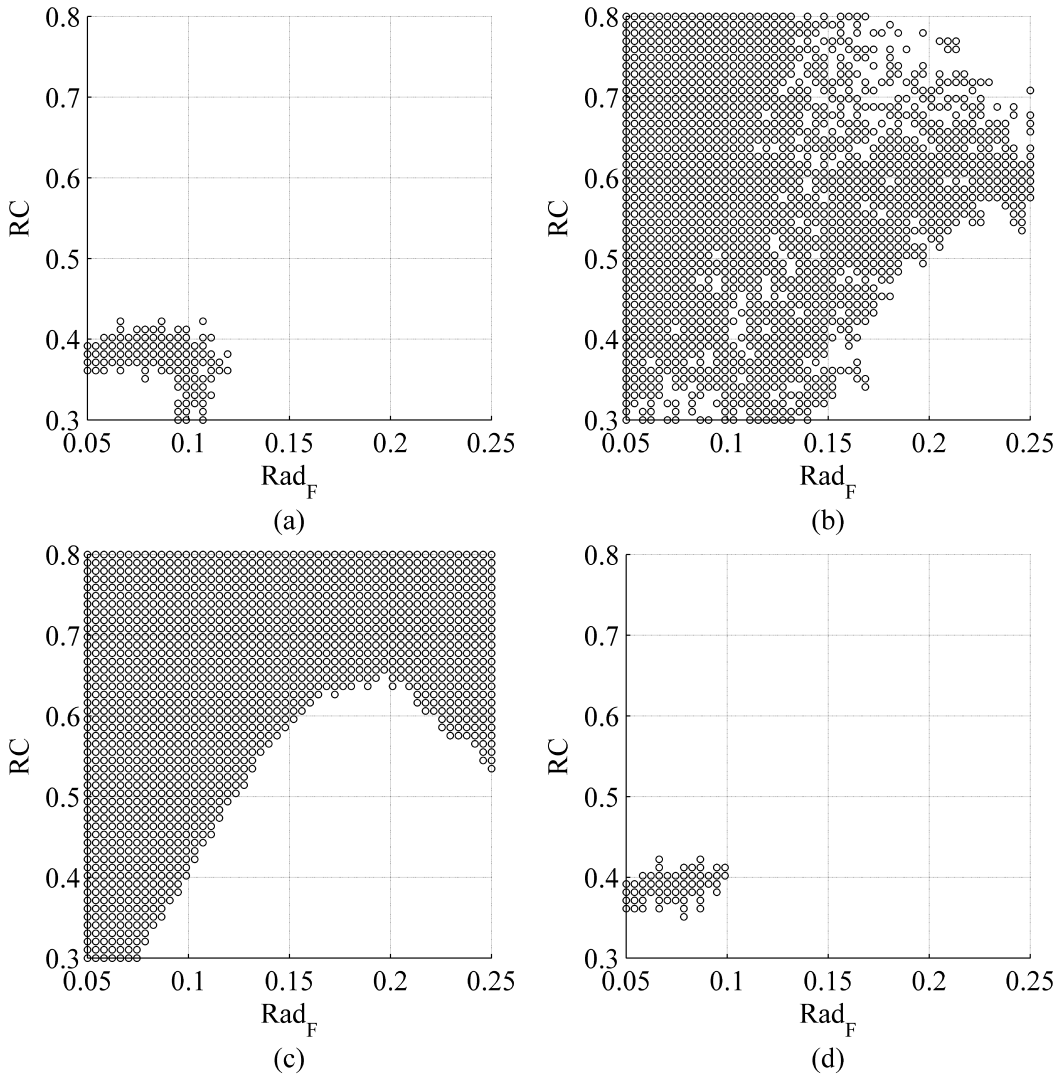

Figure 21: Cases of Interest at Each Flight Condition & At All Flight Conditions

could be inferred from the erratic distribution of cases of interest in 21b. The Mach 0.3 flight condition was clearly the most restrictive, although there were cases which produced an acceptable C_M at Mach 0.3 but an overly-negative C_M at Mach 2.5.

Once the algorithm was modified to evaluate all three flight conditions, sampling began. The same five initial space-filling samples were used for this experiment so that any deviations from the previous sample selections may be attributed to the inclusion of additional responses. The resulting distribution of samples is shown in Figure 22a. Comparing the

samples selected against the regions of interest in Figure 21 suggests that instead of reducing prediction variance where *all* responses fall within the ranges of interest, the algorithm selected cases where *any* response fell within the range of interest. Put another way, although sample 2 was not expected to fall in or near the region of interest for Mach 0.3 or 0.8, it was of great interest for the Mach 2.5 model.

The resulting prediction accuracy for each response for cases of interest (i.e., the cases in Figure 21d) are shown in Figures 22b, 22c & 22d. Note that after the ninth & tenth samples – i.e., the fourth and fifth adaptive samples after the five initial space-filling cases – the prediction accuracy for each response improved dramatically. The fifth sample was very close to the region of interest, resulting in drastic reductions for prediction error.

The distribution of samples was still much more scattered than might have been expected given the relatively few cases which were of interest for all responses. If the purpose of the sampling were to identify cases of interest for any response, this behavior would be acceptable. For the purpose of effective vehicle design, however, the algorithm must emphasize the intersection of the regions of interest rather than the union. A research engineer[49] suggested a strategy of excluding candidate points based on a probability of interest score.

At a particular candidate point, the probability that the response value would fall within the range of interest could be calculated; this was already done for test points as part of the weighting function, but was not necessary for candidate points for the one-response problem. Calculating this probability for candidate points provided an estimation of how likely the candidate was to fall within the region of interest for each response. Once this probability had been calculated for all responses, the minimum value was referred to as the "**Probability of Interest**" or POI for that candidate.

The user may then specify a required POI value before the algorithm begins. Any candidate point with a POI less than the required value will be ignored and assigned an wIMSE score equal to the highest observed value. This assigned wIMSE score designates such points as uninteresting to the algorithm, ensuring that those points will not be selected for sampling in that round.

Care must be taken when specifying a required POI value. Although POI calculations

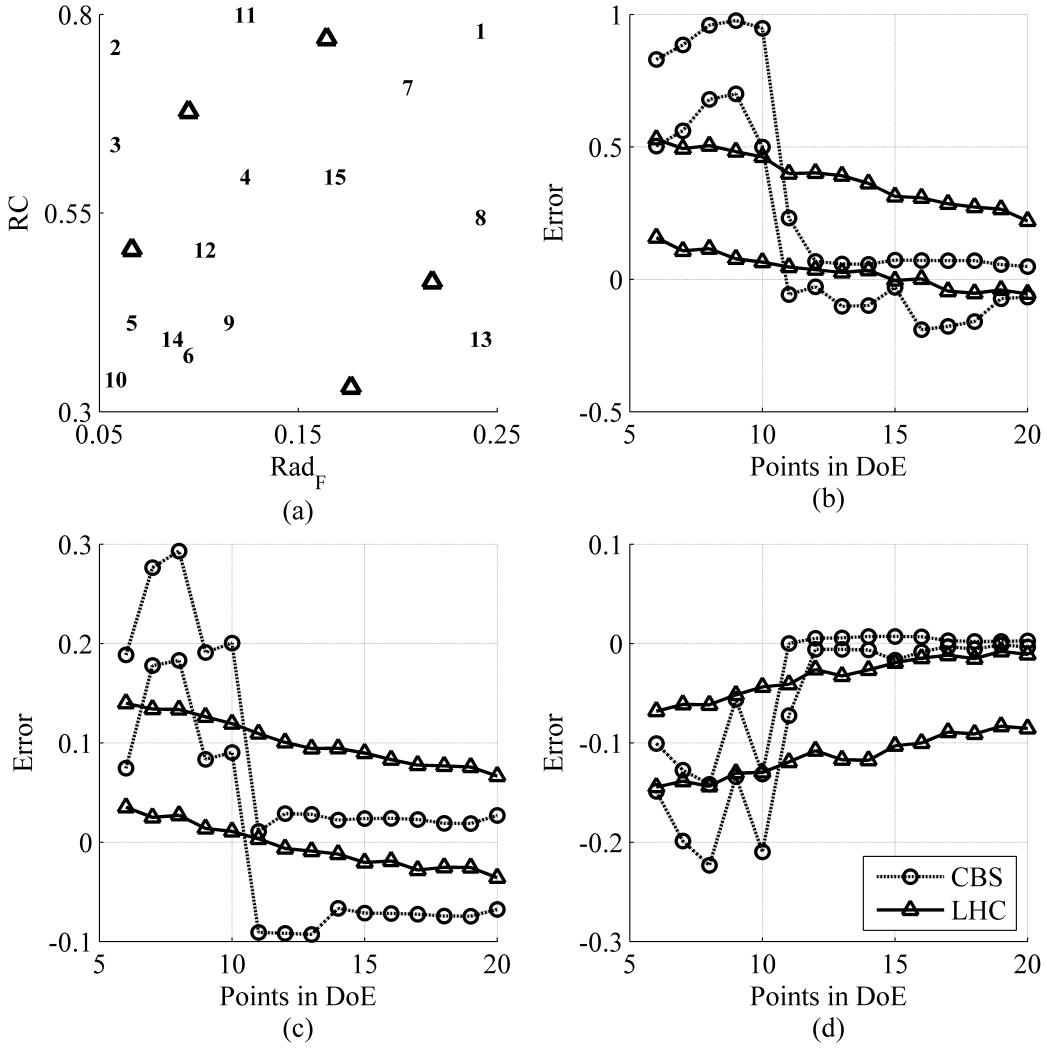

Figure 22: Initial Sampling (a) & 95% Prediction Error Quantiles for Cases of Interest at (b) Mach 0.3, (c) Mach 0.8, & (d) Mach 2.5

are based on the best available Kriging surrogates at the time, those surrogates may not accurately capture the true response behavior, especially when very few samples are available. The estimated response in Figure 11 has only passing similarities to the actual response in Figure 9. Furthermore, this sampling approach is intended for use when resources will only allow a limited number of analyses, increasing the risk that the surrogate models may be inaccurate in the early stages of sampling.

It is tempting to demand a high POI value for each sample to be sure that every point falls in or near the region of interest. As the number of responses grows or the regions of interest shrink – compare the region of interest at Mach 2.5 in Figure 21c to that at Mach 0.3 in Figure 21a – it becomes less likely that any candidate will meet the requirements, especially for large design spaces. For this verification problem, a POI requirement of 25% is enough to disqualify every point in the 23×23 grid of candidates.

If no candidate meets the required POI, the algorithm will select the candidate with the maximum POI, i.e. the candidate that is the most likely to be in a region of interest for every response. This is not necessarily a bad result, given that the point is likely to be in a promising location, but if no candidates meet the required POI then their effects on prediction variance are not considered. As a limiting case, a candidate might fall very close to a previous sample which was in the region of interest. This candidate will probably have a very high POI, but may be so close that it will not provide much new information about response behavior. Another candidate, farther away, might have a lower POI but would provide a much larger reduction in prediction variance.

This was demonstrated with a side experiment, in which the POI requirement was set to 25%. Results are depicted in Figure 23. For the first adaptive sample selection, the candidate with the highest POI was the one closest to the upper-left space-filling point. This candidate had a high POI value because that training point was the closest to having good performance in all three responses, and because prediction variance was relatively low in the region around the training point. The candidate point which was selected, being extremely close to an existing sample, did not provide much new information about response behavior which made it difficult for the algorithm to learn from its mistake. This is shown

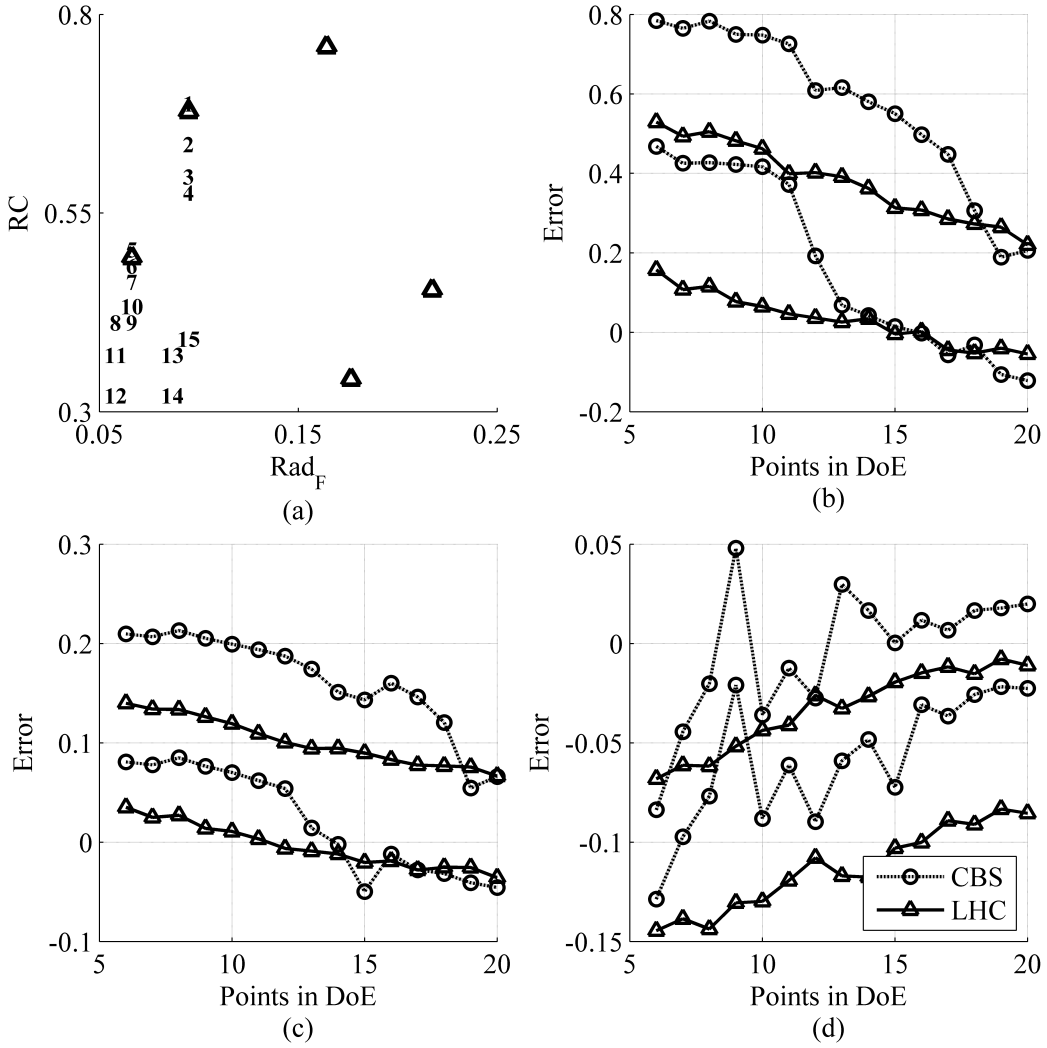

Figure 23: Sampling For Required POI of 25% (a) & 95% Prediction Error Quantiles for Cases of Interest at (b) Mach 0.3, (c) Mach 0.8, and (d) Mach 2.5

clearly in Figure 23a, where the algorithm primarily selected samples that were quite close to the existing data set.

As a result, the prediction error for cases of interest improved very gradually. The average space-filling set of samples produced a surrogate which was approximately just as good as the surrogate trained on contour-based samples, with the exception of the Mach 2.5 case (Figure 23d) where contour-based sampling still offered an improvement. In fact, the model based on contour-based sampling with a POI requirement of 25% in this scenario was *less* accurate than when no POI requirements were used at all (Figure 22). A more tolerant POI setting would allow the algorithm to spread its samples out, and as a result the accuracy of the Kriging models would improve much more rapidly.

The demonstration was repeated with a small POI requirement (1%), as shown in Figure 24. This time, most of the samples were clustered in the region of interest, with occasional forays into other regions that might be promising. The prediction accuracy for all three responses improved more rapidly than the other two cases shown.

In general, a low POI requirement would favor the selection of candidates which would have a large effect on prediction variance, but might not lie in a region of interest for all responses. A high POI requirement would encourage the selection of cases that were more likely to lie in the region of interest, but might be less beneficial from the perspective of variance reduction. Essentially, the POI requirement can be thought of as a means for the user to express a preference between exploration and exploitation.

Other POI requirement values were examined for this problem as well. The results illustrated the effect of the POI requirement in greater detail. The three examples given here should be sufficient to illustrate how the parameter affects the behavior of the algorithm; the other results may be found in Appendix C.

4.9.1 Evaluation of Accuracy

The results of these verification experiments indicated that the sample-selection algorithm was functioning properly. Based on the available information, it estimated the predicted responses and variances throughout the input space and selected samples which would reduce

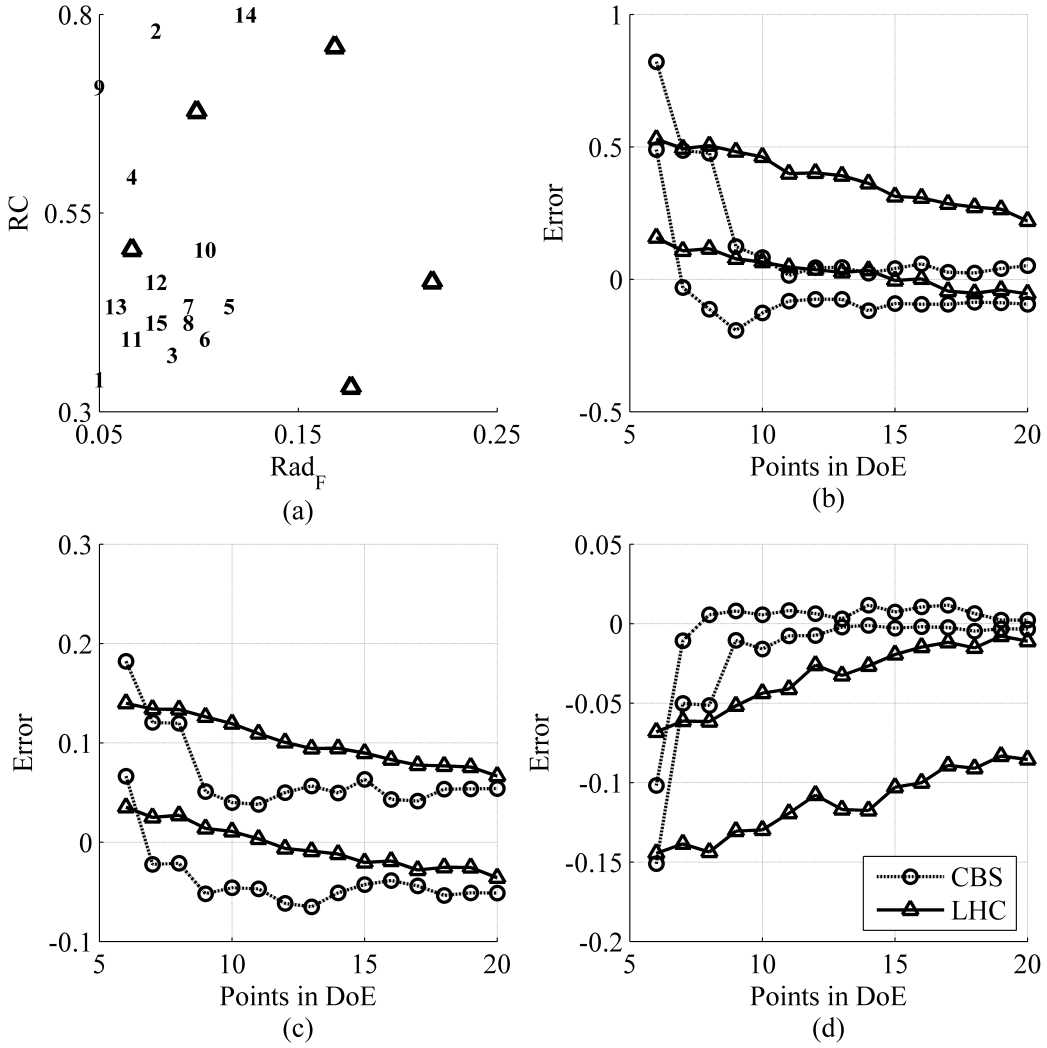

Figure 24: Sampling For Required POI of 1% (a) & 95% Prediction Error Quantiles for Cases of Interest at (b) Mach 0.3, (c) Mach 0.8, and (d) Mach 2.5

prediction variance in specified regions of interest. This reduction in prediction variance corresponded to increased prediction confidence and accuracy in those regions.

For problems with multiple responses, the Probability of Interest parameter was introduced. This parameter allows the behavior of the algorithm to be tuned according to the user's tolerance for samples which may not be of interest for all responses. Using this parameter, the algorithm was demonstrated to effectively and efficiently select samples which greatly improved prediction accuracy in the region of interest compared to space-filling sampling.

The demonstrations thus far have been done for a two-dimensional test problem. Probable applications for this method in the field of vehicle design will likely feature many more free parameters. The final test of the sampling algorithm, and by extension the final test of Hypothesis 1, would feature multiple responses as well as a larger number of free parameters.

4.10 Larger Multi-Response Experiment: Nine Inputs, Three Responses

4.10.1 Selection of Additional Free Parameters Using Sensitivity Analysis

The free parameters for this experiment were selected using a series of sensitivity studies based on the nested Latin hypercube cases that were analyzed in the original Reusable Booster System effort by ASDL and AFRL. The data sets analyzed included at least eleven thousand space-filling samples at each flight condition. The sensitivity studies were performed in JMP, a statistical analysis program by SAS Software.[91] Each study attempted to identify the effects of the forty-nine input parameters on the behavior of the vehicle pitching moment.

The results of the sensitivity study were expressed as fractional contribution, the portion of the observed variation in the response was expected to be due to variation of each input parameter. These fractions were then averaged together to identify the parameters that had the most influence on pitching moment for these flight conditions. The top ten parameters and the average sensitivity are listed in Table 3.

Scale was a parameter that set the overall size of the vehicle. Its presence in this list was unexpected because the pitching moment coefficient had already been normalized by the

Table 3: Partial Sensitivity Study Results

Parameter Name	**Sensitivity Rank**	**Fractional Contribution**
Root Chord Fraction	1	15.82%
Fuselage Radius Fraction	2	9.69%
Nose Droop	3	8.53%
Nose Fineness Ratio	4	6.31%
Wing Airfoil Maximum Camber	5	6.19%
Wing Half-Span Fraction	6	3.66%
Scale	7	3.62%
Area Ratio of Vertical Tail to Wing	8	3.47%
Top Curvature 1	9	3.44%
Top Curvature 2	10	2.99%

reference area. A change in the *Scale* parameter was equivalent to a photographic scaling of the vehicle, which should not affect the pitching moment coefficient.

The magnitude of the sensitivity results held the solution. The parameters could be grouped according to sensitivity values: parameters 2 & 3, parameters 4 & 5 and parameters 6 through 10. Many other parameters (not shown) had values close to 3%. It was inferred that, because most of the parameters had approximately the same effect on the response variation, the sensitivity test was not able to clearly differentiate between those parameters given the data set used. A larger data set would have been able to better distinguish sensitivity effects, but further data was not available.

In light of prior knowledge of how vehicle scale affects the response, it was decided after consultation with AFRL to omit the Scale parameter and perform the sensitivity study using the other nine parameters listed in Table 3. The ranges for the seven new variables which will be included are given in Appendix B.

4.10.2 Infeasibility of Grid Sampling for Test Data

Unlike the experiments with 2 free parameters, a grid search was not feasible over nine parameters. For the two-parameter experiments, each variable was sliced into 50 equal spaces, resulting in 2,500 cases for the full factorial analysis. Achieving the same resolution over nine free parameters would require 1.95×10^{15} analyses. Even a three-level full factorial sampling would require 3^9, or nearly twenty thousand, samples, almost ten times as many as were used for the two-parameter experiments. Any grid search that could be executed in a feasible time would necessarily have very low resolution. It was therefore decided that for this experiment, it would be too computationally expensive to pre-run all the data that might be necessary. Instead, each case would be analyzed as it was required.

As before, the algorithm was intended to preferentially sample cases in the region of interest, where pitching moments were close to zero at all flight conditions. Because the default settings for the seven new parameters were all within their current ranges, it was known that *some* cases of interest exist within the nine-dimensional space. Sixteen such cases were already identified for the two-dimensional experiments, all tightly clustered. More cases of interest were desired, however: the more cases of interest available, the better the prediction accuracy of various surrogate models can be evaluated.

4.10.3 Alternative Approach: Genetic Algorithms

At this stage of the research effort, time was considered more valuable than computing effort. HPCC systems were available to analyze a large number of cases in parallel, and it had been indicated that more resources could be allocated to the effort if necessary. Essentially, the per-analysis cost on HPCC systems was negligible. The ability to run dozens or hundreds of cases simultaneously allows a large population of cases to be evaluated in a timely fashion, making **genetic algorithms** (GAs) an attractive option.

As Grefenstette states, "[w]hile classical gradient search techniques are more efficient for problems which satisfy tight constraints, GAs consistently outperform both gradient techniques and various forms of random search on more difficult (and more common) problems,

such as optimizations involving discontinuous, noisy, high-dimensional, and multimodal objective functions."[69] Although the response was expected to be continuous, the plot of C_M behavior at Mach 0.8 in Figure 20 (see Figure 20 on page 105) indicated that the pitching moment at that flight condition may be noisy. The relatively complex behavior observed in the two-dimensional response at Mach 0.3, α 15° (see again Figure 20) indicated that multi-modal behavior could not be ruled out. The 9 active input parameters here did not unambiguously make the problem "high-dimensional," but this application could be considered a warm-up: if the genetic algorithm method performed well in this case, the same method would likely be used to identify test cases for the full-scale problem which has 49 dimensions. In light of these observations, a genetic-algorithm-based approach seemed plausible.

Although GA optimization can be powerful, the method has its share of drawbacks. One of the negative aspects of the genetic algorithm approach, the inability to guarantee that a true optimum has been found, was immaterial in this case: it need only find cases with small objective functions, not a global minimum. Another negative is that the technique often requires a high number of function calls relative to other optimization techniques. This was of no consequence when the per-analysis cost was considered minimal in light of HPCC resources. The drawbacks of genetic algorithms were relatively minor for this application. A review of the method is therefore appropriate.

4.10.4 Review of Genetic Algorithms

There are many different approaches under the umbrella of Genetic Algorithm optimization, and spatial limitations prevent them all from being discussed in depth in this document. For a more thorough review, see chapter six of Holland's Adaptation in Natural and Artificial Systems.[81]

The typical GA formulation will discretize each variable into a binary string. The number of bits in the string describing that variable will determine the resolution for that variable. A variable represented by one bit has two possible values (typically the minimum and maximum values allowed); two bits allows the encoding of four values, and so on. The binary strings for

each variable were concatenated to produce a single binary string, called a "chromosome," that described a particular case.

Most GA algorithms apply two operations to the n^{th} population in order to generate the $(n+1)^{th}$ population: *reproduction* and *mutation*. During reproduction, one or more "parent" population members are selected and their objective function scores, or fitness values, evaluated. The member with the better score continues on to the next step, which may involve direct descent, genetic crossover, or a combination of the two. Direct descent adds the member to the $(n+1)^{th}$ population as a "child" without any changes, which ensures that good-scoring cases will remain as "breeding stock" in the next round. Genetic crossover, on the other hand, usually takes two "parent" members and swaps a portion of the "genes" which define each case. In most applications, crossover is a probable but not certain occurrence. The user determines the likelihood that crossover will occur, with typical values in the range of 50–90% depending on the population size.[69]

Mutation is the final stage before a "child" becomes a member of the $(n+1)^{th}$ population. If mutation occurs, a single bit in the child's chromosome is selected and flipped. If the bit had been a zero, it becomes a one and vice-versa. The mutation rate is also selected by the user. As Raczynski describes it, mutation is applied "to introduce traits into a population that otherwise would not exist... [s]ince reproduction only produces offspring that are based on the parents, if a certain value in the chromosome is not found anywhere in the parent population, then it will not be anywhere in the offspring."[158] A larger mutation rate will increase the genetic diversity of the population at the cost of a reduced convergence rate due to the randomness that is introduced. Common mutation rates are in the neighborhood of 3–5%.[69]

Before a GA-based approach could be applied, however, the objective function by which population members would be ranked had to be specified.

4.10.5 Defining the Objective of the Genetic Algorithm

When multiple responses are important, an overall objective function may be constructed, such as by a weighted sum.[29] This overall objective function allows each population member

to be described by a single "fitness" score that quantifies its desirability. In this case, a simple additive objective function was constructed:

$$ObjFunc = \sum_{i=1}^{N} w_i |C_{M,i}| \tag{17}$$

Here, w_i is the individual weighting function for flight condition i, and $|C_{M,i}|$ is the absolute value of the pitching moment coefficient at flight condition i. N is the total number of flight conditions being considered.

The goal was to promote cases with $|C_M| \leq 0.1$; once this was achieved it was not important to drive the pitching moment at that flight condition to be smaller. Toward this end, a conditional weighting was applied:

$$w_i = \begin{cases} 10 & \text{if } |C_{M,i}| > 0.1 \\ 1 & \text{otherwise} \end{cases} \tag{18}$$

As $|C_M|$ decreases, this weighting function results in large objective function reductions (i.e. improvements). This holds true until $|C_M|$ falls below the threshold of interest (0.1), after which any further reduction in that pitching moment produces only a fraction of the previous rewards. The weighting function was applied equally to every flight condition.

4.10.6 Tailoring a Genetic Algorithm for the Current Application

A custom GA was synthesized for this application. Most GAs retain a "memory" of only one or two populations. It was expected that any desirable population members will be retained in the active population, and indeed direct descent explicitly tries to make sure that the best population members appear in the next population. For this application, it was decided that it was not worthwhile to spend time re-evaluating a case that had previously been analyzed. Instead, a history of observed fitness scores was maintained.

Each time a population was analyzed, its members and their associated objective function scores would be added to this population history. This population history would then be sorted by objective function, and the five hundred cases with the best objective scores would

be pulled out as "breeding stock." These cases would then be subjected to crossover and mutation to produce the next population for analysis.

For every CFD analysis, there was a small but non-zero risk that the analysis would fail to produce a result. The geometric utility might not build the outer mold line properly, the flow solver might not converge adequately, or it might converge to a nonsensical result. Efforts were made to mitigate these risks, as described in Appendix B, but the chance of a failed case could not be entirely negated. This risk was compounded by the fact that each case was simulated for multiple flight conditions. If a case failed to converge to a plausible result at *any* flight condition, that case was omitted from the list of results. To avoid re-running such a case, a list of attempted cases was maintained, separate from the list of finished cases.

Thus, the GA implemented for this application proceeded as follows: First, the total set of observed results was sorted based on objective function value, and the best five hundred results were used as "breeding stock." Two members from this set were selected and their objective functions compared. The member with the better objective score was retained. A random number was generated on the range from 0 to 1, and if the value was less than the likelihood of crossover (70%), crossover occurred.

When crossover occurred, another population member was selected from the population, and "starting" and "stopping" locations on the chromosome were picked via random number generation. The genes between these two indices on parent A were transposed to the same locations on parent B, while the same genes on parent B were moved to parent A. This produced two new cases, each of which continued the process.

Whether or not crossover occurs, there was a chance for each case to experience mutation. For this application, a wide variety of results was more desirable than a rapid convergence of the method; additionally, although excessive mutation might in some applications introduce so much randomness that progress is impossible, the retention of every observed result mitigated this concern. A relatively large mutation rate of 10% was therefore used. A random number was generated on the range from 0 to 1 and compared to the mutation rate. If the random number was less than the specified mutation rate, mutation occurred and one

random bit in the chromosome was flipped.

After the mutation operator, the case was finished and ready to enter the new population. Before this happens, it was compared against the list of cases that had been attempted previously, as well as the cases that were already in the new population. If the case matched a pre-existing one, it was discarded; if not, it was added to the new population.

This process was repeated until the new population contained five hundred new, unique cases. All members of the new population were then added to the list of pre-existing cases to ensure that they would not be duplicated in a future population. Note that ordinarily, there would be some chance that an old population member could be added to the new population unchanged (27% chance, for the specified probability values). All such cases were rejected by this formulation, and only population members which experienced crossover and/or mutation would enter the new population. As a result, the proportion of "children" which experienced crossover or mutation was higher than 70% and 10%, respectively. In tests, these proportions were commonly closer to 95% and 13%. Once again, these crossover and mutation rates were significantly higher than the recommended values cited by Grefenstette.[69] However, Grefenstette assumes an algorithm that has no memory beyond the current population, and in particular warns that high mutation rates risk introducing so much noise that progress could be lost between generations. In the current application, each new generation was based on not only the generation immediately preceding it but the best results ever observed, so the risk of losing a good "bloodline" was not present.

4.10.7 Results of Genetic Algorithm Search

Each of the nine parameters was described using an eight-bit string, allowing 256 possible values for each parameter. The algorithm was initialized with a five hundred point space-filling design. Once those cases had been analyzed at each flight condition using Cart3D, the overall objective function defined by Equations 17 & 18 was used to calculate a score for each case. The cases were then subjected to the GA algorithm, which produced a set of five hundred new cases for analysis. A total of twenty-five batches were selected by genetic algorithm. Those batches plus the initial space-filling set totaled 13,000 cases, of which 1,470

had $|C_M| \leq 0.1$ at every flight condition, a success rate just over 11%. This was deemed a sufficient quantity of test data.

Now that a block of cases of interest was available, the sampling experiment could begin. The hypothesis being tested was that:

> **Contour-based sampling will balance the selection of cases with good performance and the reduction of prediction uncertainty in promising regions, identifying samples that efficiently improve surrogate accuracy for configurations with small aerodynamic moments.**

To test this hypothesis, a set of space-filling samples and a set of samples selected by contour-based sampling (CBS) were collected and surrogate models trained using each. The accuracy of the resulting models were then compared. The test would depend on ***prediction accuracy for cases of interest***. This test would quantify the prediction error using the cases of interest identified by the genetic algorithm as detailed above. This prediction error would be quantified using Root Mean Square Error.[85]

It should be emphasized that this hypothesis did *not* claim some absolute level of performance, but rather a relative improvement in performance. The expected result was that the CBS-based surrogates would produce smaller RMSE values for a given number of samples and/or equal RMSE values for a smaller number of samples. If the CBS-based surrogates were found to achieve this objective, this hypothesis would be considered to be supported.

Before contour-based sampling could be applied, an initial set of samples was required to create the initial surrogate models for candidate evaluation. It was decided that this initial set of samples would be used by both the space-filling and the adaptively-sampled approaches to eliminate the risk that one or the other method might benefit from a particularly lucky or unlucky sample distribution. A initial sample size of five hundred cases was selected as being large enough to roughly approximate the behavior of the response while still leaving ample room for improvement. All Kriging models in this experiment, both for space-filling and adaptively-sampled cases, would use quadratic fits and Gaussian correlation models.

In order to generate a small space-filling set of cases that could be augmented without

losing the space-filling properties, a nested Latin hypercube[152] was selected to generate the space-filling sample set.

4.10.8 Null Hypothesis: Space-Filling Samples

The nested hypercube for this effort contained multiple space-filling subsets ranging from 500 to 16,000 cases, progressively doubling in size. The maximum size was intended to be substantially more than the experiment would require to minimize the risk that additional space-filling cases would be needed. It was expected that this experiment would use Kriging models, which become unwieldy when applied to a pool of more than a few thousand points, so the null hypothesis would primarily depend on the hypercubes of size 500, 1,000, 2,000 and 4,000 cases. The remaining cases would be available if larger sample sizes became necessary. This was not considered likely, but the marginal cost for the extra capacity was negligible and would be a useful resource if experimentation revealed that more samples would be necessary. Of these prepared cases, only 4,000 were deemed necessary and analyzed.

Because these samples could be selected simultaneously without any knowledge of the response behavior, these analyses were comparatively simple to prepare. Parameter values for each case were passed to the PaceLab tool,[61, 142] which defined the vehicle outer mold line (OML) using a triangular mesh that could be interpreted by the flow simulation software. Because all cases for this set were selected in advance, the analysis required little oversight by the experimenter.

Although samples were quite easy to select, not every case ran successfully. Cases were excluded from the final data sets for a variety of reasons. These reasons included difficulties creating the surface mesh, a lack of convergence in the flow solver results, or very rarely convergence to nonsensical results such as negative drag. More information about these difficulties, as well as efforts to mitigate them, may be found in Appendix D. If a case did not produce well-behaved, numerically-converged results at *every* flight condition, that case was excluded from the final data set. As a result, roughly 70% of the space-filling cases were included in the final data set.

Once the analysis results for the common core of space-filling data were available,

contour-based sampling commenced.

4.10.9 Alternative Hypothesis: Contour-Based Sampling

The 500 space-filling points from the nested Latin hypercube were used as the initial data set for the contour-based sampling algorithm. Of those 500 cases, 370 converged successfully at all flight conditions. Before the sampling algorithm could be applied in earnest, a handful of technical decisions had to be made.

4.10.9.1 Rate of Model Updates

In the adaptive-sampling literature, it is typically assumed that after the most desirable sample is selected, that response value or values for that sample are available immediately. To facilitate this, researchers demonstrating adaptive sampling algorithms often either use analytical functions as responses or generate a large pool of data in advance.[113, 149, 159] In the present application, the aerodynamic analysis was being performed on remote computing systems, and the authentication procedures required to access those systems did not allow automated login or file-transfer without an authorized human in the loop.[200] This limitation made it impractical to analyze each sample as it was selected. As for pre-generated data, although that approach was used for the two-dimensional demonstrations of contour-based sampling, the present nine-dimensional space was deemed too large for full *a priori* sampling.

An alternative was sought to analyzing each case as it was identified. Mackman and Allen describe a modeling approach in which multiple samples are taken without updating the surrogate model.[112] This approach was adopted and modified slightly to reflect differences in the sampling objectives: Mackman and Allen seek samples in regions of response nonlinearity and regions of low sample density. Their decision to not update the surrogate model affects the calculation of nonlinearity but not sample density, and so even without updating their surrogate there is no risk that any sample would be placed too close to the others.

Contour-based sampling, on the other hand, used Kriging models to estimate the response and prediction variance at a given point; if a sample was not added to the model, the

estimate for prediction variance would not reflect the effects of that sample, and there was a risk that multiple samples would be selected in the same region – samples which might have been more effectively placed elsewhere. To avoid this, each sample *had* to be added to the Kriging model once selected so that later calculations would properly estimate prediction variance. There was a mild complication due to the fact that the actual response value at that point could not be known until after it was analyzed; this was addressed by estimating the response at that point using the current Kriging model, adding the point and its estimated response to the Kriging model, and updating the Kriging model once the actual response value was known. A set of cases selected between CFD analyses was referred to as a "batch."

This decision raised a new question: how many samples should be selected before they are uploaded for analysis? Whenever new analysis results were added to the data pool, surrogate models could be re-trained to reflect the most up-to-date information. Analyzing each sample immediately would mean that all subsequent sample selection would be based on the best information possible. Conversely, delaying sample analysis and using the current surrogate models as stand-ins meant that the surrogates would not be updated as often, and thus would not be as accurate as they might have been. As a result, later sample selections would be based on information that was not as accurate as it might have been.

A larger number of samples per batch would result in decreased human workload but might reduce the overall effectiveness of the sampling algorithm. For this experiment, it was decided that batches of 50 cases ought to be an effective compromise between human workload and losses due to imperfect information. In light of the 70% success rate observed with the space-filling cases, selecting 70 cases per batch was expected to result in 50 useful results. Each batch would therefore be comprised of 70 cases. Attention then turned to selecting values for the free parameters of the algorithm.

4.10.9.2 Setting Values for Algorithm Parameters

The sampling algorithm has various free parameters that can be adjusted to match the user's preference. These free parameters include the required Probability of Interest (POI),

the number of candidate points being evaluated in each round, and the number of test points used in the evaluations. These parameters interact to determine the behavior of the algorithm.

As mentioned in Section 4.3, a larger number of test points would give the algorithm a larger set of options from which to choose the next point, but would result in increased computational effort per selection. A larger number of test points would provide a more complete understanding of how each candidate would affect prediction variance throughout the space, but would also result in increased computational effort. A higher POI requirement would exclude more candidates from consideration, reducing the effort required to select a new point, but might handicap the algorithm's ability to explore the design space (as shown in Section 4.9). If the computing resources are fixed, it might be more useful to translate the required computational effort per sample selection into terms of time elapsed per sample selection. The processing times that are cited in this section were from tests using two parallel quad-core Intel 2.83 GHz processors and 4 gigabytes of RAM.

Each case was selected from an 800-point Latin hypercube of candidate points; the candidates were evaluated using a separate 1,200-point Latin hypercube. New hypercubes were generated each time a new point was selected, as suggested in the original paper by Picheny et al.[149] to reduce the risk that any important portion of the design space is neglected during the sampling process.

Within each batch, the required POI was varied. In particular, a more-restrictive required POI was used for early samples, encouraging the algorithm to prioritize the most promising regions, and this value was progressively reduced to allow a wider variety of candidates to be considered. This approach was intended to maximize the effectiveness of the batch as a whole. In Section 4.9 it was demonstrated that when the POI requirement is high, the algorithm tended to select cases closer to regions that it had already sampled.

4.10.9.3 Early Observation: Low POI Values

In the initial batches, the first 5 rounds were given a required POI of 10%. The required POI was then progressively reduced to a minimum of 1%. The per-case selection times ranged

from around 3 minutes for the highest POI to around 2 hours for the lowest POI. When the effects of these settings were investigated, it was found that even for repeated samplings, no candidates were found to have a POI value exceeding 9%.

A set of candidates was investigated in order to determine the reason for these low probability of interest scores. Recall that to calculate POI score for a candidate, all predicted responses for that candidate were considered. Based on the predicted response values and the prediction variance in each response, it was possible to calculate the likelihood that the actual response values for that candidate fell within the specified ranges of interest. This likelihood was calculated for each response, and the lowest likelihood value was the probability of interest for that candidate.

It was found that the Mach 0.3 pitching moment was the response least likely to fall within the region of interest for almost every candidate that was investigated. The candidate with the highest POI score (8.56%) was predicted to have a C_M at Mach 0.3, AoA 0° of -0.06, which is definitely within the range of interest. The estimated variance for that prediction, however, was 0.86 – very large compared to the range of interest, which had a width of 0.2. Thus, even though candidates were predicted to have attractive response values, the prediction confidence was so low that every candidate received low POI scores.

There were two possible interpretations for this behavior. The first interpretation was that the Kriging surrogate for Mach 0.3 had low confidence in the region of interest because it had not yet sampled those regions adequately yet, and candidates with higher POI scores could not be generated until more analysis was completed. The second interpretation was that the candidate points that were being analyzed were sampling the space too sparsely, and more-attractive candidates could be identified by a denser sampling.

To determine which interpretation was correct, the number of candidate points was increased from 800 to 3,000 for the fifth batch of samples. The number of test points remained 1,300. Once again, no candidates were identified to have POI scores above 10%. To avoid a four-fold increase in sample selection time, the POI requirements were increased until roughly the same number of candidates would meet the requirements. The number of candidates was briefly increased to 9,000, and again no candidates were found to have high

POI values.

Based on this evidence, it appeared that high prediction variance for the Mach 0.3 flight condition was driving the low POI scores, even for relatively dense distributions of samples. In universal Kriging models, the prediction variance is in part driven by the ability of the underlying linear or quadratic model used in the predictor to replicate the data points. When the underlying model captures response behavior well, the prediction variance will be low; when the observed data is not well-matched by the underlying model, the prediction variance will be high, indicating that the response is more likely to be far from the prediction of the underlying model.

Going back to Equation 4 on page 81, it should be noted that the process variance is an important factor in the estimation of prediction variance. This is particularly clear when it is known that $c(x)$ and C, the covariance vector and matrix, may also be written as the product of the process variance and the correlation vector and matrix, respectively. Thus, the prediction variance estimate is directly proportional to the process variance. This process variance is estimated by the DACE toolbox when fitting a model to each response. When the estimated process variance was investigated for the space-filling and CBS-based models, it was found that the average estimated process variance value for Mach 0.3 C_M models was 1.36, while for Mach 0.8 and 2.5 C_M models the values were 0.13 and 0.30, respectively. This indicated that the Kriging models for Mach 0.3 C_M were significantly more dependent on correlation effects to match the observed responses, rather than the underlying quadratic model, and thus only candidates very close to existing samples would have small prediction variances.

After these observations were made, it was accepted that increasing the number of candidates evaluated was not likely to result in more attractive candidates, and the number of candidate and test points were left at 3,000 and 1,300, respectively. Ultimately, 10 batches of adaptive sampling cases were selected and analyzed. The results were then fit with Kriging models and the predictive accuracy of those models quantified.

4.10.10 Evaluation of Accuracy

The various models would be evaluated using the cases of interest identified via genetic algorithm, as described in Section 4.10.6. This data set was comprised of 1,471 cases, each of which exhibited pitching moments within ± 0.1 at all 3 flight conditions. The hypothesis being tested was that:

> Contour-based sampling will balance the selection of cases with good performance and the reduction of prediction uncertainty in promising regions, identifying samples that efficiently improve surrogate accuracy for configurations with small aerodynamic moments.

In order to consider this hypothesis supported, then, the Kriging models based on samples selected by contour-based sampling should be more accurate – that is, should have less prediction error – than models based on space-filling cases.

To test this, Kriging models were created using the different sets of points. After ten rounds of contour-based sampling, the adaptive data set contained a total of 913 cases: the initial 500-point space-filling design, of which 363 or 73% met all convergence requirements, plus 550 adaptive points, or 79% of the selected cases. As for the nested Latin hypercube which served as the space-filling sampling, a number of training data sets were available. The first set which was larger than the largest adaptive set was the 2,000-point hypercube, of which 1,387 cases or 69% met all convergence criteria.

As a result, the space-filling data sets used to test this hypothesis were the 1,000- and 2,000-point hypercubes. The 500-point hypercube was shared by both methods as common ground. The adaptive data sets included the shared hypercube points, and each adaptive data set was added in turn to illustrate how each batch of samples affected prediction accuracy. Kriging models were built using each data set in turn, and then the predictive accuracy of every model was tested.

4.10.10.1 Using Root Mean Squared Error

To evaluate the accuracy of each surrogate model, it was used to predict the three response values (i.e. pitching moments) for every case of interest in the test set. The actual values

were then subtracted from the predicted values to obtain prediction errors. The prediction errors for each response were then used to calculate the Root Mean Squared Error, or RMSE, for that response using Equation 15.[85]

Figure 25 shows the RMSE scores for each combination of model and response. The models based on contour-based sampling (CBS) are represented with circular icons, while models based on the space-filling nested Latin hypercube (NLHC) samples are represented with square icons.

Figure 25a shows the RMSE values for the first flight condition, Mach 0.3 α 0°. It is clear that models based on CBS cases have improved predictive accuracy compared to the models based on NLHC cases. For an equal number of samples contour-based sampling could produce an RMSE 10% less than the 1,000-point NLHC training set. Alternatively, to achieve equivalent prediction accuracy for this response, contour-based sampling required only one batch of samples, producing an RMSE of 0.630 using 416 cases versus the space-filling RMSE of 0.643 using 710 cases. This would be a savings of 41% of the space-filling sample evaluations.

Note, however, that the last few CBS models exhibited a shallower slope than that of the NLHC models. If this trend were to hold constant, the 2,000-point NLHC model would out-perform a equally-sized CBS model. Twice in the early rounds – during the first and fourth rounds, specifically – the prediction accuracy increased relatively sharply. No such jumps occurred after the fourth round of sampling.

Interestingly, the fifth round of adaptive sampling was when both the number of candidates and the required POI were increased. This is suggestive, especially in light of the observation from Section 4.9 that the POI requirement sometimes causes the algorithm to stay close to known points rather than exploring the space. It is possible that in this case, the increased POI requirements which were intended to keep the sample selection time within acceptable limits had a secondary effect: those requirements may also have constrained the algorithm to only evaluate relatively conservative candidates, those which fell quite close to existing samples. These candidates had a high likelihood of falling within the region of interest, indicated by their high POI scores, but because they were close to known cases the

algorithm's understanding of response behavior did not progress very rapidly. This is an intriguing idea, and further study may be worthwhile.

In the meantime, Figure 25b shows the second flight condition, Mach 0.8 α 0°. In this case, the 1,000-case NLHC model was less accurate than the 500-case model. It would seem that, by attempting to model this set of training cases as accurately as possible, the resulting Kriging model actually did a *worse* job of predicting the region of interest. The 2,000-case NLHC model offered predictive accuracy that is better than both smaller NLHC models.

The CBS models showed an even faster rate of improvement than for the previous response. After ten rounds, the prediction RMSE score had improved by one-third compared to the original 500-point NLHC model. This is not to say the progress was smooth: the third CBS model actually had slightly worse prediction accuracy than the preceding model. This hiccup is quickly rectified though. Note that again, after the fourth batch of samples, the improvement in the CBS models was consistent but somewhat slow relative to the progress that was made by the earlier batches.

After the fourth batch of adaptive cases, the CBS models were more accurate than the 2,000-point NLHC model despite using only 556 samples compared to the 1,387 of the NLHC set. Put another way, equivalent accuracy – CBS RMSE of 0.463 compared to NLHC RMSE of 0.471 – was achieved using less than half the number of samples.

Lastly, Figure 25c illustrates the third flight condition, Mach 2.5 α 0°. The NLHC models exhibited smooth, steady improvement as the number of samples was increased. The CBS models, on the other hand, exhibited behavior that was somewhat less than intuitive. The very first batch of samples resulted in a markedly *worse* predictive accuracy, and it took 3 batches of samples until the predictive accuracy was as good as before contour-based sampling began. It is believed that the first batch of CBS cases, while demonstrably beneficial for the accuracy of the other models, resulted in a less-accurate model for the third response. It was gratifying to see that the algorithm was error-tolerant, rectifying the early inaccurate models and still offering better accuracy.

The space-filling models showed approximately linear improvement as more cases were included. The adaptive models showed what was roughly a linear trend with a steeper

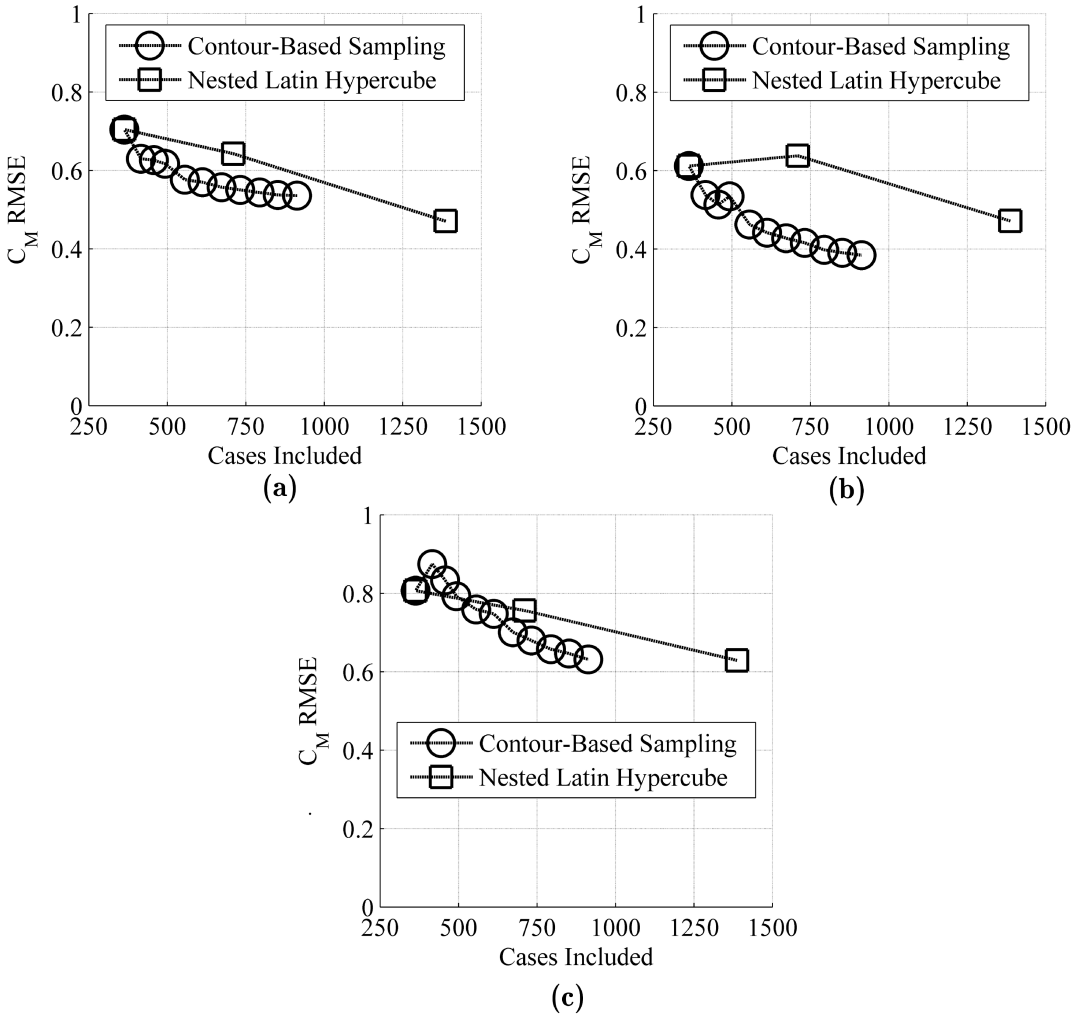

Figure 25: Prediction Accuracy for Cases of Interest: RMSEs at (a) Mach 0.3, α 15°; (b) Mach 0.8, α 0°; & (c) Mach 2.5, α 0°

slope, but some suggestion of becoming less steep in the later rounds. There was a slightly-larger-than-average improvement between the fifth and sixth batches. This indicated that increased POI requirements for these batches did not negate the chance of a breakthrough, although those chances might have been restricted somewhat.

4.10.10.2 Summarizing the RMSE Observations

> "That which is overdesigned, too highly specific, anticipates outcome; the anticipation of outcome guarantees, if not failure, the absence of grace." – William Gibson[64]

In general, it was seen that contour-based sampling did indeed produce surrogate models that had better predictive accuracy than those based on *a priori* space-filling sampling. For a given number of space-filling samples, contour-based sampling was shown to offer the user the choice between achieving the same performance (quantified here by prediction accuracy) using fewer samples, or achieving better performance using the same number of samples. In essence, the proposed sampling method created the opportunity for interplay between the observed responses and the samples that are selected, often producing a more effective sample distribution than could have been selected *a priori*.

The hypothesis that was being tested was that:

> **Contour-based sampling will balance the selection of cases with good performance and the reduction of prediction uncertainty in promising regions, identifying samples that efficiently improve surrogate accuracy for configurations with small aerodynamic moments.**

The preceding evidence has shown that contour-based sampling *did* in fact improve surrogate accuracy for configurations with small aerodynamic moments. In light of this evidence, Hypothesis 1 was considered to be *supported*.

CHAPTER V

EVALUATING MULTI-FIDELITY MODELING & UNCERTAINTY

The previous chapter evaluated the first supporting hypothesis, which addressed the process of selecting the most useful data points. The best way to leverage those data points has not yet been addressed. That is the purpose of this chapter.

Section 3.2 described a number of alternative multi-fidelity techniques which would allow the user to estimate a response value using multiple sources of data. These techniques included data harmonization, additive correction, proportional correction, and Ghoreyshi cokriging. Eventually the predictive accuracy of these techniques will be compared; first, though, each method needed to be implemented

5.1 Implementing the Methods Under Consideration

Most of the methods are fairly straightforward variations on a single-fidelity Kriging model once the low-fidelity response values are available. The most challenging aspect of their implementation was the modification of the DACE Kriging toolbox to incorporate user-defined uncertainty estimates in the form of nuggets.

5.1.1 Kriging With Nuggets

This description of Kriging with nuggets is based on the work of Gramacy & Lee.[66] To add a nugget to the Kriging mathematics (given in Section 4.2) is fairly straightforward. The covariance matrix C, which describes how the existing training data points relate to each other, is augmented by an extra term along its diagonal. Put more precisely, the entries of C are calculated as:

$$C^*\left(x_j, x_k | g\right) = C\left(x_j, x_k\right) + \delta_{j,k}\frac{g}{\sigma^2} \tag{19}$$

Here, $C\left(x_j, x_k\right)$ is the covariance between any two points x_j and x_k, calculated using a user-selected covariance function such as the Gaussian function (given in Equation 3 on

page 80). σ^2 is the process variance of the response, which – if not known in advance – is commonly estimated while fitting the Kriging model to the data. g is the nugget, an expression of the inherent uncertainty in the measured response values. This uncertainty may stem from measurement errors or "micro-variations" operating at scales too small to be captured by the available data, either of which would manifest as "white noise" in the response behavior.[67, 93] Note that g is always > 0 and has the same units as the process variance. $\delta_{j,k}$ is the Kronecker delta function,[165] which is given as:

$$\delta_{j,k} = \begin{cases} 1 & \text{if } j = k \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

The DACE Kriging toolbox already adds a small nugget, with a value on the order of machine precision, to the covariance matrix; see Section 5.1 in Lophaven et al.[108] for more details. This nugget is intended to regularize the system of equations and increase numerical stability for ill-conditioned problems. The DACE function *dacefit.m* contains the code which fits a Kriging model to data points. For reference, the nugget is added to the covariance matrix on lines 127–129 of this function.

This application of a nugget adds a constant value to every diagonal term of the covariance matrix. This is a common approach, especially when the nugget is intended to improve numerical stability (as in the DACE toolbox) or when measurement error is considered to be constant for every measurement.[93, 95, 182] However, it is also possible to apply a different nugget value to each diagonal term, as described by Yin et al.[197] Such an approach would capture any variances which were not constant throughout the design space, such as that stemming from the iterative solver of Cart3D.

Once the portion of DACE code which applied the stabilization nugget was identified, modifications were made so that other nugget values could be specified. The *dacefit* function was extended to allow the user to pass an extra vector of inputs which contained nugget values for each data point. This extension was relatively simple since, as previously noted, *dacefit* already generated the machine precision nugget as a vector and added it to the covariance matrix. The only change that needed to be made was to add the user-specified vector

to the existing vector of nuggets, and to modify the function input-output specifications so that the extra input parameter would not cause an error message.

Note that in Equation 19 the nugget is scaled by the process variance, which is unlikely to be known in advance. To obtain this value, DACE is first used to fit the model as if the data were noiseless. The resulting model includes an estimate for the process variance. The desired nugget may then be divided by the process variance so that it is in the format that the modified *dacefit* function expects. Casual tests indicated that the addition of a custom nugget in this manner does not significantly alter the estimated process variance value.

This modification was necessary to implement data harmonization; conveniently it was the *only* change necessary to allow the other multi-fidelity methods to incorporate uncertainty via nuggets as well. Due to their relatively simple formulations, these other methods will be described in detail before data harmonization is treated.

5.1.2 Additive Correction

An additive correction model is conceptually the simplest. One may be constructed by taking the difference between the high-fidelity (Y_{high}) and low-fidelity (Y_{low}) results for a set of training data (X):

$$Y_{diff} = Y_{high} - Y_{low} \tag{21}$$

A Kriging model would then be fit to this difference, Y_{diff}. To estimate the response of the high-fidelity code at some new point x, predictions are made for the low-fidelity response value and the value of the difference. These predictions are then added together to estimate the high-fidelity response:

$$\hat{Y_{high}} = Y_{low} + \hat{Y_{diff}} \tag{22}$$

The symbol $\hat{}$ denotes a value that is estimated rather than obtained from a data source.

5.1.2.1 *Using Surrogate Models for Low-Fidelity Data*

If the low-fidelity source of data runs quickly enough, it can be used directly to calculate the low-fidelity response whenever required. Typically this would include every point in the training data set, as well as every point for which a high-fidelity response must be predicted.

In the case of contour-based sampling, the category of points for which the high-fidelity response must be predicted would include all candidate and test points in every round. For large problems incorporating thousands of candidates and test points, unless the low-fidelity analysis runs exceedingly quickly – much faster than a second per case – it may be worthwhile to create a surrogate model of the low-fidelity analysis rather than waiting for the low-fidelity model to process all the necessary cases.

By replacing the low-fidelity data source with a surrogate, another source of uncertainty is introduced: the discrepancy between the response estimated by the surrogate and the response calculated by the low-fidelity data source. The process of estimating this uncertainty will depend on the type of surrogate model that is used. If a response surface or artificial neural network is used, the prediction variance may be estimated from the Model Representation Error (MRE) standard deviation, calculated as part of the goodness-of-fit checks.[36] Alternatively, a Kriging model of the low-fidelity response can analytically calculate the prediction variance at any given point. Either way, this uncertainty should then be included as a nugget when fitting a Kriging model to Y_{diff}, as that value depends on the low-fidelity prediction.

Uncertainty data pertaining to individual high-fidelity data points, such as solution convergence, would also be included in the nugget for the Y_{diff} Kriging model. Uncertainty due to model limitations, which would be considered aerodynamic *variances* in Section 3.5.3 on page 66, would *not* be included in the nugget. This is because additive correction is intended to produce a surrogate model that matches the response from a particular source – such as the pitching moment calculated by Cart3D – rather than trying to estimate the "true" response – such as the pitching moment measured by a full-scale flight test vehicle. This difference is crucial and will be revisited in later sections.

5.1.3 Proportional Correction

The proportional correction approach is also conceptually straightforward. These models are constructed by taking the *ratio* between the high-fidelity and low-fidelity results:

$$Y_{ratio} = \frac{Y_{high}}{Y_{low}} \tag{23}$$

A Kriging model is fit to this ratio, Y_{ratio}. To estimate the response of the high-fidelity code at some new point x, predictions are made for the low-fidelity response value and the ratio at that point. These predictions are then multiplied to produce a prediction for the high-fidelity response:

$$\hat{Y_{high}} = Y_{low} \times \hat{Y_{ratio}} \tag{24}$$

This approach is very similar to the additive corrector, but is better suited to problems where the discrepancy between the low- and high-fidelity response values are proportional to the magnitude of the response, rather than being a constant bias. It may not be possible to know in advance which formulation will result in better predictions, so some experimentation may be necessary.

5.1.4 Ghoreyshi Cokriging

Ghoreyshi cokriging is conceptually less intuitive than the two previous methods, but equally easy to implement. As described by Ghoreyshi et al.,[63] a Ghoreyshi cokriging model is almost identical to a standard Kriging model with the exception that the low-fidelity response value is treated like an extra input variable to the surrogate model for the high-fidelity response.

For example, consider a problem with two input variables, x_1 & x_2. The matrix of inputs F for a single-fidelity Kriging model with a linear underlying trend would be expressed as:

$$\begin{bmatrix} 1 & x_{1,1} & x_{2,1} \\ 1 & x_{1,2} & x_{2,2} \\ 1 & x_{1,3} & x_{2,3} \\ 1 & x_{1,4} & x_{2,4} \\ 1 & x_{1,5} & x_{2,5} \end{bmatrix} \tag{25}$$

In contrast, the matrix of inputs F for a Ghoreyshi cokriging model with a linear underlying trend would take the form:

$$\begin{bmatrix} 1 & x_{1,1} & x_{2,1} & \hat{Y_{low,1}} \\ 1 & x_{1,2} & x_{2,2} & \hat{Y_{low,2}} \\ 1 & x_{1,3} & x_{2,3} & \hat{Y_{low,3}} \\ 1 & x_{1,4} & x_{2,4} & \hat{Y_{low,4}} \\ 1 & x_{1,5} & x_{2,5} & \hat{Y_{low,5}} \end{bmatrix} \tag{26}$$

where $Y_{low,i}$ represents the low-fidelity response for point i. Thus, if the problem of interest has k input dimensions, the Ghoreyshi cokriging model will have $k+1$ input dimensions. The process of fitting the Kriging model (or any other form of surrogate model) is unchanged.

5.1.5 Data Harmonization

Data harmonization is unique among these methods for a number of reasons. First, while the other three methods fit separate surrogate models to each type of data (e.g., one surrogate model is fit to the low-fidelity data, and then a second surrogate is fit to the discrepancy between the low- and high-fidelity data), data harmonization accounts for all data simultaneously. This method is therefore more vulnerable to the increasing modeling costs of Kriging for large data sets. This vulnerability stems from its conceptual heritage, being

intended to capture the behavior of a single response as described by multiple observers.

Secondly, data harmonization has a different conceptual objective. The other three methods, by fitting each source of data separately, are explicitly attempting to model the response *as understood by that source of data*. Data harmonization, on the other hand, has some flexibility in this regard. The pre-analysis effort includes the subtraction of any known effects, as seen in Equation 32. In the gamma radiation example by Baume et al., the effect of elevation on the radiation measurements is removed from the data using an analytical relation before the Kriging model is trained.[14] Because this effect has been removed from the data, elevation is not included as a dimension in the Kriging model. It falls to the user to recognize any predictable effects and account for them.

5.1.5.1 Mathematical Formulation

Like in universal Kriging, the response at some point, $Z(s)$, is taken to be the sum of a mean function m and a stochastic residual δ:

$$Z(s) = m(s) + \delta(s) \tag{27}$$

The notation in this section is based on the work of Baume et al.[14] The mean function m is considered to be a combination of known effects and effects which must be estimated:

$$m(s) = F_a(s)\ a + F_\alpha(s)\ \alpha \tag{28}$$

Here, F_a represents the components for which the coefficients, a, are known, while F_α represents the components for which the coefficients, α, must be estimated. Categorical variables, such as country code in the demonstration given by Baume et al., are handled by adding binary column vectors to the set of input parameter values.[13, 15] For example, if the country code can take one of three values, three binary column vectors are introduced. Cases for which the country code takes the first value will have a one in the first binary column and zeros in the other two, and so on. Bias values that correspond to each categorical variable value may be known in advance, appearing in the model as part of $F_a(s)\ a$, or they may be estimated as part of the model fitting process, appearing as part of $F_\alpha(s)\ \alpha$.

Baume et al.[13] noted that the use of binary vectors for categorical variables results in an over-determined problem, which here would take the form of an ill-conditioned matrix of basis functions during the creation of a Kriging model (written as F in equation 2 in Section 4.2, not to be confused with F_α or F_a above). Two solutions are described by Baume et al.: either the least-squares coefficients for the binary vectors are subject to the additional constraint that they must sum to zero, or one bias is assumed to be zero and omitted from the model. This latter option is enacted by eliminating the binary column corresponding to the bias in question from the matrix of basis functions, F.

Note that the use of a binary vector for each option in a category is equivalent to making the assumption that data from each source will have a constant bias error. More complex relationships between data sources can be described if more basis functions are included for each categorical variable. A constant-bias data harmonization model for m parameters and p data sources would add p columns; a linear-bias data harmonization model, in which the discrepancy between data sources is a linear function of the input dimensions, would add $(m+1) \times p$ columns to the Kriging model.

Returning to Equation 27, the stochastic residual δ accounts for any variations from the mean trend. It is modeled as a random process with a mean of zero and a user-specified covariance function.[170] Often, the covariance function includes free parameters such as dimensional weights; we again assume that most or all of these parameters are unknown and must be estimated while fitting the surrogate model.

Baume et al. then go beyond the traditional Kriging formulation that is used for most applications by addressing the idea of uncertain data. Rather than assuming that response values are known exactly, the observations are treated as a combination of the true response at each point, $Z(s)$, and some measurement error term $\epsilon(s)$:

$$Y(s) = Z(s) + \epsilon(s) \tag{29}$$

This measurement error can be split into known biases, unknown biases (which must be estimated), and random measurement error or noise. These biases correspond to discrepancies between data sources. For our purposes, this can be written as:

$$\epsilon(s) = G_b(s)\ b + G_\beta(s)\ \beta + \zeta(s) \tag{30}$$

$G_b(s)$ is a vector of biases for which the coefficients, b, are known and fixed. $G_\beta(s)$ is a vector of measurement biases at location s for which the coefficients, β, must be estimated, while ζ is a random process representing the measurement error, which is assumed to have zero mean.

Lastly, Baume et al. assume that the stochastic residual δ and random measurement error ζ are normally distributed and mutually uncorrelated. The covariance matrices of δ and ζ for the given set of observations are denoted as

$$\begin{aligned} V &= Var(\delta) \\ W &= Var(\zeta) \end{aligned} \tag{31}$$

W is a diagonal covariance matrix of random measurement errors. Baume suggests that W be set by repeating a measurement multiple times using the device or method in question. Alternately, expert knowledge may be used to set one or more values in W. If the likely noise behavior of data points is not uniform for all data points, each entry in W may have a different value to reflect the noise at that point. Essentially, W acts as matrix of nugget values for each data point.

After any known mean or bias effects have been subtracted from the observations, the result is the expression:

$$U = Y - (F_a\ a + G_b\ b) = F_\alpha\ \alpha + G_\beta\ \beta + \delta + \zeta \tag{32}$$

The two random processes can be combined into a single process, ϕ, while the mean and bias functions can be combined into a vector of functions to obtain:

$$U = X\ \theta + \phi \tag{33}$$

where $X = [F_\alpha\ G_\beta]$ and $\theta = [\begin{smallmatrix}\alpha\\ \beta\end{smallmatrix}]$.

As a result, the expected value of the model (i.e., the Kriging mean) is given by $X\,\theta$ while the variance is $V + W$. Fitting the model requires the inversion of a covariance matrix with dimension equal to the total number of data points, similar to cokriging or the autoregressive model of Kennedy & O'Hagan. As was the case with those methods, fitting the model can become expensive or infeasible for large data sets.

As mentioned earlier, Baume et al. demonstrated their model using gamma ray dose measurements from multiple European countries. The elevation of the sample location was treated as a known bias using an analytical relation. Soil composition served as the unknown effect in F_α, and a country code was introduced as a bias term, G_β, to capture variations between sets of data provided by the contributing nations. The results of data harmonization demonstrated better agreement across national boundaries compared to regular Kriging, as well as increased prediction confidence.

5.1.5.2 Implementation of the Method

Baume et al.[13] suggest two possible ways to construct the data harmonization model in order to avoid an over-determined system of equations: either an additional constraint can be added to the least-squares bias estimates so that all biases sum to zero, or one of the biases can be assumed to be zero. The latter approach removes one of the binary columns from the matrix of samples. For the problem at hand, the source of each data point was thus represented in a single binary column, which took a one if the case in question was from Cart3D or a zero if the case was from APAS. This was equivalent to assuming that the Cart3D data had no bias, which was reasonable when the surrogate would be used to estimate Cart3D results.

As mentioned earlier, the single-binary-column formulation of data harmonization is equivalent to an assumption that the discrepancy between APAS and Cart3D is best represented by a constant offset. More complex representations of the discrepancy are possible, albeit at the expense of a larger number of supplemental columns. Consider a two-dimensional problem with inputs x_1 and x_2 and two data sources S_1 and S_2. If a linear trend is used and the discrepancy between the two data sources is treated as a constant value, fitting a Kriging

model will require least-squares calculations for four coefficients: one for the overall mean, one to weight x_1, one to weight x_2, and one to weight the binary column which captures the discrepancy between the two data sources. The matrix of basis functions would look like the following:

$$\begin{bmatrix} 1 & x_{1,1} & x_{2,1} & b_1 \\ 1 & x_{1,2} & x_{2,2} & b_2 \\ 1 & x_{1,3} & x_{2,3} & b_3 \\ 1 & x_{1,4} & x_{2,4} & b_4 \\ 1 & x_{1,5} & x_{2,5} & b_5 \end{bmatrix}$$

The b column contains the binary entries which indicate whether a given row is from data source S_1 or S_2.

If instead the discrepancy is considered to be linearly dependent on x_1 and x_2, the least-squares calculations must estimate values for six coefficients: one for the overall mean, one to weight x_1, one to weight x_2, and three to weight the discrepancy columns, which now take the following form:

$$\begin{bmatrix} 1 & x_{1,1} & x_{2,1} & b_1 & b_1 \times x_{1,1} & b_1 \times x_{2,1} \\ 1 & x_{1,2} & x_{2,2} & b_2 & b_2 \times x_{1,2} & b_2 \times x_{2,2} \\ 1 & x_{1,3} & x_{2,3} & b_3 & b_3 \times x_{1,3} & b_3 \times x_{2,3} \\ 1 & x_{1,4} & x_{2,4} & b_4 & b_4 \times x_{1,4} & b_4 \times x_{2,4} \\ 1 & x_{1,5} & x_{2,5} & b_5 & b_5 \times x_{1,5} & b_5 \times x_{2,5} \end{bmatrix}$$

One extra column is added for each dimension that will be incorporated into the discrepancy calculations. Recall that the bias for one data source was assumed to be 0; cases

from this source will have $b_i = 0$. As a result, cases from this source will have only zeros in the extra data harmonization columns:

$$\begin{bmatrix} 1 & x_{1,1} & x_{2,1} & 0 & 0 & 0 \\ 1 & x_{1,2} & x_{2,2} & 0 & 0 & 0 \\ 1 & x_{1,3} & x_{2,3} & 0 & 0 & 0 \\ 1 & x_{1,4} & x_{2,4} & 1 & x_{1,4} & x_{2,4} \\ 1 & x_{1,5} & x_{2,5} & 1 & x_{1,5} & x_{2,5} \end{bmatrix}$$

For comparison, when fitting an additive correction model, the first step is to model the data from the low-fidelity source. For this model, the matrix of basis functions is as follows:

$$\begin{bmatrix} 1 & x_{1,1} & x_{2,1} \\ 1 & x_{1,2} & x_{2,2} \\ 1 & x_{1,3} & x_{2,3} \end{bmatrix}$$

Once this model is available, its predicted values are subtracted from the response values from the high-fidelity source, and another model is trained to capture the linear trend in the high-fidelity data using a similar matrix of basis functions:

$$\begin{bmatrix} 1 & x_{1,4} & x_{2,4} \\ 1 & x_{1,5} & x_{2,5} \end{bmatrix}$$

These matrices of basis functions are used when estimating the various model parameters, such as the coefficients of the underlying trend model and the parameters of the correlation function (e.g. θ_i in equation 3). If the model parameters for the low-fidelity model and the correcting model were fit simultaneously, the matrices of basis functions would be combined:

$$\left[\begin{array}{ccc|ccc} 1 & x_{1,1} & x_{2,1} & 0 & 0 & 0 \\ 1 & x_{1,2} & x_{2,2} & 0 & 0 & 0 \\ 1 & x_{1,3} & x_{2,3} & 0 & 0 & 0 \\ \hline 1 & x_{1,4} & x_{2,4} & 1 & x_{1,4} & x_{2,4} \\ 1 & x_{1,5} & x_{2,5} & 1 & x_{1,5} & x_{2,5} \end{array}\right]$$

The terms in the upper-left correspond to a surrogate model that fits the low-fidelity data only, hence the zeros in the upper-right: high-fidelity data does not directly affect this process. Once these coefficients have been fit, the terms in the lower left capture the predicted low-fidelity results at the high-fidelity data sites. The discrepancies between the predicted low-fidelity results and the high-fidelity results are then modeled using the terms in the bottom right.

It now becomes more clear that when data harmonization is extended to capture more complex discrepancy behavior, the method begins to resemble additive correction. The two methods differ in that additive correction fits a model to one data source first (i.e. fitting a model only to the rows where $b = 0$) before fitting a separate model to the discrepancy values. In contrast, data harmonization fits all model parameters simultaneously, and thus must take all available data points into account at the same time. Huang et al.[83] and Kennedy & O'Hagan[95] both address the question of whether to fit these parameters simultaneously or separately; both conclude that fitting two separate models is significantly easier and "very little is lost in this simplification".[1] Effectively, additive correction is equivalent to a linear data harmonization model, although data harmonization fits all aspects of the surrogate simultaneously while additive correction fits them separately.

Data harmonization also explicitly captures uncertainty in the data by using nuggets.

[1] Curiously, both sets of authors express this sentiment in almost exactly the same words, although the text surrounding each statement varies greatly.

The modifications to DACE toolbox functions that were necessary to implement data harmonization would allow *any* Kriging model to capture uncertainty via nuggets; thus, once nuggets had been implemented for data harmonization, they were available for use by every other modeling approach.

Data harmonization is still unique among these methods in that it handles all sources of data simultaneously. A trial application of the method was developed in order to gain familiarity with its behavior. For the sake of simplicity, this trial was based on the two-dimensional data set that was used for early contour-based sampling tests in Section 4.9.

5.2 Two-Dimensional Test

The data set for the pitching moment coefficient at Mach 0.3, α 15° was used because the response had low levels of noise, but was still difficult to model using only Cart3D data and a linear or quadratic underlying trend. A response with linear or quadratic behavior could be modeled with relatively few samples; more complex behavior would require more data to improve the prediction accuracy of a surrogate model.

5.2.1 Selecting Samples

Relatively few high-fidelity data points would be used in this test in order to determine what improvement, if any, a multi-fidelity approach (and in particular data harmonization) could offer over the single-fidelity approach. For the results presented in this section, 6 high-fidelity samples were used; this test was repeated for as few as 4 and as many as 9 high-fidelity samples without qualitative changes in the results.

Those 6 high-fidelity samples would be selected using the Matlab function *lhsdesign*, part of the Statistics Toolbox.[120] This function generates a Latin hypercube for a user-specified number of samples and dimensions. These hypercube cases would be mapped to the design space. Recall that the space had been sampled with a 50×50 grid of Cart3D samples; the hypercube cases were replaced with the nearest sampled Cart3D case. Although this does affect the distribution of samples, the sampling resolution ensured that the replacement cases would differ from the hypercube cases by no more than 1% of the range of each variable. This was considered a negligible discrepancy.

Low-fidelity samples were selected from the available APAS results, which were also generated via a 50×50 grid of samples. The ratio of low-fidelity samples to high-fidelity samples was varied from 1 to 15, and so the total number of low-fidelity samples for the results depicted here ranged from 6 to 90.

The distribution of Latin hypercube samples is to some degree random, so the sampling process was repeated multiple times for each ratio value. Because the number of high-fidelity samples would be a constant in this test, the single-fidelity results were used to determine the number of repetitions necessary for the average results to converge to some average value. The single-fidelity results did not depend on the low-fidelity samples, so the number of repetitions was increased until the average prediction accuracy of the single-fidelity models remained essentially constant when the number of low-fidelity samples was varied. For the results presented here, the sampling was repeated 700 times for each data pool size.

5.2.2 Modeling Approaches

The baseline approach was to train the surrogate using only Cart3D samples, as was done in the motivating study. The assertion under consideration was that if a cheaper source of data such as APAS could be incorporated, the resulting surrogate models would have better predictive accuracy.

Although efforts were undertaken to ensure that the configurations analyzed by Cart3D and APAS were as similar as possible, certain aspects of the vehicle geometry were not represented in the APAS model. Whereas in the PaceLab environment (and consequently in Cart3D) the vertical tails at the wingtips could be oriented in three dimensions using a cant angle and toe-in angle, the vertical tails described to APAS were held to be vertical and aligned with the vehicle x-axis. Control surfaces generated by PaceLab could not be accurately reproduced for APAS due to the low granularity of the cross-sections used by the latter tool; the smallest control surfaces that could be represented in APAS might be many times larger than the same surfaces as defined in PaceLab. In light of this, control surface deflections were omitted entirely. Lastly, the body flap at the rear of the fuselage was also omitted.

The low-fidelity samples from APAS, although much less expensive than high-fidelity samples, might be considered something of an unfair "head start" for the multi-fidelity methods: the multi-fidelity methods will benefit from the extra computational effort, however slight, that went into generating the low-fidelity data. To address this, an additional single-fidelity approach was tested where the number of high-fidelity samples was increased from 6 to 7. Each high-fidelity analysis took on the order of 2,000 times the computational effort of the corresponding low-fidelity analysis, far more than the maximum amount of low-fidelity data used by the multi-fidelity methods, so one extra high-fidelity sample was more than equal to the extra computational investment in the low-fidelity samples.

Data harmonization was the most interesting modeling approach in this test, as it appeared in the literature only in the original author's publications. There was some uncertainty as to what (if any) "noise" should be captured. The model was intended to reproduce Cart3D data as precisely as possible and the iteration noise was negligible relative to the magnitude of the response, so the high-fidelity cases were considered to have zero noise. The low-fidelity data source was deterministic, with no obvious uncertainty in its results.

The low-fidelity samples would easily outnumber the high-fidelity samples, which led to some concern that the low-fidelity data might skew the data harmonization model in favor of matching APAS data rather than Cart3D. To test this, two data harmonization models were created. The first treated both the high- and low-fidelity samples to be deterministic, with no noise or uncertainty. The second applied a constant nugget, equal to 5% of the range of the observed low-fidelity response values, to all low-fidelity cases. This meant that the data harmonization model would interpolate every high-fidelity sample exactly without having to do the same for the low-fidelity samples.

Additive correction, as arguably the most popular multi-fidelity method described in the literature, would also be applied as a further point of comparison to determine the effectiveness of data harmonization. The predictive accuracy of each modeling approach would be evaluated using Root Mean Squared Error, as defined in Equation 15 on page 100.

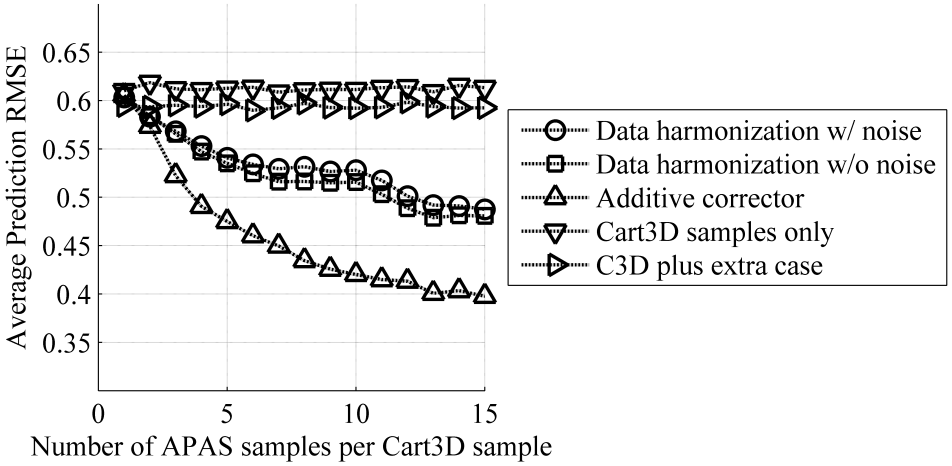

Figure 26: Prediction RMSE for All Multi-Fidelity Methods at Mach 3, AoA 15°

5.2.3 Evaluating the Results

The results of this study are depicted in Figure 26. The triangular icons pointing down and those pointing to the right, which represented the single-fidelity models with and without an extra case respectively, maintained a consistent RMSE value as the number of low-fidelity samples was varied. This served as confirmation that sufficient repetitions were performed and the results were adequate representations of the average performance of each method.

The two data harmonization approaches – one including a nugget for low-fidelity data and one treating all data as deterministic – track each other closely, which indicates that the low-fidelity nugget did not significantly affect prediction accuracy for high-fidelity response. In fact, the data harmonization models with noise often performed slightly *worse* than the noiseless models, which suggested that the quantity of noise included (equal to 5% of the observed range of the low-fidelity data) was excessive for this problem, acting to "wash out" useful information rather than downplaying misleading effects.

The data harmonization models demonstrated improved predictive accuracy compared to the single-fidelity methods; the prediction error as quantified by RMSE was reduced by 10–20% when sufficient low-fidelity data was available. However, data harmonization was out-performed by the additive correction model, which tended to offer roughly twice as much

improvement in prediction accuracy.

These results confirmed that multi-fidelity modeling could enhance prediction accuracy relative to a single-fidelity approach. Although additive correction out-performed data harmonization in this case, the relatively simple nature of the test did not allow either method to be declared "best" for the large-scale problem. Additional testing with a more complex data set was required. Furthermore, there still existed some question as to which sources of uncertainty were useful to capture and which could be neglected. The next section is a review of the sources of uncertainty under consideration.

5.3 Selecting Sources of Uncertainty

Numerous uncertainty sources have been identified and detailed in the literature.[43, 95, 186, 202] A full description of every source of uncertainty in computational analysis is far beyond the scope of this work. Entire books could be written on the subject of estimating the uncertainty in a computational result – and have been.[138] Instead, a few uncertainty sources were identified as being of particular interest based on observations made during the research effort described in Chapter 2. These sources are:

- Uncertainty resulting from finite precision and discretization during analysis, such as measurement or numerical iteration;[163]
- Uncertainty resulting from the use of a surrogate model rather than the original analysis, i.e. imperfect emulation;[95] and,
- Uncertainty due to use of a data source which does not perfectly emulate the response, i.e. imperfect fidelity.[44]

The first source of uncertainty, discretization and iteration effects, was directly observed during the motivating effort (see Figure 7 on page 42). Cart3D uses an iterative simulation process to converge to a solution. Responses such as the force and moment coefficients were calculated for every iteration step and written to an output file, and thus the convergence history of each response can be investigated. It was found that most cases exhibited iteration noise which was on the rough order of 0.001 to 0.01, which could be considered negligible

relative to the observed pitching moment coefficients (which were often on the rough order of 1 or 10) but that same noise was significant relative to the observed rolling and yawing moment coefficients (which were on the rough order of 0.001 to 0.1). It is plausible that capturing this iteration noise during the training of surrogate models might improve prediction accuracy for lateral responses.[109]

The second source of the uncertainty, imperfect emulation, is introduced when a data source is replaced by a surrogate model. It was mentioned in Section 3.5.3 that, despite the relatively rapid execution time of APAS, a second or two per case is still too long when thousands of candidate and test points must be evaluated by the contour-based sampling algorithm before a sample can be selected. A surrogate model could estimate low-fidelity response values at those points more quickly, significantly reducing the time required to select each sample, but would introduce additional uncertainty due to small prediction errors.[202] If this surrogate is Kriging-based, the prediction uncertainty can be calculated analytically; if another surrogate model type is used, the standard deviation of the Model Representation Error can be used to calculate the prediction variance. Model Representation Error is calculated as part of the "Goodness of Fit" tests.[134]

The third source of uncertainty, imperfect model fidelity, is introduced whenever modeling is used. This occurs for all experiments short of full-scale testing under operational conditions, as evidenced by the Lockheed C-141 Starlifter mentioned in Section 1.5.[23] Although that effort included wind tunnel testing of the wing design at appropriate Mach numbers, there was sufficient discrepancy between the viscous flow behavior over the subscale test article and the full-scale vehicle that the designers were forced to modify the control schedules and reinforce the wing and fuselage to avoid exceeding the structural limits of the design.

For this research effort, different surrogate models would be used at each flight condition. The independent variables for each surrogate model would be geometric parameters, unlike other efforts which fixed the geometric shape of the vehicle and varied the flight condition.[63] Given the low probability that multiple sources of validation data will fall within any given design space (described in Section 3.5.3), it was unlikely that enough information would be

available to describe how the uncertainty due to imperfect fidelity would vary with respect to the free parameters. Instead, this uncertainty would be modeled as a uniform value for each flight condition.

Two hurdles remained before uncertainty could be integrated into this modeling framework. First, methods must be available to quantify or estimate the contribution to overall uncertainty that stems from each source. And secondly, although uncertainty could be captured by a Kriging model via the nugget parameter, and uncertainty present in Kriging model predictions could be estimated via prediction variance, it had yet to be demonstrated that the uncertainty defined via nugget is preserved and represented in variance estimates.

5.3.1 Quantification of Uncertainty

Of the three sources of uncertainty considered in this research – uncertainty due to discretization, uncertainty due to imperfect emulation, and uncertainty due to imperfect fidelity – two could be quantified in a straightforward manner. The scripts used to extract results from Cart3D output files were modified to extract response values over the final 30 solver iterations. The average and standard deviation of the series were calculated for each response and reported as part of the solution data set. This provided a quantitative assessment of the uncertainty due to discretization.

Uncertainty due to imperfect emulation is estimated once a surrogate model has been trained, during the Model Representation Error test.[36] This test uses the surrogate model to predict the response at a set of test points for which the true response is known. Thus, if a surrogate model has been subjected to Goodness of Fit tests, an estimate of the uncertainty due to imperfect emulation should already be in hand.

Only uncertainty due to imperfect fidelity remained. As discussed in Section 3.5, the literature indicates that this uncertainty is quantified similar to the process of validating a tool: the analysis tool is applied to one or more analyses for which high-fidelity results are available. Sponsors at the Air Force Research Laboratory provided wind tunnel data and surface meshes for three variants of the XCOR Lynx suborbital vehicle.[127, 177] One variant of this vehicle is displayed in Figure 27.

Figure 27: XCOR Lynx Suborbital Vehicle Variant

The data set included force and moment measurements taken at various flight conditions. The reported data also included measurements of factors such as local pressure and temperature, factors which are not of importance when running an inviscid tool such as Cart3D but would be required information if this data set were used for validation of a viscous tool. If such data were not recorded, the analyst would have to select a value for those parameters; values might be chosen to maximize agreement with the experimental data, a process closer to **calibration** than validation and one that is strongly advised against by validation experts.[140] Instead, the analysis should be done in ignorance of the high-fidelity result values whenever possible to avoid accidental or deliberate calibration.

The wind tunnel data set was parsed to match each surface mesh with the flight conditions for which data was available. Mach number varied from 0.29 to 4.5 and angle of attack varied from -1.7° to 43.5°. Sideslip angle was set to 0° for most cases, but 20 of the 71 parameter sweeps fixed the Mach number & angle of attack and varied sideslip angle. The flight conditions in the data set are plotted as angle of attack versus Mach number in Figure 28. Some sweeps re-sampled a previous range; it is thought that these repetitions were intended to estimate the variability in the results as recommended by Aeschliman & Oberkampf.[3]

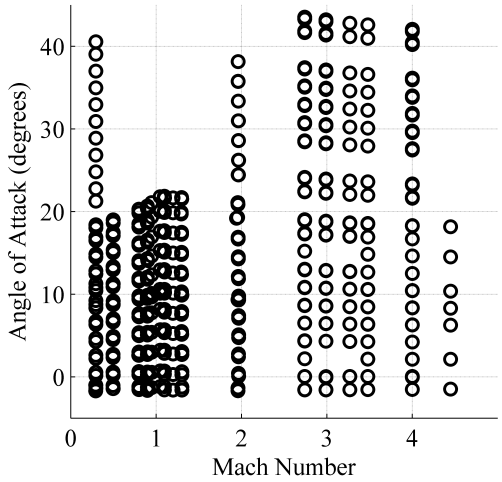

Figure 28: Flight Conditions for XCOR Data

In all, 632 data points were available for the surface meshes that were obtained.

Cart3D was used to analyze each vehicle variant at flight conditions which matched the wind tunnel data. Note that some model comparison efforts prefer to match lift coefficient rather than angle of attack.[102] Such an approach was rejected here due to the effort required to match the observed lift coefficient for each of the 632 data points. Instead, the vehicle variants were assessed using a once-through approach identical to that used for all other Cart3D analyses in this effort. The reference wing area, mean aerodynamic chord, and wing span were obtained by scaling the values recorded in the wind tunnel data set to match the dimensions of the surface triangulations; the best match between the dimensions specified for the wind tunnel model and the observed dimensions in the triangulation was a factor of 63, so the reference lengths were multiplied by 63 and the reference area by 63^2. A center of gravity at 70% of the vehicle's length was assumed.

It should be noted that decisions on the part of the analyst concerning appropriate parameter settings, such as grid resolution or turbulence model, can have a very significant effect on the analysis results, at times even larger than the selection of the analysis tool itself.[80] Any discrepancies between wind tunnel measurements and Cart3D calculations that are presented in this document should not be construed as inherent limitations of Cart3D; it is quite possible that the tool was applied in a sub-optimal manner by the

author, and that better agreement with the data might be achieved by applying Cart3D in a different manner.

With that caveat given, the Cart3D results were compared against the wind tunnel data. The results presented here will emphasize pitching moment coefficient, as this was the response that most significantly restricted the feasible design space in the RBS study. As in the X–33 uncertainty database, discrepancies between code prediction and experimental measurement were plotted versus Mach number.[32] The discrepancies were calculated as:

$$\delta = Y_{Cart3D} - Y_{WindTunnel} \tag{34}$$

When the discrepancies are calculated in this manner, an overly-positive prediction by Cart3D results in a positive discrepancy. These discrepancies are plotted in Figure 29. Figure 29a shows the data for Mach numbers less than 0.6, Figure 29b shows results between Mach 0.6 and 1.35, and Figure 29c shows the data for Mach numbers above 1.35.

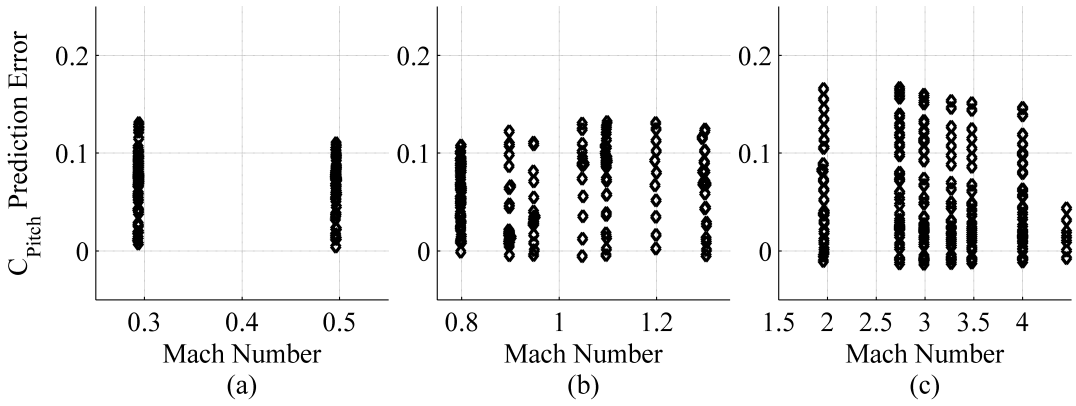

Figure 29: Comparison of Cart3D Results to Wind Tunnel Data, Grouped by Mach Number

The Mach numbers analyzed are clearly identifiable, but with respect to the magnitude of the discrepancy, the icons are more or less continuous. This may not be surprising given the relatively fine resolution of the experimental plan shown in Figure 28. Although the X–33 uncertainty database was grouped by Mach number, for these results that grouping did not offer any insights into the accuracy of Cart3D.

No obvious trends were discernible when the discrepancies were plotted against angle of attack, aside from a general upward trend with increasing angle of attack. It was found that

after the results were grouped by Mach number as in Figure 29, clear trends with respect to angle of attack could be seen within the group. Those results are visible in Figure 30.

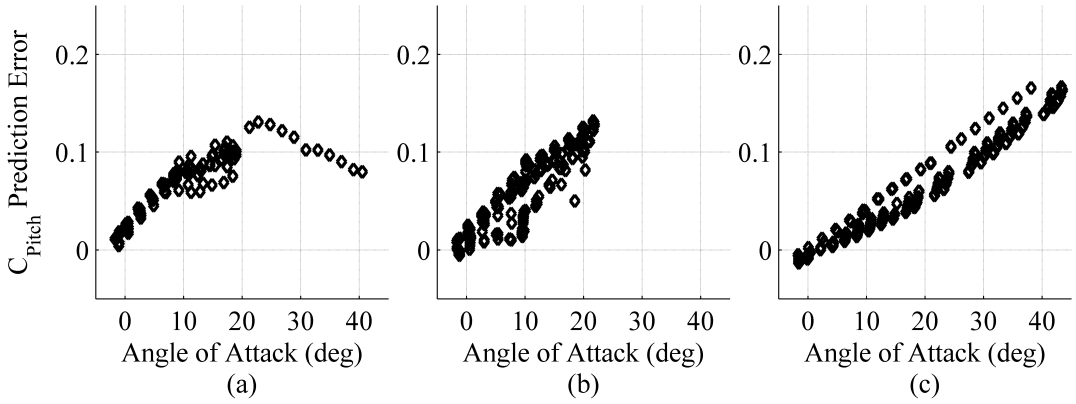

Figure 30: Comparison of Cart3D Results to Wind Tunnel Data, Grouped by AoA for (a) Mach < 0.6; (b) $0.6 <$ Mach < 1.35; (c) Mach > 1.35

In Figure 30a, which shows results for Mach < 0.6, the discrepancies show a fairly clear trend: they are small close to 0° angle of attack and grow larger as the angle of attack increases to 20°. This trend appears to reverse itself for higher angles, but since only one sweep was performed for angles of attack greater than 20° in this speed regime, the author is hesitant to generalize from that data.

Figure 30b shows results for Mach numbers between 0.6 and 1.35. Here the trend resembles what was observed in Figure 30a, with discrepancies increasing for larger angles. These results exhibit more spread, even at lower angles of attack, which is not surprising given the known limitations of Euler methods at transonic speeds.[7]

Finally, Figure 30c shows the results for Mach numbers greater than 1.35. At these speeds, the discrepancies at 10° are less than for the subsonic data set, but the discrepancies for supersonic flight conditions steadily grow with angle of attack. For a given angle of attack, a higher Mach number corresponds to a smaller discrepancy.

In every speed regime, the discrepancies between Cart3D and wind tunnel results increased with angle of attack. This indicated that, for larger angles of attack, Cart3D would

over-predict the pitching moment by progressively larger amounts. It may be that the effective center of mass of the wind tunnel model was closer to the nose of the vehicle than the point at 70% of body length that was used for Cart3D calculations. The 70% position was based on the best available information; barring additional information, the remainder of this work will assume that the results of this section are representative of the discrepancy between Cart3D estimates produced by the author and experimental data.

These results could be used to estimate the likely bias and random errors in Cart3D predictions that were due to model fidelity limitations. The bias error would be estimated as the average prediction error. The predictions would then be corrected by this bias estimate, and the new prediction errors calculated. The variance of the new prediction error could be used as an estimate of the random error variance. These two estimates, along with the estimates for uncertainty due to iteration noise and imperfect fidelity, would enable the user to compute the effects of all three uncertainty sources relevant to this effort.

5.3.2 Categorizing Sources of Uncertainty

Note that not all the uncertainty sources have the same implications with respect to the data. The first two sources, uncertainty due to discretization and uncertainty due to imperfect emulation, pertain to the *simulated* response. They describe uncertainty with respect to the response at the current level of fidelity (e.g. the response as calculated by APAS or Cart3D).

Uncertainty due to model fidelity limitations, on the other hand, describe the difference between the response *produced by the simulation* – e.g., pitching moment as calculated by Cart3D – and the response *being simulated.* By capturing the first two uncertainty sources, surrogate models will be more accurate with regard to the training data. If the last uncertainty source can also be captured effectively, the uncertainty predictions of the resulting surrogate model might be used to estimate the accuracy of the surrogate compared to the data used to estimate model limitation error.

Using the validation results, the uncertainty due to model fidelity limitations can be estimated. The estimated values can be added to the nugget when training the Kriging surrogate. However, because uncertainty due to inadequate fidelity reflects the probable

discrepancy between that level of fidelity and the "true" value of the response being simulated, it is imperative that this quantity be preserved.

Although it was demonstrated that uncertainty information can be passed *into* a Kriging model via the nugget, and all Kriging models are capable of estimating the predictive variance for the predictions that come *out* of the model, it had not yet been demonstrated that uncertainty information is *preserved* during this process. Preservation of this information is required if uncertainty due to model fidelity limitations is to be captured in this manner. A small-scale test was therefore devised to determine whether this information is preserved.

5.3.3 Preservation of Uncertainty In Kriging Models

A set of "truth" data was derived from wind tunnel tests for a reusable booster configuration published by Post et al.[150] The 16-case data set corresponding to the 78% center-of-gravity location was selected. This selection was arbitrary. The exact shape of the vehicle was not available, precluding analysis with other tools. Instead, a notional lower-fidelity data set was created using only the points with small angles of attack where the relationship between angle of attack and pitching moment was approximately linear. The rate of change of pitching moment in this region was approximately -0.01 per degree. Pamadi et al.[144] showed that for the Langley Glide Back booster, the low-fidelity tool estimated this slope fairly well but failed to capture nonlinear effects at higher angles of attack. Bias offsets were also sometimes present in the low-fidelity data. This behavior is illustrated in Figure 6 on page 39. Here, a bias offset of -0.04 was added for visual clarity.

The uncertainty due to simplifying assumptions – in this case, linear aerodynamics – could not be estimated in a general form based on this information. Instead, a very rough estimate of uncertainty was made: the discrepancy between the linear results and the wind tunnel results was used as the standard deviation of this uncertainty. If the error introduced by use of a linear method is normally distributed, the true result would fall within ± 1 standard deviation, or 1σ, of the linear estimate around 68% of the time. A $\pm 2\sigma$ range would encompass the true result roughly 95% of the time. Figure 31 shows the relative distribution

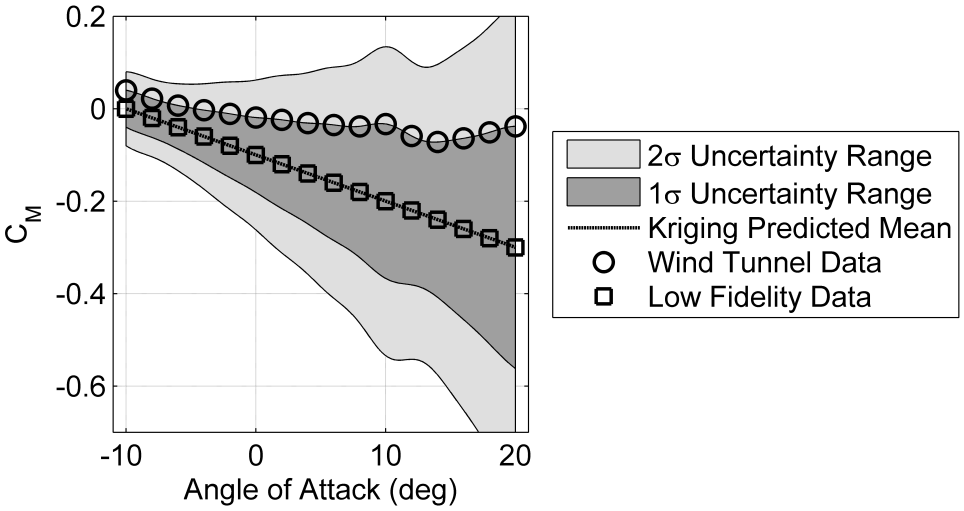

Figure 31: Data Points & Defined Uncertainty Ranges

of wind tunnel results, low-fidelity linear results, and uncertainty bounds around the low-fidelity data points. The uncertainty bounds were interpolated for ease of representation. The uncertainty ranges, particularly the $\pm 2\sigma$ bounds, enclosed a broad swath of possible values. This reflected the potentially large discrepancies present at higher angles of attack where the linear approximation was the least accurate.

A Kriging model was trained using the 16 low-fidelity data points. A linear underlying trend was used in the Kriging model, and the standard deviation at each data point was squared to produce variances. These variances were incorporated into the Kriging model via nuggets. This Kriging model was then used to estimate the response and prediction variance for 1,000 points interpolating the existing data set in order to assess how well the new surrogate could reproduce the uncertainty that was described using nuggets. The prediction variance values were converted back to standard deviation for plotting purposes. The data points, $\pm 1\sigma$ bounds, and $\pm 2\sigma$ bounds are plotted in Figure 32, just as they were in Figure 31.

It was immediately apparent that the uncertainty bounds that were obvious in Figure 31 were not visible in Figure 32. The presence of the uncertainty ranges on the legend indicated that the bounds are being plotted. A closer inspection of the black line indicating the predicted response values, seen in Figure 33 held the answer. Figure 33 is a highly-zoomed

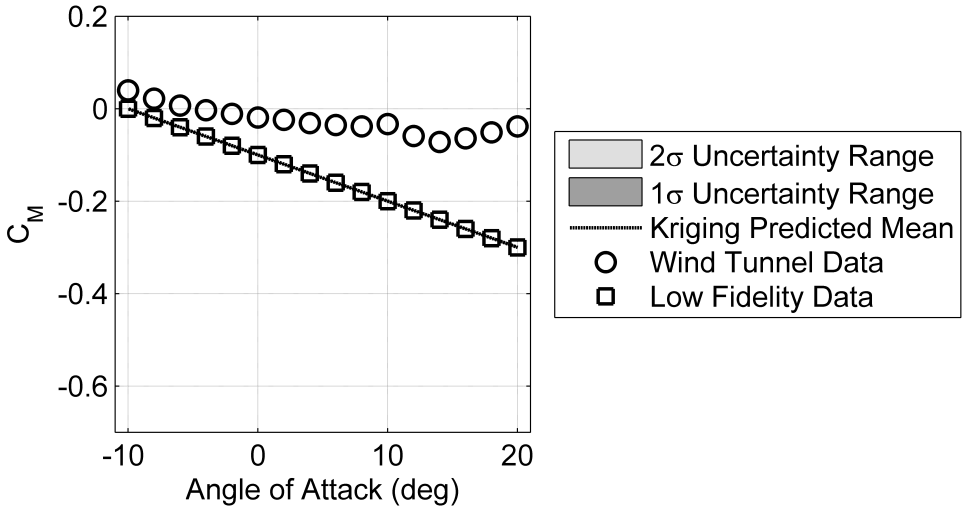

Figure 32: Data Points & Estimated Uncertainty Ranges

view of Figure 32. The $\pm 1\sigma$ and $\pm 2\sigma$ bounds *were* present in Figure 32, but were multiple orders of magnitude smaller than the uncertainty which was described using nuggets.

This behavior was explained by referring back to the mathematical formulation of Kriging given in Section 4.2 on page 80. In particular, note that the estimated process variance appears both explicitly and implicitly in the equation for prediction variance estimation (Equation 4); remember that the covariance function is the product of the process variance and the correlation function. The estimated process variance (Equation 5) depends on the quantity $\left(Y - F\hat{\beta}\right)$, which is the difference between each observed response and the value of the underlying trend at that point.

From these equations it was inferred that, when the observed response behavior is approximated well by the trend, the estimated process variance (and by extension the prediction variance) would be close to zero, no matter how much uncertainty was present in the original data set. As a result, uncertainty information that is fed into a Kriging model may not be preserved in the predictions of that Kriging model.

This loss of information is of little concern when it comes to the effects of finite precision or surrogate model imperfections; both of those uncertainties relate only to the quantity being estimated, i.e. pitching moment *as calculated by Cart3D* (in this case). Uncertainty due to model fidelity limitations, on the other hand, describes the possible discrepancies

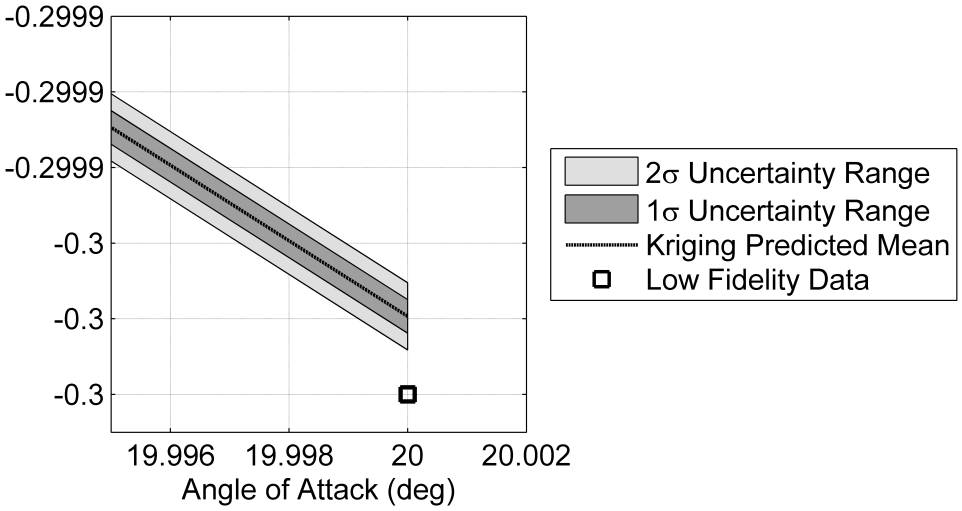

Figure 33: Zoomed Region of Data Points & Estimated Uncertainty Ranges

between the quantity being simulated (e.g., pitching moment as estimated by Cart3D) and the actual quantity of interest (e.g., pitching moment experienced by a full-scale vehicle at representative conditions).

As with the Space Shuttle and X–33, knowledge of the uncertainty due to model fidelity limitations could be used to identify where further analysis is needed, and – if the present data set had inherent fidelity limitations – where a better analysis method would be needed as well. Such knowledge could be critical for risk-reduction operations.

More to the point, the *loss* of such information could lead to dangerous over-confidence on the part of the designers, who might think that the Kriging prediction variance was accurately capturing *all relevant sources of uncertainty*. To avoid this situation, it is recommended that uncertainty due to model limitations *not* be included during Kriging modeling, and instead that such information be applied as a sort of "error bar" after the modeling is completed.

The two remaining sources of uncertainty still had to be assessed to determine their effect on prediction accuracy. In addition to determining which multi-fidelity method would produce the best prediction accuracy, different combinations of uncertainty sources would be investigated to determine the effect on model accuracy. Multiple versions of each multi-fidelity method would be applied to the same set of data, differentiated by the uncertainty

information being captured via nugget:

- No uncertainty (deterministic results);
- Only uncertainty due to surrogate model prediction error;
- Only uncertainty due to solver iteration effects; or,
- Both surrogate prediction error and solver iteration effects.

The results would determine which types of uncertainty should be incorporated and which multi-fidelity method would be selected for this application.

5.4 Comparing Prediction Accuracy: Pitching Moment

Unlike the tests of contour based sampling, tests of multi-fidelity methods would not emphasize any particular response range. Instead, prediction accuracy would be quantified using the Root Mean Squared Error[85] of the predictions and the 95% confidence quantiles of the prediction error.[105]

Prior work[40] showed that the lateral responses exhibited much more iteration noise at 40° angle of attack than at 0° angle of attack. Both angles were of interest for the present application, so data from both angles of attack were used to evaluate the multi-fidelity methods and uncertainty sources. It was expected that, due to the increased iteration noise at high angles of attack, models which captured this iteration noise would have better prediction accuracy at that flight condition than models which did not.

Data was generated at Mach 4.0, α 0°, β 0° and Mach 4.0, α 40°, β 0°. Cases from the nine-dimensional space-filling nested Latin hypercube, described in Section 4.10, were analyzed at these new flight conditions to see how the methods fared when applied to a problem with a moderately large design space.

Of the space-filling nested Latin hypercube cases, 6,000 were evaluated. This completed the 4,000-case level of the NLHC, plus half of the cases needed to fill the 8,000-case level of the NLHC. Although the partial data set could not be guaranteed to be space-filling, this was not expected to affect the present test. The objective was to evaluate the relative accuracy

of each method, and thus the supporting data pool need not be perfectly space-filling to be of service.

Five alternatives were tested:

- Mono-fidelity data (only Cart3D cases);
- Additive correction;
- Proportional correction;
- Ghoreyshi cokriging; and,
- Data harmonization.

Of these methods, all but the first required low-fidelity data. APAS acted as the low-fidelity data source for this research effort. In the interest of expediency, surrogate models were created to estimate aerodynamic response values more quickly than if APAS were run directly.

5.4.1 Surrogate Models of Low-Fidelity Data

The 16,000-case nested Latin hypercube described in Section 4.10 was analyzed with APAS at Mach 4, α 0° and Mach 4, α 40°. Neural networks were chosen as the surrogate modeling technique for this demonstration due to the ability of neural networks to incorporate many thousands of cases during the training process,[74] whereas a Kriging model attempting to include the same number of cases was likely to be intractable due to memory limitations and excessive computational requirements.

BRAINN, which stands for "Basic Regression Analysis for Integrated Neural Networks," is a software utility used to train the neural networks and an internal tool at the Aerospace Systems Design Laboratory developed by Carl Johnson and Jeff Schutte.[20] BRAINN interfaces with utilities from Matlab's Neural Network Toolbox and automates the process of fitting neural networks of various sizes to the data. The user specifies a minimum and maximum network size to consider, as well as the error definition, and the software will cycle

through those network sizes and retain the network that best fits the data without over-fitting. Over-fitting occurs when the model being trained becomes progressively better at reproducing the training data set while in turn becoming progressively *worse* at estimating the response at other data points.[74] The surrogate models of the pitching moments were both of good quality as determined by goodness-of-fit checks.[36]

BRAINN includes the option to export the best-performing neural network in a format that can be easily interpreted by Matlab. Those exported files were adapted into functions so that any other Matlab script or function could submit the values of the independent variables to the function and receive estimated response values in return. The functions were vectorized so that multiple predictions could be performed simultaneously.[122]

Once the surrogate models were accessible by Matlab, the tests could begin.

5.4.2 Execution of Analysis

For this analysis, certain factors were held constant. The Kriging models used anisotropic Gaussian correlation functions and linear underlying trends. The first 6,000 of the 16,000-case nested Latin hypercube cases had been analyzed previously and sorted according to flight condition. The appropriate data set was loaded and divided into two sets. The cases which made up the 4,000-case hypercube became the first set, as this set of cases was known to be space-filling. The remaining cases went into the second set.

Some number of cases would be randomly selected from the first set using the function *randsample* from the Matlab Statistics Toolbox and, based on those cases, one surrogate model would be trained using each multi-fidelity technique. The surrogates would then be used to predict response values at all the cases in the second set; the error of these predictions would be quantified via RMSE.

The number of cases selected varied from 100 to 1,000 cases in increments of 100 to assess how rapidly the accuracy of each method improved when more training data was available. Most multi-fidelity methods needed to have both the high- and low-fidelity response for each case in the training set; the high-fidelity response came from Cart3D analysis, while the low-fidelity response was estimated with the neural networks trained to replicate APAS

results.

Because data harmonization fits a model to both high- and low-fidelity data simultaneously, the cases selected as described above were used as the high-fidelity samples, while a separate set of 1,000 random cases were randomly selected in the same manner for use as low-fidelity samples. The constant size of the low-fidelity data pool was chosen based on a desire to balance method effectiveness – the two-dimensional results (see Figure 26) indicated that the method became more accurate as more low-fidelity data was included, but attempting to include 15 low-fidelity points for each of the 1,000 high-fidelity cases at the high end of the range would be computationally devastating. A constant set of 1,000 low-fidelity cases would equate to a data source ratio between 10 (for 100 high-fidelity cases) and 1 (for 1,000 high-fidelity cases). The two sets of data were then combined and modeled jointly, as described in Section 5.1.5. If data harmonization performance degraded relative to the other methods as more high-fidelity samples were included, the experiment could be repeated with a constant proportion of low-fidelity samples.

All cases in this study were randomly selected, and case selection would affect the performance of the model. To account for this random effect, the sampling-and-prediction process was repeated at least 500 times for each training data set size – i.e., 500 repetitions with 100 Cart3D samples, 500 repetitions with 200 Cart3D samples, etc. – and the resulting RMSE scores averaged.

5.4.3 Relative Speed of Each Method

The reader may recall from Section 3.3 that the time required to fit a Kriging model increased as the number of points in the data set increased. In general, for a set of m points, the effort to fit the Kriging model grows as $\mathrm{O}(m^3)$.[136] All surrogate models evaluated in these experiments used Kriging, and thus the training time would increase as more points were considered.

As the number of high-fidelity samples ranged from 100 to 1,000, the amount of time required to build and evaluate each model grew. The time required to create a surrogate and evaluate its predictive accuracy was averaged over the 500 repetitions performed; the

results are reported in Table 4. Most methods demonstrated similar time requirements. The exception here was data harmonization, which required almost two orders of magnitude more time to train models and evaluate predictive accuracy.

Table 4: Average Time To Build & Evaluate a Surrogate (in seconds)

Number of Points	Single Fidelity	Additive Correction	Proportional Correction	Ghoreyshi Cokriging	Data Harmonization
100	1	2	1	1	95
200	2	3	2	3	96
300	4	5	4	5	116
400	7	8	7	9	141
500	12	12	11	14	175
600	18	20	18	21	204
700	26	28	27	31	233
800	35	39	36	42	279
900	44	49	44	54	300
1,000	56	63	57	69	336

As stated earlier, data harmonization surrogate models for these tests were trained using the specified number of high-fidelity samples and 1,000 low-fidelity data points. Even when only 100 high-fidelity samples were used, the full data pool contained 1,100 samples – much more than the 100 samples being modeled by the other methods. Furthermore, data harmonization sometimes threw out-of-memory errors when attempting to predict response values for all test points simultaneously. As a work-around, the response was predicted for each test point individually, which reduced the memory burden but increased execution time.

The bulk of the processing in support of these experiments was done on High Performance Computing clusters. Jobs were submitted for processing using a queue system wherein the user specified the number of computing nodes needed for the job, as well as the expected job duration. The queue system limited jobs to a maximum duration of 7 days. Running in parallel on 12 cores, the full data harmonization test with 500 repetitions at each size would

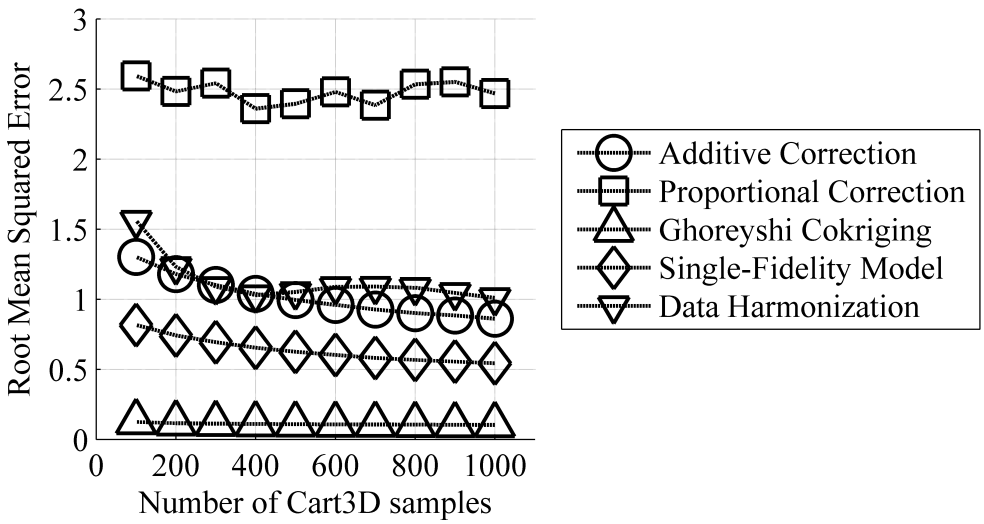

Figure 34: Prediction RMSE for All Multi-Fidelity Methods at Mach 4, AoA 0°

have required over 11 days, compared to around 1.4 days for the full Ghoreyshi cokriging test. The computational investment required for data harmonization was clearly much larger than for the other methods.

5.4.4 Results of Mach 4, AoA 0° Test

The average RMSE of the prediction error was tracked for each modeling method; the results may be seen in Figure 34. Curiously, the single-fidelity approach was competitive in this case, being on average more accurate than additive or proportional correction.

Unlike the other methods, proportional correction behaved somewhat erratically; note that there were multiple instances where the accuracy of proportional correction did not increase as more training data was made available. The smooth increase in accuracy displayed by the other methods suggested that the accuracy of proportional correction was more capricious than the others, with substantially more variability in the quality of the surrogates. This was not a desirable quality.

Data harmonization and additive correction exhibited similar performance, which was unsurprising given the similarity between the two methods that was identified in Section 5.1.5.2. Ghoreyshi cokriging exhibited very good predictive accuracy, even when relatively few high-fidelity samples were available. As seen in Table 4, this method did take

slightly longer than most of the other methods (commonly 10-20% longer). The improvement in accuracy for this flight condition was considered sufficient to justify the increased processing time.

Note that the effects of uncertainty information were not called out in these results. It was observed that, for this response, the choice of uncertainty data had a much smaller effect than the choice of multi-fidelity technique. Given the scale of the ordinate axis in Figure 34, if the results for each of the Ghoreyshi cokriging models (no uncertainty, iteration noise effects, surrogate approximation noise effects, and both noise effects) were plotted, all of the resulting icons would appear to be coincident.

5.4.5 Results of Mach 4, AoA 40° Test

The four multi-fidelity methods were then applied to the Mach 4, α 40° data set. Once again, each method was applied using four different noise types which were captured via nugget: zero noise, noise from the iterative solution process, noise from low-fidelity prediction error, and noise from both the iterative solution process and the low-fidelity prediction error. The results are plotted in Figure 35.

Unlike the previous test, in this case the proportional corrector demonstrated very good performance. Proportional correction and Ghoreyshi cokriging had effectively the best predictive accuracy as judged by RMSE. Once again, the single-fidelity method out-performed the additive corrector. The data harmonization predictive accuracy gradually improved, approaching that of the additive corrector, as more high-fidelity cases were included in the data set. Given the lackluster performance of data harmonization and the very large computational expense associated with the method as noted in Table 4, tests of data harmonization were curtailed before the full analysis was complete.

As with the α 0° results, the choice of uncertainty information did not significantly affect the predictive accuracy of the surrogates. The choice of multi-fidelity technique had a much larger effect on predictive accuracy.

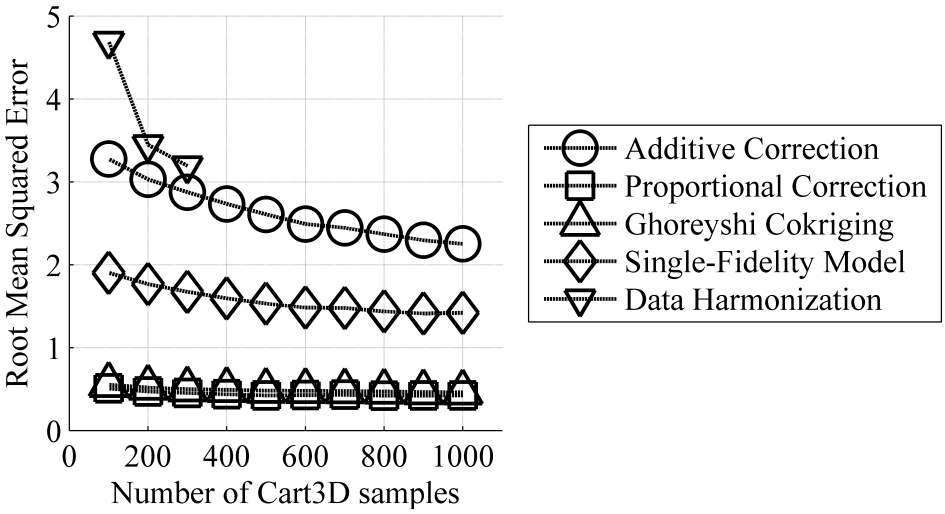

Figure 35: Prediction RMSE for All Multi-Fidelity Methods at Mach 4, AoA 40°

5.5 Observations & Further Inquiry

Given the strong performance of Ghoreyshi cokriging for both sets of test data, it was selected as the most effective multi-fidelity technique for this application. It served as the default modeling approach for subsequent experiments.

A number of curious observations were made in the course of these tests, such as the highly variable performance of proportional correction and the negligible effect of incorporating uncertainty. The next section will address those observations in greater depth.

5.5.1 Discrepancy in Proportional Correction Performance

Qualitatively, most of the results for Mach 4, α 40° resemble those observed for Mach 4, α 0° with the exception of proportional correction. Proportional correction had the worst performance at the low angle of attack but extremely good performance at the high angle of attack.

A few statistical details about the data points used to test predictive accuracy – 1,719 points for α 0° and 1,693 for α 40° – are given in Table 5. Critically, note the very large variation in ratios between the low- and high-fidelity C_M values for α 0°, and the very small range of variation for the same ratio at α 40°.

Proportional correction fits a surrogate model to the ratio of the high-fidelity response

Table 5: Distribution of C_M For Both Flight Conditions

	Mach 4, α 0°			Mach 4, α 40°		
	Cart3D C_M	APAS C_M	Ratio of Cart3D to APAS	Cart3D C_M	APAS C_M	Ratio of Cart3D to APAS
Minimum	-12.2	-33.1	-113	0.02	0.88	0.02
Maximum	2.99	6.14	61.7	27.4	72.3	0.60
Average	-0.63	-1.93	0.22	5.10	12.8	0.41
Standard Deviation	1.13	3.05	4.17	3.61	9.46	0.06
Within ±1	79%	52%	–	0.1%	0.0%	–

to the low-fidelity response (as seen in Equation 24 on page 137). When the response values are close to zero, especially the low-fidelity response, small absolute changes in the values may produce large variations in the ratio. This was observed in the α 0° results. These large variations can be difficult for a surrogate model to fit effectively. As a result, proportional correction models may have poor accuracy when the ratio varies rapidly, such as when the low-fidelity response is often close to zero.

Conversely, at α 40°, neither the low- or high-fidelity data encompassed zero, and very few of those cases fell within ±1. The ratio values which result were much less volatile and therefore easier to predict. This led to the excellent predictive accuracy demonstrated by the method in Figure 35.

5.5.2 Modeling Uncertainty in Pitching Moment Coefficient

At both flight conditions, the four options for incorporating uncertainty resulted in very small effects with respect to pitching coefficient prediction RMSE, on the order of 1–2%. This is very small relative to the differences in predictive accuracy between multi-fidelity methods, which were on the order of 80% or larger. Given the scale of the vertical axis in Figure 34, if every variant of a given multi-fidelity method were plotted, the set would appear to be coincident.

Modeling uncertainty did not improve the prediction of pitching moment coefficient at either flight condition. However, pitching moment coefficient had relatively low levels of iterative noise. In fact, the pitching moment was the most highly-weighted factor for the adaptive grid refinements of Cart3D (see Appendix D.1.3.2 for more details). As a result, the pitching moment was often well-converged.

The use of nuggets was primarily intended to improve the accuracy of surrogate models for lateral responses, which were more susceptible to noise. Although this improvement was not observed for pitching moment (a longitudinal response), it could still occur when modeling lateral responses. The nine-dimensional experiments were therefore repeated, using yawing moment coefficient as the response of interest rather than pitching moment coefficient.

5.6 Comparing Prediction Accuracy: Yawing Moment

To review from Section 4.10, the data set featured 9 independent variables – nose droop, nose fineness ratio, two nose spline shape parameters, wing half-span fraction, wing airfoil camber, vertical-tail-to-wing area ratio, wing root chord fraction and fuselage radius fraction – all of which affected the vehicle outer mold line (OML) in a symmetric fashion. Additionally, the available data points were for flight conditions with zero sideslip. The flow solver does not capture viscous effects, so effects such as vortex shedding,[16] which might introduce oscillating lateral forces and moments, were not present in the simulations.

In short, the mechanisms that would introduce real lateral forces or moments – asymmetric OML, asymmetric flight condition, viscous effects, etc. – were not present, and thus any such forces or moments which appear in the data set could be assumed to be spurious, resulting from the simulation itself rather than the phenomena being simulated. A similar conclusion could be made for the low-fidelity data from APAS.

The Unified Distributed Panel analysis (UDP), which acts as the subsonic and transonic analysis tool for APAS, consistently estimated asymmetric effects for symmetric cases to be zero. The Supersonic Hypersonic Arbitrary Body Program (S/HABP) is used for supersonic and hypersonic analyses in APAS and may predict small but nonzero lateral responses for

symmetric cases. The flight conditions included in this study were Mach 4, α 0° and Mach 4, α 40°, and both the high-fidelity and low-fidelity data demonstrated some degree of noise.

When a configuration is analyzed using Cart3D, the program discretizes the volume around the configuration into cells and repeatedly solves the three-dimensional Euler equations for a perfect inviscid gas within each cell until the solver converges to a steady-state solution.[6] Because the equations are solved on a grid of cells rather than a continuum, some small errors may be introduced. As the mesh is refined, this discretization or "truncation error" is reduced;[165] this also corresponds to increasing computational effort, as more calculations must be performed for each iteration. The discretized nature of the analysis can lead to oscillatory behavior, as shown in Figure 7 on page 42.

Because both data sources included some degree of noise – iteration noise for Cart3D data, surrogate prediction error for estimated APAS data – it was expected that this study would test whether the inclusion of uncertainty information via the Kriging nugget would improve model prediction accuracy for noisy or uncertain data. For each of the four modeling techniques (additive correction, proportional correction, Ghoreyshi cokriging, and single-fidelity modeling)[2], four variants were tested. Each variant was defined by the uncertainty data which was incorporated using nuggets. The options for uncertainty were:

- No uncertainty, i.e. all data points are deterministic;
- Uncertainty from solver iteration, which is unique to each high-fidelity data point;
- Uncertainty from surrogate model prediction error, which is a constant value for every low-fidelity data point; and
- Uncertainty from both solver iteration and surrogate model prediction error, which would be a combination of unique and constant values.

As in the previous study, random points from the first 4,000 space-filling data points were selected and used to build a surrogate model. That surrogate model was then used

[2]Data harmonization was not included in this experiment in light of the excessive computational effort required and its poor performance in the previous test.

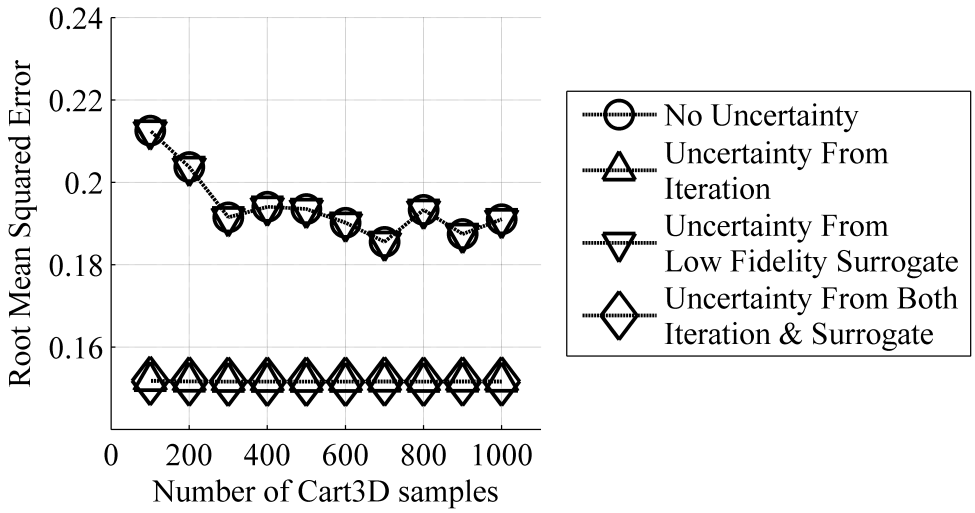

Figure 36: Ghoreyshi Cokriging Prediction RMSE for Yaw at Mach 4, AoA 0°

to estimate the response values for a separate set of roughly 1,700 data points, and the discrepancies between predicted and recorded values were used to calculate the prediction RMSE for that method and form of uncertainty.

5.6.1 Results of Mach 4, AoA 0° Study

For each multi-fidelity method, the same pattern was observed. Variant #1, the noiseless variant, had the largest (i.e. worst) prediction RMSE. Variants #2 & 4, the former using iteration uncertainty only and the latter using both iteration and surrogate prediction uncertainty, would have the smallest prediction RMSE. Curiously, the performance of variant #3, which included only surrogate prediction uncertainty, differed between multi-fidelity methods. For additive correction and single-fidelity modeling, variant #3 performed equally well as variants #2 & 4. For Ghoreyshi cokriging and proportional correction, variant #3 had equivalent performance to variant #1, the noiseless case. Figure 36 shows the results for Ghoreyshi cokriging. As noted above, these results are for the most part indicative of the results for the other multi-fidelity methods with the exception of variant #3.

The actual-by-predicted plots for two types of Ghoreyshi cokriging models, a noiseless model and a model which included iteration noise, may be seen in Figures 37a & 37b, respectively. In an actual-by-predicted plot, a perfect fit would appear as a straight line

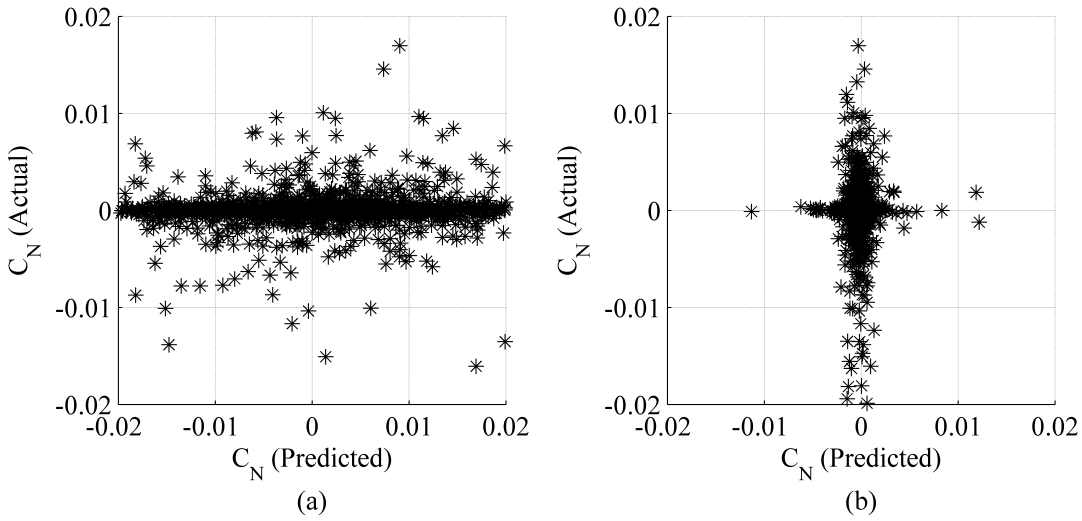

Figure 37: Actual-By-Predicted Plots for Yaw at Mach 4, AoA 0° For (a) Noiseless Model, and (b) Model Incorporating Iteration Noise

from the lower-left corner to the upper-right corner. Since all data came from steady-state symmetric models, the "true" yawing moment was expected to be zero for all cases. The bounds on the graphs were selected as a balance between capturing the range of the deterministic predictions and showing details of the "noisy" model's predictions.

As expected, the deterministic model attempted to reproduce the training data exactly. The strong horizontal spread in the data indicated that this model would over-predict yawing moments compared to the calculated values. In fact, the left and right bounds on this graph cut off 19% of the data points, while the upper and lower bounds cut off only 2.6% of the points.

Conversely, the model which incorporated iteration noise (the "noisy" model shown Figure 37b) had a strong vertical trend. This indicated that the predicted yawing moments were smaller than those calculated by Cart3D. Given that the flight condition and vehicle were symmetric, it was plausible that the predicted results might actually be a better representation of the results than the raw Cart3D results.

Lastly, it should be noted that, while the actual-by-predicted plot gives a decent sketch of the relationship between the actual and predicted response values, it lacks precision: points lay atop one another, making it difficult to convey the true distribution of results.

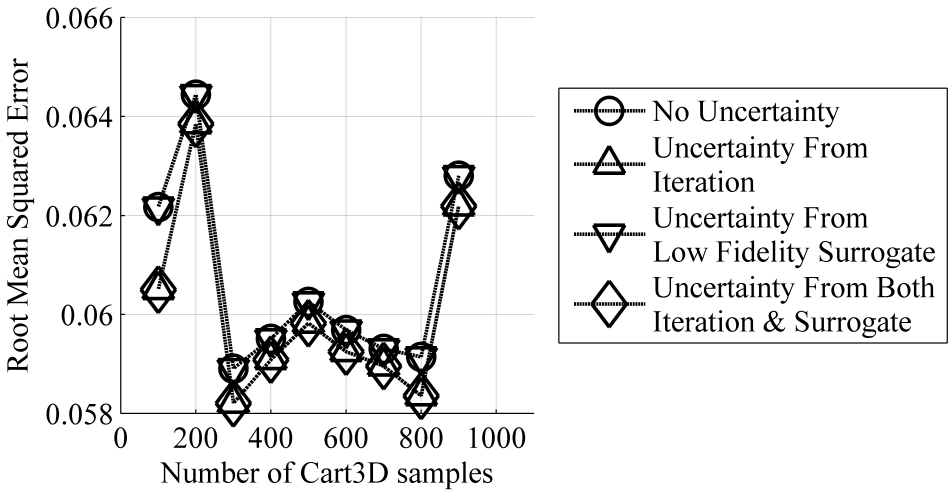

Figure 38: Ghoreyshi Cokriging Prediction RMSE for Yaw at Mach 4, AoA 40°

It is worth stating that 94% of the Cart3D results exhibited yawing moments within ± 0.005. Of the predictions by the noisy model, 99.6% fell within that range; the deterministic model predicted that only 33% of the cases had yawing moments within that range. This highlighted the degree to which the deterministic model over-fit the response behavior.

5.6.2 Results of Mach 4, AoA 40° Study

The results at the higher angle of attack were qualitatively similar to those at 0°: variants which included uncertainty due to iteration consistently had smaller RMSE values (i.e. more accurate predictions) than the noiseless variants, while the effect of each uncertainty variant depended on the multi-fidelity method being applied. The results for Ghoreyshi cokriging are shown in Figure 38. Unlike the previous flight condition, the relative difference between variants was much smaller in this study. In fact, the worst prediction RMSE at this flight condition by any method (0.081) was better than the best prediction RMSE at α 0° by any method (0.155). Although capturing uncertainty due to iteration noise improved surrogate accuracy, this improvement was smaller at this flight condition.

The results for this flight condition showed less variation between options. Qualitatively, both actual-by-predicted plots in Figures 39a & 39b have moderate scatter. Each exhibits a pattern which is more strongly horizontal than vertical, indicating that both models were

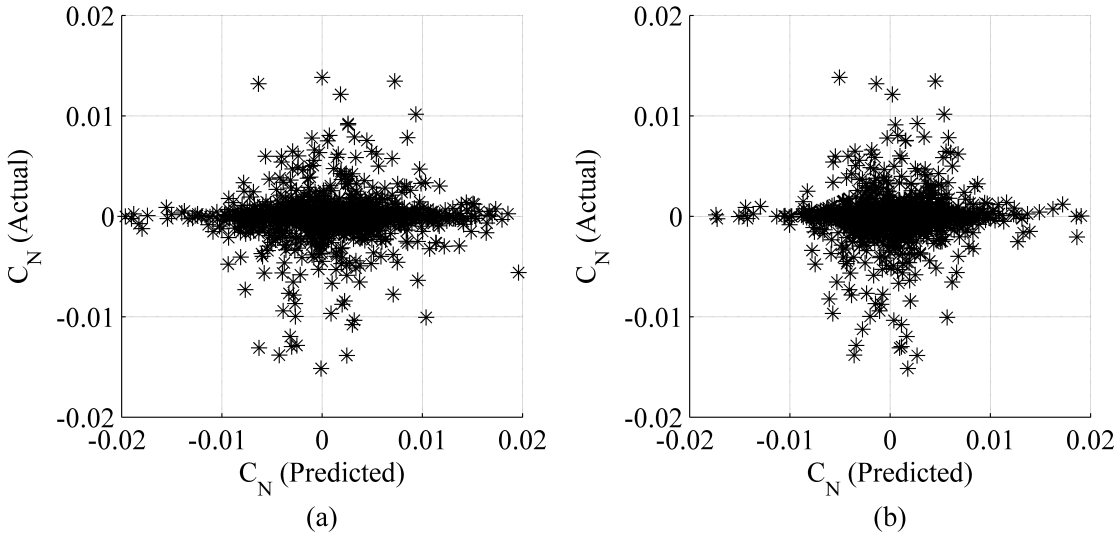

Figure 39: Actual-By-Predicted Plots for Yaw at Mach 4, AoA 40° For (a) a Deterministic Model, and (b) a Model Incorporating Iteration Noise

over-fitting the data to some degree. This was reflected in the distribution of predictions: 93% of the Cart3D results had yawing moments within $\pm$ 0.005, whereas 82% of the "noisy" model's predictions and 70% of the deterministic model's predictions fell outside that range. At first glance, the distribution of the Cart3D results were roughly equivalent, but the noiseless model had much better agreement. The data sets were investigated to determine the cause of the differences in performance.

JMP statistical software[91] was used to analyze the yaw data at each flight condition. The first analysis was to evaluate the distributions of the yaw and iteration noise values. Unexpectedly, the iteration noise at 40° was typically an order of magnitude *less* than at 0°, not greater. This was in stark contrast to what was observed in the motivating example (Section 2.5.3) which evaluated iterative noise at Mach 0.9. The reduced iteration noise would indicate that the high-α results at Mach 4 are less subject to random effects; this may be the reason for the smaller RMSE values observed for even the deterministic models when applied to the 40° data.

The second step was to analyze the correlation between the variables. Given a set of data, the correlation between two parameters can be estimated with the sample correlation

coefficient,[76] also known as the Pearson product moment correlation coefficient:

$$r = \frac{\sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y}}{\sqrt{\sum_{i=1}^{n} x_i^2 - n\bar{x}^2}\sqrt{\sum_{i=1}^{n} y_i^2 - n\bar{y}^2}} \tag{35}$$

Here, r is the correlation coefficient, n is the total number of samples in the data set, x_i and y_i are the individual observations of variables x and y respectively, and $\bar{x}$ and $\bar{y}$ are the average values of x and y over the n samples.

If two variables behave similarly, e.g. when one increases the other tends to increase, they are said to be correlated. The sample correlation coefficient quantifies this relationship. A correlation coefficient value of 1 indicates perfect positive correlation, i.e. when x increase, y will always increase. A value of -1 indicates negative correlation: when x increases, y will always decrease. A value of 0 indicates *no* correlation, i.e. knowledge of x tells you nothing about the behavior of y.

Table 6 displays the correlation between the absolute value of the yaw response calculated by Cart3D, the standard deviation of the Cart3D yaw solution over the last 20 iterations of the analysis, and the absolute value of the APAS yaw solution. Note the negligible correlation between the Cart3D and APAS yaw values. This was unsurprising given that both values were expected to be spurious.

In the AoA 0° data, there was a moderate positive correlation between the yaw response and the observed iteration noise, 0.45. The standard deviation cannot take a value less than zero, so taking the absolute value of yaw would indicate whether there was more observed noise for cases that were reported to be farther from zero, which seemed to be the case. Iteration noise being larger for cases farther from zero suggested two things: first, that those cases were more likely to be spurious and thus negligible; and second, the Kriging models which capture that iteration noise via nugget would be more likely to disregard those cases and tailor the model to the cases with less noise (and likely smaller yaw values).

The AoA 40° results showed much less correlation between the iteration noise and the Cart3D yaw result. That observation would indicate that even when iteration noise was

Table 6: Correlation of Yaw Data

	Mach 2.5, α 0°			Mach 2.5, α 40°		
	|Yaw| (Cart3D)	σ_{Yaw}	|Yaw| (APAS)	|Yaw| (Cart3D)	σ_{Yaw}	|Yaw| (APAS)
|Yaw| (Cart3D)	1.00	0.45	0.04	1.00	0.16	0.00
σ_{Yaw}	0.45	1.00	0.05	0.16	1.00	-0.02
|Yaw| (APAS)	0.04	0.05	1.00	0.00	-0.02	1.00

taken into account, the resulting models would not be as effective at screening out that noise to identify the underlying response behavior. This matched the results seen in Figure 39 where both modeling approaches were approximately equally accurate. A small correlation was still present, however, and capturing iteration noise *did* result in a mild improvement in prediction accuracy. This indicated that, although capturing uncertainty with nuggets did improve prediction accuracy, this improvement was to some degree proportional to the extent that the data exhibited significant noise.

5.6.3 Discrepancy Between Expectations & Observations

In light of observations made during the Reusable Booster System project, the AoA 0° flight condition was expected to exhibit relatively low noise, while the AoA 40° flight condition would exhibit around an order of magnitude greater noise. The data obtained during this research effort showed those behaviors to be exactly reversed.

Fortunately, the data sets still presented the opportunity to test the methods on responses with varying degrees of iteration noise, and the results did in fact confirm the expectations: capturing uncertainty via nuggets had greater effect when the degree of noise in the response was larger. This outcome did little to address the question: why did the observed behavior differ from expectation?

The changes in the way that Cart3D was operated were investigated to determine which, if any, produced the unexpected solver behavior. First, most analyses during the Reusable Booster project ran Cart3D in "robust" mode,[4] which forced the flow solver to calculate

the solution gradients at every stage of the Runge-Kutta iteration scheme, rather than just at the first stage. Robust mode was used to address cases which were failing to run to completion. It was not found to be necessary for the current analyses. Running cases in robust mode at Mach 4.0, α 0° and 40° did not produce a discernable change in iteration noise.

Secondly, the convergence tolerance defined in the **aero.csh** script (see Appendix D for more details) was set to a more restrictive default value in the current version of Cart3D (1.4.7) than in the version used for RBS tests (1.4.3). During the RBS tests, the functional error tolerance was set to 0.005 by default, and this value was reduced to 0.001 after consultation with the software developers.[135] In version 1.4.7, the default value is 0.000001, or 10^{-6}, a difference of three orders of magnitude. Cases run with the relaxed error tolerance did not produce any discernable effect on the iteration noise.

The expectation of iteration noise increasing with angle of attack was drawn from observations made during the RBS study. Those observations were made at Mach 0.9. It was possible that the increasing iteration noise was specific to the transonic flight regime. When the distribution of iteration noise with respect to angle of attack was investigated for Mach 2.5 using the RBS data set, it was found that those observations matched the ones described in this chapter. Rather than being a general characteristic, it would appear that the iteration noise was strongly dependent on the speed regime being simulated.

5.7 Observations & Conclusions

The yawing moment study showed that the use of nuggets to capture uncertainty resulted in improved prediction accuracy at both flight conditions. Specifically, representation of uncertainty due to solver iterations was found to significantly improve the prediction accuracy of all modeling techniques evaluated. Representation of uncertainty due to surrogate model prediction errors improved prediction accuracy for *some* modeling techniques but not others. The degree of improvement depended on the degree to which iteration noise corresponded to extreme results.

Uncertainty due to surrogate prediction error was estimated when the low-fidelity surrogate was trained and tested, which means that this information was effectively free. Including this source of uncertainty did not uniformly improve prediction accuracy, but it was never observed to degrade accuracy either. Because the information cost nothing to quantify and could possibly improve model accuracy, it was decided that uncertainty due to solver iteration and uncertainty due to surrogate prediction error would both be retained for use in future uncertainty calculations.

Hypothesis 3 asserted that:

When creating a Kriging model, the use of nuggets will capture uncertainty in the data, improving predictive accuracy for noisy responses.

In light of the fact that models which capture response uncertainty using nuggets were shown to have better prediction accuracy than those which did not, Hypothesis 3 could be considered *supported*.

In a similar vein, Hypothesis 2 asserted that:

Data fusion techniques will allow results from high-fidelity analyses to be augmented with cheaper sources of data to produce surrogate models that are more accurate yet require less computationally-expensive data.

Experiments in this chapter demonstrated that the use of data fusion techniques for multi-fidelity modeling improved prediction accuracy over what could be achieved with only one source of data. This result indicated that Hypothesis 2 could also be considered *supported*. This concluded the evaluation of the lower-tier hypotheses, and cleared the way for the testing of the main hypothesis in the next chapter.

CHAPTER VI

INTEGRATED MODELING & SAMPLING PROCEDURE

"Cuius rei demonstratioem mirabilem sane detexi hanc marginis exiguitas non caperet."

"I have a truly marvelous demonstration of this proposition which this margin is too narrow to contain." – Pierre de Fermat[116]

The two preceding chapters illustrated how the selected methods produced the desired effects: contour-based sampling improved prediction accuracy for a particular response range of interest in Chapter 4; Chapter 5 illustrated that capturing uncertainty using nuggets resulted in better accuracy for lateral responses, while leveraging cheaper data sources using Ghoreyshi cokriging improved prediction accuracy for the longitudinal response.

Although these enhancements were individually significant, further benefits could be derived by combining the techniques. The effectiveness of contour-based sampling depends on its ability to accurately estimate response values at various points in the design space. Combining contour-based sampling with multi-fidelity modeling would improve the accuracy of response estimates, allowing a more accurate assessment of candidates. This was expressed by the primary hypothesis:

By placing samples intelligently, reducing dependence on the expensive models, and better quantifying the level of confidence in each data point, the selected methods will reduce the computational expense of high-fidelity modeling to sufficient extent that it becomes a feasible option earlier in the design process.

A relatively modest demonstration was desired to determine what degree of improvement the integrated methods could offer. The nine-dimensional design space, first described in Section 4.10, was chosen for this demonstration as it would allow the results obtained to be compared against those from both space-filling samples and single-fidelity contour-based

sampling.

6.1 Simplified Test: Nine Input Dimensions, Three Responses

6.1.1 Creating an Integrated Algorithm

The scripts and functions created for the contour-based sampling exercise in Chapter 4 were augmented to use Ghoreyshi cokriging in place of single-fidelity modeling. The generic steps to implement the multi-fidelity sample-selection method will be given in **boldface**, while details about the author's implementation of the method will be given in plaintext.

First, the source of low-fidelity data is identified. In this case, APAS was used. **Secondly, the user must decide whether this data source can be applied directly (i.e. analyzing each case directly) or if a surrogate model is necessary.** Due to the number of evaluations required by contour-based sampling and the non-negligible time required for each APAS solution, surrogate models were used for this implementation.

To generate these surrogate models, the 16,000-case nested Latin hypercube was analyzed at each flight condition using APAS. BRAINN was then used to create neural networks for the responses of interest (C_M at each flight condition), as described in Section 5.4.1. Not all 16,000 cases were used in training: 20% of the cases were used for validation and 15% were used as test cases. The neural networks passed all goodness-of-fit tests with excellent performance. The neural networks were then formatted as Matlab functions which would take inputs, in the form of the geometric parameters which made up the nine-dimensional design space, and return estimates for the response for each input case. The functions were vectorized so that many cases could be assessed with a single function call.

Next, whether the data source is applied directly or surrogates are used, the low-fidelity response values are calculated for each high-fidelity training case. Once high- and low-fidelity responses are available for each training case, multi-fidelity Kriging models are fit to the training data, incorporating the low-fidelity responses as extra input dimensions as described in Section 5.1.4.

During implementation, the question arose whether it was more effective to fit models to Y_{high}, the high-fidelity responses, or $(Y_{high} - Y_{low})$, the discrepancies between the high- and

low-fidelity responses. The latter option might be considered a hybrid of Ghoreyshi cokriging and additive correction. When tested, it was found that the two methods produced predicted response values that differed by no more than 10^{-14}. Given that the two alternatives were indistinguishable with regard to predictions, it was decided to fit models to Y_{high} directly, since the alternative would require additional arithmetical operations for every model trained or response estimated.

Just as in single-fidelity sample selection, the updated Kriging models are used to select the next sample. The difference between the new algorithm and single-fidelity contour-based sampling (described in Section 4.1) is that multi-fidelity adaptive sampling requires that both low- and high-fidelity responses be estimated for all candidate and test points, because the multi-fidelity Kriging models need the low-fidelity responses to estimate the high-fidelity responses. Otherwise, the process of evaluating and selecting candidates remains the same.

6.1.2 Applying the Integrated Algorithm

When single-fidelity contour-based sampling was applied to the 9-dimensional problem in Section 4.10.9, the 500-case level of a nested Latin hypercube was used as the initial data set. This same data set was used to initialize multi-fidelity contour-based sampling to elucidate the effects of the extra data source. Each sample was selected using a fresh set of candidate and test points.

A tapering probability-of-interest (POI) requirement was used: the first five selection rounds required POI values to exceed 25%, while the next five rounds required POI to be greater than 15%. Rounds 11–35 required POI values above 5%, and rounds 36–70 simply excluded any candidates with POI values equal to zero. This POI schedule was intended to place early samples in regions that were expected to have very good performance while allowing later samples to explore regions of lower prediction confidence. If no candidate met the POI requirements, the candidate with the best POI value was selected as the next sample.

Once 70 new cases had been selected, their surface meshes were built in PaceLab and

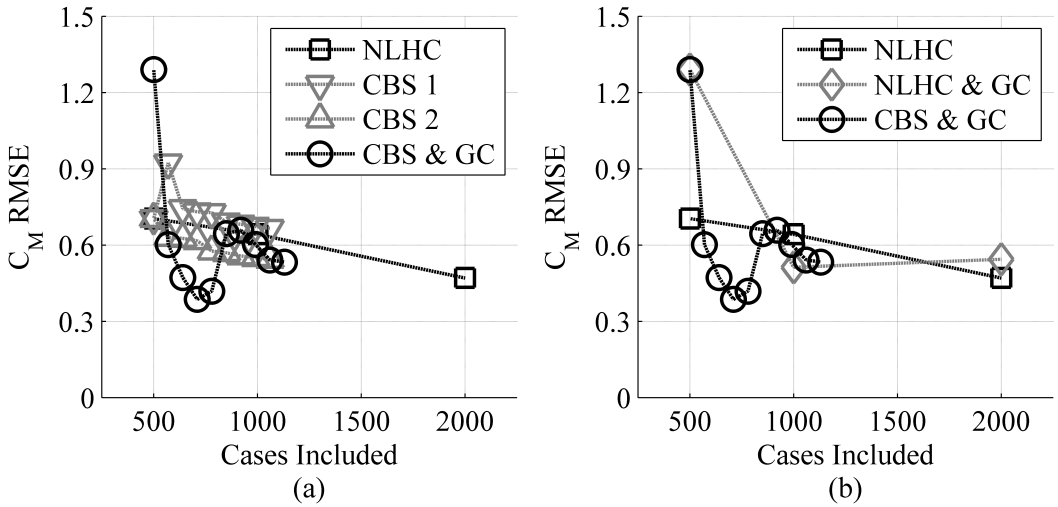

Figure 40: Prediction RMSE at Mach 0.3, α 15°

their aerodynamics analyzed using Cart3D. The results were added to the training data set. Kriging models were fit to each response based on the available data, and those models were used to predict response values for the set of independent test data. As described in Section 4.10.7, these test cases were a set of 1,470 configurations within the nine-dimensional space that produced pitching moment coefficients within ±0.1 at all three flight conditions (Mach 0.3, α 15°; Mach 0.8, α 0°; and Mach 2.5, α 0°). The prediction error for the Kriging models was quantified using root mean squared error (RMSE).

6.1.3 Evaluation of Accuracy

Figures 40a & b show the results for Mach 0.3, α 15°. The black square icons represent the models based only on nested Latin hypercube sampling for 500, 1,000 & 2,000 cases. As stated in Chapter 4, roughly 70% of the configurations analyzed produced well-converged results at all flight condition. Surrogate models based on these space-filling samples produce a smooth trend of improvement as more samples are included. Together, these images demonstrate not only the performance of the proposed approach compared to the baseline, but also the relative contributions of each technique.

In contrast to the single-fidelity modeling of space-filling samples, the results from applying single-fidelity contour-based sampling (i.e., selecting new samples based only on Cart3D

data) are represented by grey triangles in Figure 40a. These results were previously shown in Section 4.10 on page 113. The downward-pointing triangles represent the use of a low probability-of-interest (POI) requirement for the sample selection algorithm, which allowed it to explore the design space more freely. This series is labeled "CBS 1".

As a review from when these results were first presented in Section 4.10, the initial batch of samples for the low POI requirement actually worsened the predictive accuracy for this flight condition, although the second batch almost balanced that out. Progress was somewhat slow but steady as the model learned about the response behavior. For this flight condition, the rate of improvement was approximately equal to space-filling sampling.

The upward-pointing triangles, labeled "CBS 2," represent the use of a higher POI requirement, which resulted in the selection of new cases that were fairly close to existing data points. This series showed slow but steady improvement in predictive accuracy. The rate of improvement was roughly equal to that of the later rounds of the low-POI series. However, because the high-POI series did not suffer from early missteps, the overall performance of the high-POI approach was better than the low-POI approach, at least for this particular response.

Figure 40b compares the baseline and the combined techniques against the use of data fusion with space-filling samples. For the 500-case space-filling data set at this flight condition, using multi-fidelity modeling negatively affected predictive accuracy, resulting in an RMSE value (1.29) that was significantly larger than the RMSE value for the single-fidelity model of the same data (0.704). The diamond icons show that data fusion, when applied to space-filling samples, had mixed results for this response. It produced a mild improvement for the 1,000-case data set, a mild degradation for the 2,000-case set, and a sharp degradation for the 500-case set.

In contrast, the black circles denote models based on the full proposed method, which leveraged both Goreyshi cokriging *and* contour-based sampling. The series starts with the same performance as the space-filling multi-fidelity surrogate, representing the performance of the combined techniques before adaptive sample selection began. Because the combined techniques were initialized using the multi-fidelity surrogate of the 500-case data set, the

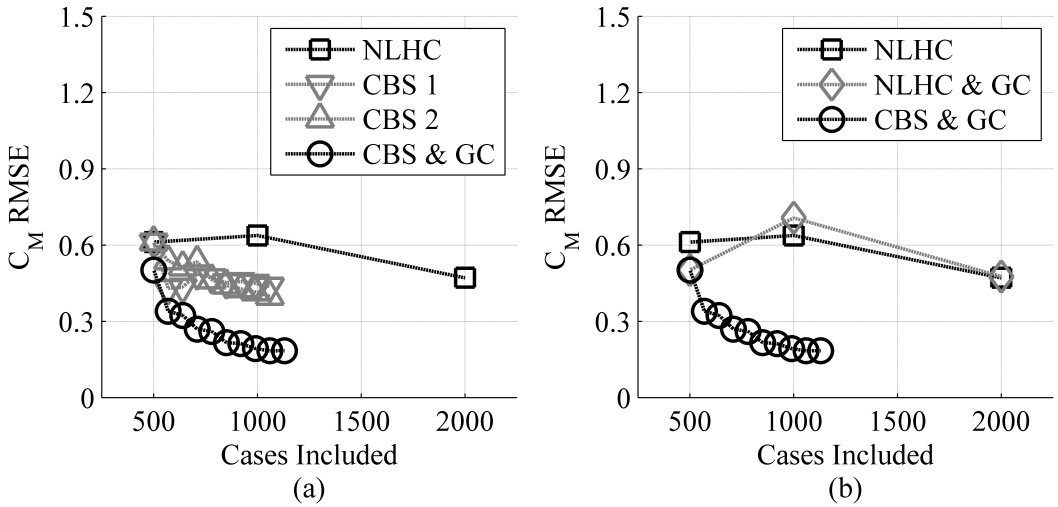

Figure 41: Prediction RMSE at Mach 0.8, α 0°

same poor initial accuracy was observed. Despite this, the proposed approach showed its worth quickly, and more than made up for the initial degradation after a single batch of adaptive samples. Model accuracy continued to improve with further sampling until the fourth and fifth rounds. Recall that in each round, half the samples were selected using a minimum POI of zero, allowing the algorithm to select cases which might be of interest to *any* response rather than only those expected to be of interest to *all* responses. It is possible that samples were selected in the fourth and fifth rounds that were in regions with poor performance at this flight condition, and when the Kriging model was updated to capture those samples, its ability to model the actual region(s) of interest for this flight condition was negatively affected. After one or two batches, the predictive accuracy once again began to improve at a rate faster than that of the space-filling or single-fidelity approaches.

The RMSE results for Mach 0.8, α 0° are displayed in Figures 41a & b. The 1,000-case level of the NLHC appeared to contain misleading results for this flight condition, based on the observation that the associated surrogate model was less accurate than the one based on the smaller 500-case NLHC set. The 2,000-case set produced a marked improvement over both smaller sets.

The two single-fidelity adaptive sampling approaches, marked as grey triangles in Figure 41a, showed a gradual improvement as more samples were selected by the adaptive

sampling algorithm. This held true for both the high-POI-requirement series (denoted by upward-pointing triangles) and the low-POI-requirement series (denoted by downward-pointing triangles). No matter which POI approach was used, single-fidelity adaptive sampling out-performed space-filling sampling for this response. In fact, the single-fidelity adaptive sampling approaches out-performed the space-filling approach even when almost twice as many space-filling samples were available.

The use of space-filling samples and multi-fidelity modeling, marked with grey diamonds in Figure 41b, once again produced mixed results. The initial gain in predictive accuracy for surrogates trained with the 500-case data set (from an RMSE of 0.61 for single-fidelity modeling to 0.50 for multi-fidelity modeling) became a loss of accuracy for the 1,000-case data set; there was minimal difference in predictive accuracy between single-fidelity and multi-fidelity surrogates trained on the 2,000-case data set.

The black circles, on the other hand, denote the results of using the proposed approach, multi-fidelity modeling *and* adaptive sampling. The first batch of samples selected by the combined techniques produced a substantial improvement in prediction accuracy, reducing RMSE from 0.50 to 0.34. Later batches continued to produce improvements, although none so substantial as the first batch. Some evidence of diminishing returns was observed. Overall, the proposed approach was very effective at improving surrogate model predictive accuracy for this response, out-performing single-fidelity models based on either adaptive sampling or space-filling sampling.

Lastly, Figures 42a & b depict the results at Mach 2.5, α 0°. The space-filling cases exhibited a more-or-less linear trend of improving accuracy as the training data pool grew, although at a more gradual rate than was observed for Mach 0.3.

The single-fidelity adaptive sampling approaches, denoted by grey triangular icons in Figure 42a, had very different initial behavior. The series with the low POI requirement, which was more tolerant of exploratory sampling, initially produced a sharp reduction in prediction error, but lost some of those gains after a few rounds. Still, even after that reduction in accuracy, this series out-performed both other single-fidelity approaches. The more restrictive single-fidelity adaptive sampling approach, with the high POI requirement,

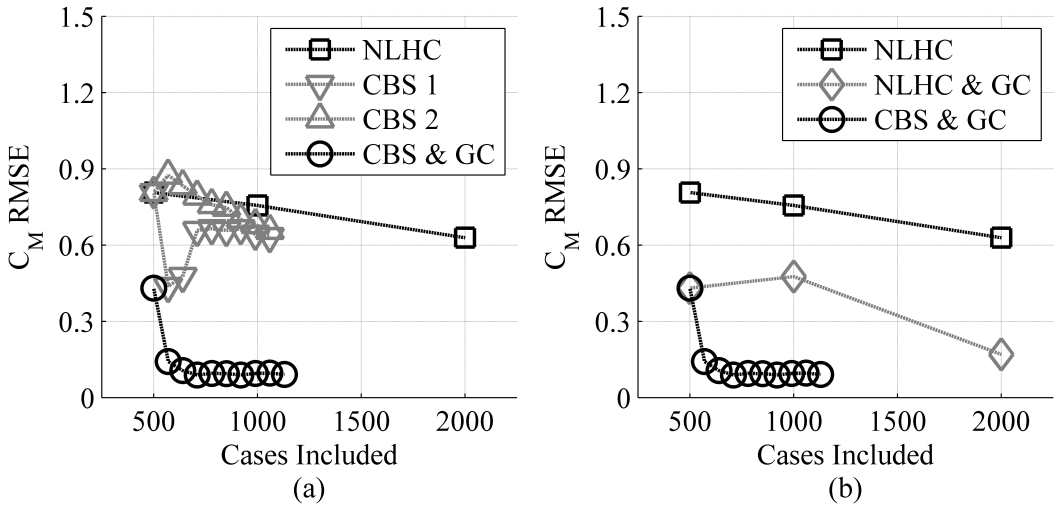

Figure 42: Prediction RMSE at Mach 2.5, α 0°

initially became less accurate but then improved rapidly. In fact, despite the initial misstep this series outperformed the space-filling approach when both approaches had an equal quantity of data available.

The grey triangles in Figure 42b illustrate the effects of space-filling samples and multi-fidleity modeling, incorporating APAS data through the use of Ghoreyshi cokriging. A substantial improvement was observed for all space-filling data sets, and in most cases, prediction error was halved. For example, for the 500-case data set, RMSE was reduced from 0.81 for single-fidelity modeling to 0.43 for multi-fidelity modeling of the same samples.

The black circles in both Figures 42a & b show the predictive performance for surrogates made with both multi-fidelity modeling and contour-based sampling. Even though prediction error had already been reduced substantially by the use of multi-fidelity modeling alone, the first batch of multi-fidelity contour-based samples provided almost an equal improvement. This brought the prediction RMSE below 25% of the original space-filling value. This result suggests a relatively simple relationship between the behavior of the low- and high-fidelity responses. Later rounds also improved performance slightly, although for the most part the predictive accuracy appeared to be steady.

Overall, the combination of multi-fidelity modeling and contour-based sampling produced significant improvements in prediction accuracy for all three flight conditions. At Mach 2.5,

using only 210 samples the integrated algorithm achieved a level of prediction accuracy that would likely require thousands more space-filling cases to match. This appears to be strong support for the hypothesis that these methods will enable the use of high-fidelity modeling earlier in the design process than was previously possible.

This test problem with its 9 free parameters was still fairly simple compared to a typical design problem. Before the primary hypothesis of this research could be confirmed, the method must be applied to a more challenging problem: the complete design space of the Reusable Booster System study.

6.2 Full-Scale Test: Forty-Nine Input Dimensions, Twelve Responses

This experiment was intended to test the effectiveness of the proposed method for a representative engineering problem. The design of a Reusable Booster System (RBS) was difficult because of the relatively large number of input dimensions and the complexity of the response. Such a vehicle must fly a very demanding trajectory, with a broad range of attitudes and speed regimes.[71] Planned trajectories for such vehicles include angles of attack up to and including 40°, much larger than most air vehicles.[19, 25, 98] Pamadi et al.[144] showed that a minimum of Euler CFD would be required for some of the flight conditions, particularly the higher angles of attack where lower-fidelity analysis tools become less accurate.

Higher-fidelity modeling such as Euler CFD or viscous simulation would entail significantly increased computational effort per analysis. Additionally, the many input parameters made for a large design space; if the response behavior was even moderately complex with respect to the input parameters, understanding the response at a resolution adequate for decision-making purposes could require a very large number of analyses. At the present time, creating an accurate surrogate model for a large design space would require tens or hundreds of thousands of analyses, with correspondingly daunting computational costs.

The wide range of flight conditions meant the analyst must simulate a potential configuration at numerous conditions to evaluate its effectiveness – a configuration which has tolerable aerodynamic moments at one flight condition might be unmanageable at another, and a useful RBS design *must* be controllable over its entire return-to-launch-site trajectory.

However, this challenge held the seeds of success: by modeling only the design space regions which were likely to be controllable for all flight conditions, the computational requirements might be reduced significantly. Incorporating a cheaper source of data would further diminish the number of high-fidelity analyses required. These observations led to the proposed method that was the subject of this research.

6.2.1 Flight Conditions of Interest

The flight conditions for the full-scale experiment were selected based on likely trajectories. The Fly-Back Booster designed by DLR[51, 98] was expected to fly a return-to-launch-site (RTLS) trajectory from a reentry speed just under Mach 6, with an angle of attack schedule that peaked at 35° at Mach 6 and fell below 10° at Mach 4. This vehicle featured a secondary turbine propulsion system for powered subsonic flight, enabling higher staging speeds. At the other end of the spectrum, the Langley Glide-Back Booster[26, 144] did not include a secondary propulsion system. Instead, its reference trajectory featured an unpowered aerodynamic turn and gliding return to the launch site. This trajectory placed an upper limit on the staging Mach number: if the booster staged much faster than Mach 2, it would travel too far from the launch site to glide back. The nominal trajectory featured a peak speed of Mach 2 at an angle of attack of 48°, with the angle of attack falling rapidly to below 20° at speeds below Mach 1.4.

The present effort focuses on the "rocketback" RTLS maneuver,[79] in which the main propulsion system of the reusable booster would be used to decelerate the vehicle after staging. This extinguishes the horizontal velocity of the vehicle, limiting the downrange distance traveled before the vehicle can perform an atmospheric turn and begin its unpowered flight back to the launch site. This maneuver would also drastically reduce the heating experienced by the vehicle during its descent.[19] Reference trajectories started with reentry at angles of attack up to 40° and reached peak speeds between Mach 2.5–3. The angle of attack would then be reduced to roughly 10°, dipping close to 0° as the booster fell below the speed of sound and then returning to 5–10°.

To approximate these reference trajectories, four flight conditions were selected from the

RBS modeling effort described in Chapter 2:

- Mach 2.5, α 40°, β 0°;
- Mach 2.5, α 15°, β 0°;
- Mach 0.8, α 0°, β 0°; and
- Mach 0.3, α 15°, β 0°.

These flight conditions sketched out a rocketback RTLS trajectory, including each of the various flow regimes such a vehicle would experience. Table 7 details the performance of the neural networks produced during the RBS effort, each of which was trained using roughly 10,000 cases. The test R^2 values indicated that the neural networks are capturing almost all of the variability observed in the data; this showed that the overall variability in the data set was also large, supporting the conclusion reached in Section 2.5.1 that many of the configurations being analyzed had poor aerodynamic qualities. Although the R^2 values for test data were high, which is one indication of a good fit, the prediction error had a fairly large standard deviation.

Table 7: Neural Network Prediction Accuracy for C_M

	C_{roll} Test R^2	C_{roll} σ_{Error}	C_{pitch} Test R^2	C_{pitch} σ_{Error}	C_{yaw} Test R^2	C_{yaw} σ_{Error}
Mach 0.3 α 15°	0.838	0.066	0.992	0.395	0.801	0.043
Mach 0.8 α 0°	0.949	0.076	0.961	0.264	0.915	0.044
Mach 2.5 α 15°	0.959	0.017	0.997	0.262	0.929	0.015
Mach 2.5 α 40°	0.956	0.028	0.999	0.456	0.925	0.015

These large standard deviation values indicated that any prediction made with these surrogates would have significant uncertainty. Error bars could be used to quantify the

magnitude of that uncertainty. For example, the standard deviation for C_{pitch} at Mach 0.3, α 15° was 0.395; if a prediction were made with this neural network, a set of error bars that would have 95% confidence of enclosing the actual response (as calculated by Cart3D) would have to extend $\pm 2\sigma$, or ± 0.79. This would be a very large range, especially compared to the stated response range of interest (± 0.1), and could make the difference between a viable design and one that cannot be controlled. Given that range, it was unlikely that a designer could use those surrogate models for design purposes with any confidence

The full-scale experiment was therefore intended to determine whether the combined methods – contour-based sampling, multi-fidelity modeling, and incorporating uncertainty – would offer any improvement in performance relative to the previous results. The objective had changed somewhat from what the RBS project attempted to do: whereas that effort attempted to produce surrogate models which would be equally accurate throughout the design space, the present objective was to be as accurate as possible for regions where moments are close to zero, and sufficiently accurate in other regions that those regions could be identified as having moments far from zero. Testing the effectiveness of each method would require a pool of test cases with moments close to zero. These test cases first had to be identified.

6.2.2 Selecting Test Cases

Acceptable test cases for this experiment had to have good performance at all flight conditions, representing a set of vehicle designs which were likely to be controllable along the return-to-launch-site (RTLS) trajectory. The same genetic algorithm approach which was used to identify test cases for the 9-dimensional problem (see Section 4.10.6) was applied to the current set of 49 free parameters. Although ultimately there were 12 responses of interest – the 3 aerodynamic moments at each of the 4 flight conditions – it was expected that the pitching moment coefficient at each flight condition would be the primary factor which determined whether a given configuration would be feasible. Thus, although there were 12 responses to be modeled, it was expected that test cases need only be selected on the basis of 4 of those responses: the pitching moment coefficient at each flight condition.

All data points from the previous RBS study which were analyzed at all 4 flight conditions relevant to the present study, and which met all the convergence criteria detailed in Appendix D, were included in the initial data set for the genetic algorithm sampling. A total of 7,371 such cases were found. As before, each variable was represented with an 8-bit string, effectively transforming continuous variables into discrete variables with 256 possible settings. Each case was mapped to the binary settings which mostly closely approximated its parameter values. A full factorial sampling of the space at this resolution would require somewhat more than 10^{118} cases.

The fitness function used was the same as for the 9-dimensional example:

$$ObjFunc = \sum_{i=1}^{N} w_i |C_{M,i}|$$

$$w_i = \begin{cases} 10 & \text{if } |C_{M,i}| > 0.1 \\ 1 & \text{otherwise} \end{cases}$$

A record of every case ever analyzed, and the associated results, was kept. The best 500 cases were selected and subjected to the genetic algorithm operators – tournament selection, crossover, and mutation – to create new cases for the next population. As before, the crossover rate was 70% and the mutation rate was 10% to encourage exploration of the design space. The case-creation process would continue until 500 new cases, which did not match any previous cases, had been generated. Those 500 new cases would then be analyzed using Cart3D and the results added to the case records.

Forty iterations of this process were performed, requiring the evaluation of 20,000 cases at each of 4 flight conditions. 2,370 of the cases analyzed, roughly 12%, had pitching moment coefficients within ± 0.1 for all 4 flight conditions. The algorithm took quite a few rounds to find useful test cases, but as more good cases were identified the rate increased. No test cases were found in the first 13 batches (6,500 cases); the final 5 rounds averaged just over 185 new test cases per batch of 500, or 37%. The search for test cases was curtailed after 40 batches because the 2,370 cases available at that point were felt to be sufficient.

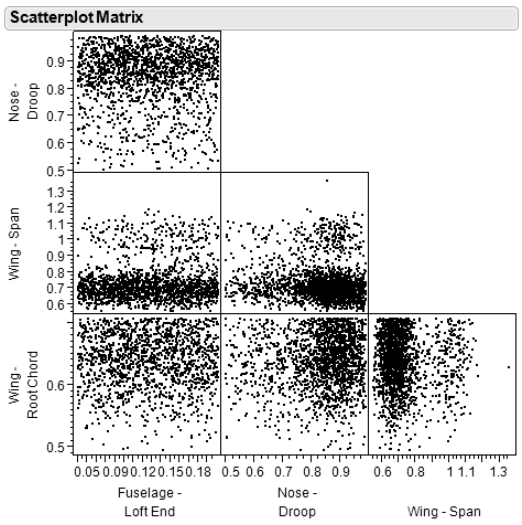

Figure 43: Partial Scatterplot for Selected Cases

6.2.2.1 Investigation of Test Cases

The accumulated test cases were investigated using JMP as a sanity check. Scatterplot matrices, which are a method of visually identifying trends and patterns in a data set, were generated for all 49 variables to see what deductions could be from the distribution of test cases. A partial scatter plot showing 4 variables is given in Figure 43.

A scatterplot is a set of 2-variable distributions. For example, the uppermost block in Figure 43 displays cases with the Fuselage Loft End value as the abscissa or horizontal component and Nose Droop as the ordinate or vertical component. Each black dot in this block represents a single configuration. Each configuration is plotted in every block. The distribution of dots can reveal trends in the data, such as regions with unusually dense or sparse sampling.

For example, for cases in the test set – i.e., cases which were found to have small pitching moments at all flight conditions – it was likely that the Nose Droop value was at the high end of the range: 1,908 of the 2,370 test cases have Droop values of 0.8 or above, which indicates that for most of these test configurations the tip of the nose was close to the bottom of the vehicle. Likewise, most of the selected test cases had small Wing Span Fraction values and large Wing Root Chord Fraction values. In contrast, there was no trend visible with respect

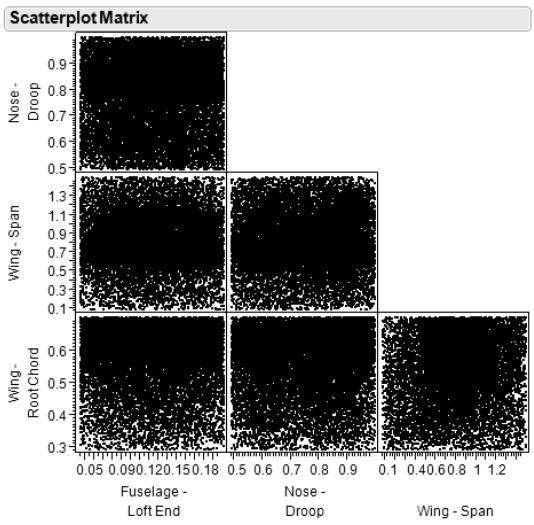

Figure 44: Partial Scatterplot for All Evaluated Cases

to the Fuselage Loft End parameter.

Because genetic algorithms were used to select these test cases, the cases were not independent, so it was possible that clustering behavior might appear that stemmed from the way cases were selected rather than the underlying behavior of the response. To determine whether such spurious clustering was likely to be a problem in this manner, another scatterplot was generated using the full set of 20,000 cases which were evaluated during the genetic algorithm search. A partial scatter plot showing the same 4 variables is given in Figure 44. Each black dot represents a single configuration; each configuration is represented in every box.

In Figure 44, the design space is shown to be sampled fairly thoroughly. No regions were left unsampled, although some regions were sampled more thoroughly than others. For example, in the lower-left scatterplot, it can be seen that there were more cases with high Wing Root Chord Fraction values than with low Wing Root Chord Fraction values. This is indicated by the dense distribution of black dots in the upper region of the scatterplot, with almost no white space visible, and mild increase in white space visible in the lower region of the scatterplot.

The thorough sampling observed in Figure 44 suggested that the cases which were evaluated sampled the design space fairly thoroughly, and thus that inferences about the design

space could be safely drawn from the distribution of test cases. Parameters which tended to take high values included Nose Top Curvature 2, Nose Bottom Curvature 2, Nose Droop, Wing Root Chord Fraction, Wing Outboard Taper Ratio, Wing Dihedral, Wing Maximum Camber Location, Wing Airfoil Thickness-to-Chord Ratio, and Wing Leading Edge Radius Parameter. Parameters which tended to take low values included Vehicle Scale, Wing Span, Wing Twist, Wing Incidence, Wing Camber, Wing Maximum Thickness Location, and Vertical Tail Leading Edge Sweep.

Curiously, there were two strong clusters of cases observed for Nose Spatularity Ratio; one group had values at or near the minimum of the range, while another cluster had values near the midpoint of the range. Using JMP, when points are selected within one scatterplot those points are highlighted in all other scatterplots, allowing the user to see the distribution of those points with regard to other parameters without having to generate a new set of plots. When the cases with larger Spatularity values were highlighted, it was revealed that these cases almost exclusively have very low Nose Fineness Ratios. Conversely, cases with small Spatularity values tended to have higher Nose Fineness Ratios, although some cases with both small Spatularity values and small Fineness Ratios were observed as well.

6.2.3 Generation of Low-Fidelity Data

Before sampling began, a decision had to be made as to whether APAS could be used directly as the low-fidelity data source, or if it had to be replaced with a surrogate model. Section 5.1.2.1 established that when multi-fidelity methods are used, contour-based sampling required such a large quantity of low-fidelity estimates that supplementary surrogate modeling could be a necessity. As in the 9-dimension example given earlier in this chapter (Section 6.1.1), a large space-filling set of cases was analyzed with APAS and a neural network fit to the results. Because the per-evaluation cost of APAS was so low, a very large space-filling data set was generated. However, it was uncertain whether all of the data would be necessary. Furthermore, the portion of the data that might be used was unlikely to be a regular fraction of the whole. To account for this, a sampling approach was sought such that even arbitrary fractions of the sample set would still have good space-filling characteristics.

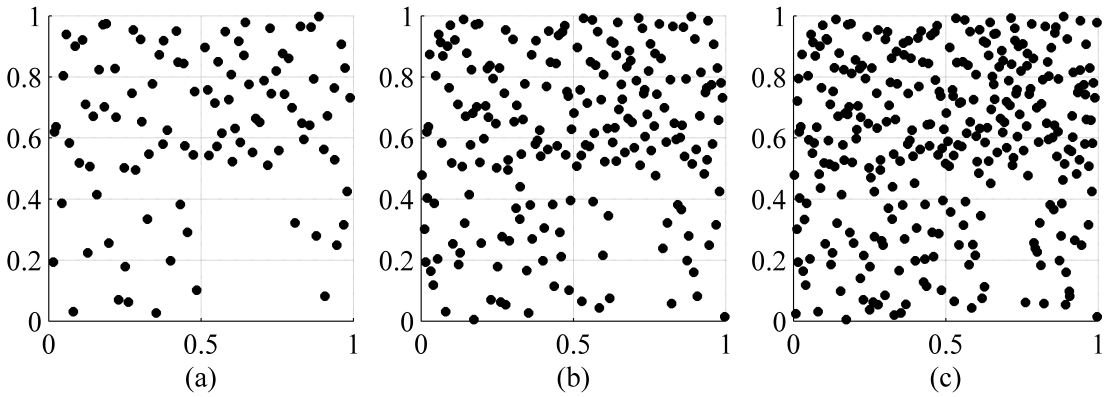

Figure 45: Subsets of a Sobol Sequence

This search led to the use of Sobol sequences.

6.2.3.1 Sobol Sequence for Space-Filling Samples

To allow a subset of the data to be used without sacrificing the space-filling qualities, the Matlab function **sobolset** from the Statistics Toolbox was used to generate data using a **Sobol sequence**. A Sobol sequence is a series of quasi-random numbers that have good space-filling properties.[21, 114, 180] The sequence can be generated for relatively little computational effort and subsets of the sequence also have relatively good space-filling properties.

Figure 45 shows how a Sobol sequence can be sampled sequentially without loss of space-filling characteristics. Figure 45a depicts the first 100 samples from the sequence. Figure 45b plots the first 200 samples, and Figure 45c plots the first 300 samples. Thus, samples from a Sobol sequence can be added progressively.

Note, however, that in all three sample sets, a gap is left unsampled near (0.6, 0.3) and the upper region is somewhat more heavily sampled than the lower. These results demonstrate that the **sobolset** function requires some degree of oversight by the user. In addition to the number of dimensions desired, the user can input values for two free parameters ("skip" and "leap") which control which numbers in the sequence are used to create samples. By varying these free parameters, new sequences can be generated, but these sequences sometimes have poor space-filling characteristics. A few sample Sobol sequences were generated using the commands:

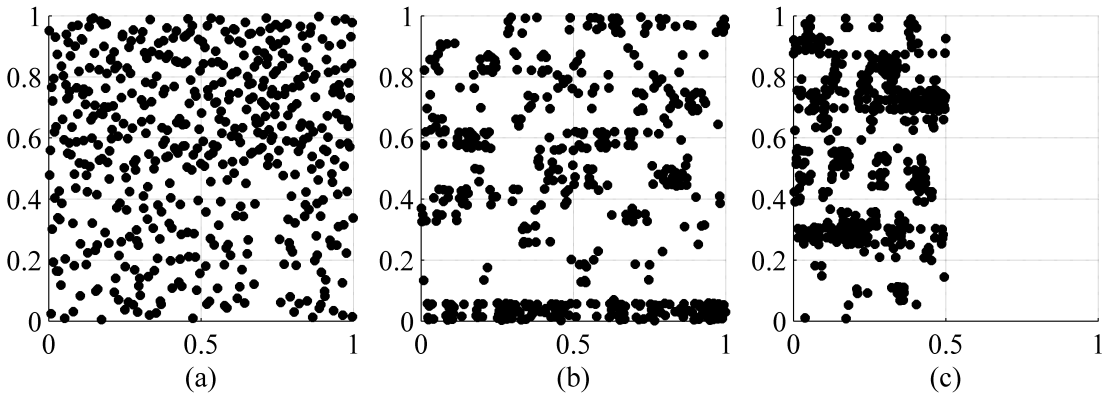

Figure 46: Distribution of Points from Sobol Sequences

- p = sobolset(2,'skip',skip,'leap',leap);
- p = scramble(p,'MatousekAffineOwen');
- POINTS = net(p,500);

Here, *skip* and *leap* are user-defined Matlab variables. Once Matlab has generated the sequence, it will skip the first *skip* points in the sequence and then use every $leap^{th}$ point as a sample.[121]

Three Sobol sequences are demonstrated in Figure 46. Figure 46a uses a *skip* value of 66 and a *leap* value of 32 to select 500 points; these are the same parameters that were used to generate the sample sets in Figure 45. Figure 46b is based on a *skip* value of 36 and a *leap* value of 16. The samples from this sequence exhibited significant clumping, and the lower edge of the space was sampled heavily while other regions were neglected. Figure 46c is based on a *skip* value of 42 and a *leap* value of 29. Once again, clumping was observed; in this example, however, the samples did not address the full horizontal range of the space.

These observations were the reason why Sobol sequences were not selected for most space-filling sampling in this research: there was too much risk that the Sobol sequence that was generated would have poor space-filling characteristics, which would adversely affect the experiment. However, this risk was considered acceptable for generating training data from APAS when the per-analysis cost was cheap and only a portion of the total samples

would be used. Checks were added to ensure that the sequence sampled the full range of each parameter.

6.2.3.2 Fitting Surrogate Models to APAS Data

After the Sobol sequence was created, APAS was used to analyze the points. The results were then modeled using the neural network tool BRAINN (described in Section 5.4.1). At first, the tool was applied to the full set of 500,000 cases, which made the fitting process quite slow and time-consuming. It was found that networks trained on 100,000 cases were equally accurate and could be trained much more quickly.

Because the APAS model would not include control surface deflections and only zero-sideslip flight conditions were being simulated, it was expected that once again any lateral responses estimated by APAS would be spurious. No surrogate models were trained to reproduce lateral responses from APAS. Instead, the primary goal of the low-fidelity surrogate models was to accurately predict pitching moment. A partial list of the goodness-of-fit metrics for the C_M models at each flight condition is given in Table 8. These metrics included the distribution of the Model Fit Error (MFE), which measured how well the model fit the training data, and the distribution of the Model Representation Error (MRE), which measured how well the model fit new data points that were not used in training. These error distributions were described by the mean (μ) and standard deviation (σ) of the observed prediction errors.

The two supersonic surrogates had good performance for the most part. Both the training and test R^2 values were very close to 1. The Model Fit Error, which quantified how well the surrogate model reproduced its training data, had a mean close to zero and a standard deviation that was not overly large, if a bit larger than might be desired. The same was true for the Model Representation Error, which quantified how well the surrogate reproduced a separate set of test data which was not used in its training.

The subsonic surrogates were not quite as accurate. The test R^2 values were below 0.9, which may be cause for concern, and the standard deviations of both the Model Fit Error and Model Representation Error were significantly larger. It was decided to proceed

Table 8: Goodness of Fit Metrics for Surrogates of APAS Data

	R^2 Training	R^2 Test	MFE μ	MFE σ	MRE μ	MRE σ
Mach 0.3 α 15°	0.927	0.844	-2.1×10^{-6}	0.581	0.0160	0.901
Mach 0.8 α 0°	0.946	0.812	3.2×10^{-8}	0.317	0.00846	0.648
Mach 2.5 α 15°	0.997	0.995	1.6×10^{-8}	0.114	0.00130	0.140
Mach 2.5 α 40°	0.998	0.998	5.8×10^{-8}	0.242	0.00338	0.255

with these surrogates rather than trying to improve the models; this would serve as a test to determine the sensitivity of the method with respect to the accuracy of the low-fidelity data. In effect, a less-accurate source of data was being used for the two subsonic conditions.

Once the low-fidelity surrogates were completed, the main effort of this experiment could begin in earnest.

6.2.4 Null Hypothesis: Space-Filling Samples

The null hypothesis for this experiment was that space-filling sampling would result in the most accurate surrogate models for the given test points. Space-filling sampling in general, and Latin hypercube sampling in particular, has been known to be an effective approach for understanding and modeling response behavior.[126] Such sampling is particularly useful when, as Cioppa and Lucas put it, "there may be multiple responses of interest and little *a priori* knowledge about the forms that the response function may take."[30]

As addressed in Section 2.4.1, space-filling methods usually select all samples simultaneously, forcing the user to decide in advance how many samples would be necessary. Qian's method of nested Latin hypercubes[152] allows some flexibility in this regard: a nested Latin hypercube (NLHC) contains multiple space-filling subsets, referred to as levels, which give

the user a variety of potential sample set sizes without losing the space-filling quality. However, the number of cases in each space-filling subset grows at a geometric rate, with the smallest possible growth rate being a doubling of size at each level. If more than a handful of levels are required, the rate of growth can quickly become significant. This was highlighted during the initial RBS effort described in Chapter 2: when the 8,000-case subset was insufficient with regard to model accuracy, the researchers had to run another 8,000 cases to produce another space-filling data set, doubling the computational effort.

Consider the nested Latin hypercube results depicted in Figures 40, 41 & 42. The plotted results include sample sets of 500, 1,000 & 2,000 cases; the next level of that NLHC would be 4,000 cases, a significant increase compared to the number of cases in the competing data sets. It would be preferable if the step size between space-filling sets was more uniform, since this would allow the user a greater degree of granularity with respect to sampling size.

6.2.4.1 Sliced Latin Hypercubes

Qian has also proposed **Sliced Latin Hypercube** designs,[153] which are intended for problems with one or more discrete variables. For the purpose of clarity, the following discussion will assume that only one discrete variable is present.

For each possible setting of the discrete variable, one hypercube is generated. This hypercube, referred to as a "slice," has good space-filling qualities with respect to the continuous variables. When all slices are combined, the resulting sample design is also a true hypercube, with good stratification over every dimension.

To illustrate the concept, consider sampling along a single continuous variable with 4 slices, each slice consisting of 4 samples. Each input variable would be split into 16 equal "bins". The combined hypercube would place 1 sample in each bin. The bins would be grouped into sets of 4, and each slice would put a single sample in each group.

For example, one group would consist of bins 1, 2, 3 & 4; another group would consist of bins 5, 6, 7 & 8, etc. The first slice (i.e., the first hypercube in the set) would include one bin from each group, such as $(3, 5, 11, 13)$. The second slice would also include one bin from each group, such as $(1, 6, 9, 16)$. No bin would appear in more than one slice, and each slice

would include only one bin per group.

After the algorithm to create sliced Latin hypercubes was implemented, a number of two-dimensional sliced Latin hypercubes were generated and investigated visually, with regard to both the distribution of points in each slice and the distribution of points in the combined set. Based on this admittedly qualitative examination, it was observed that some hypercubes exhibited a tendency toward clumping, with some regions being sampled more densely than others.

To reduce the risk of clumping and sample the space more evenly, a variation of sliced Latin hypercubes was devised. This variation would combine independently generated hypercubes rather than permutations of a common core. Because the overall sample set was composed of multiple unrelated hypercubes, it was referred to as a "**stacked Latin hypercube**."

6.2.4.2 Stacked Latin Hypercubes

By now, the reader should be aware that, due to the computational resources available to this effort, there was low emphasis placed on finding elegant solutions. It was likely that, by the time a complex approach could be implemented, a simpler brute-force method might already have accomplished the task at hand. This mindset informed the method of generating stacked Latin hypercubes.

To create a stacked Latin hypercube, the user first determines the desired step size and total number of cases. The total number of cases should be an integer multiple of the step size. For this effort, it was decided that a step size of 500 cases would be a good compromise between granularity and simplicity; the total number of cases was set to 8,000, which was expected to be close to or in excess of the largest viable data set for Kriging models. Thus, 16 hypercubes, each with 500 cases, would be combined to create the overall set of 8,000 cases. The challenge was to select a set of 16 hypercubes such that the combination of all cases was well-spaced.

A pool of 800 Latin hypercubes was generated, each of which included 500 cases of 49 dimensions. This pool size might seem small to researchers more accustomed to Monte

Carlo analyses using tens of thousands of cases, but such instincts can be misleading. The pool was set at 800 hypercubes because that value offered 50 hypercubes for each 1 being selected.

A selection of k members from a pool of n options, when the order of selection is unimportant, is known in probability theory as a "**combination**."[76] The number of possible combinations, given n and k, can be calculated by:

$$C_k^n = \frac{n!}{(n-k)!\ k!} \tag{36}$$

Here, ! indicates the factorial of the number, calculated as:

$$n! = n\,(n-1)\,(n-2)\cdots(1) \tag{37}$$

By convention, 0! is equal to 1. The number of possible combinations can grow much faster than might be expected. For example, if the objective was to select 3 options from a pool of 10, the number of possible combinations would be $\frac{10!}{3! \times 7!}$ which is equal to $\frac{3,628,800}{6 \times 5,040}$ or 120 combinations. Due to the nature of factorials, the number of combinations increases very rapidly with the size of the pool.

Although the direct calculation of 800! in common programs like Microsoft Excel 2007 or Matlab R2010b returns an answer of infinity – factorials grow *very* rapidly, and 100! is on the order of 10^{157} – the inspection of Equation 37 shows that when n is much larger than k, many of the terms in $n!$ and $(n-k)!$ will cancel out, resulting in a numerator and denominator that are much less elegant but far more computationally tractable:

$$C_{16}^{800} = \frac{800!}{784! \times 16!} = \frac{785 \times 786 \times \cdots \times 799 \times 800}{16!} = 1.16 \times 10^{33} \tag{38}$$

Thus, even a pool of 800 hypercubes will offer a very large number of possible combinations.

Genetic algorithms had been used for other portions of this research effort (see Section 4.10.6). The generation of a stacked Latin hypercube lent itself directly to the use of genetic algorithms. The "chromosome" for this problem consists of a binary string 800 bits long, with one bit for each hypercube in the pool. A 1 would indicate that that hypercube was included, while a 0 would indicate that the hypercube was excluded. The number of hypercubes included was constrained sum to 16, i.e. there must be 16 hypercubes for each

population member so that the total number of points in the stacked Latin hypercube would be 8,000.

As described on page 115, genetic algorithms are well-suited for problems which are discontinuous, highly-dimensional, noisy and/or multimodal. The creation of a stacked Latin hypercube was known to be highly-dimensional, since there were 800 dimensions, and discontinuous: each member of the pool of hypercubes had to be either included in its entirety or excluded, and could not be partially included.

The population size for this optimization was 500 members, with a probability of crossover of 70% and a probability of mutation of 5%. Rather than recording the performance of every population member ever analyzed, as in Section 4.10.6, the 10 best members of each population were carried over to the following population unchanged. Tournament selection, crossover, and mutation operations were then used to create the other 490 population members.

The fitness function for the GA was the **Euclidean maximin distance**, which is the smallest Euclidean distance between any two points in the set. This distance is a common metric to assess how well-spaced a set of cases are,[30, 132, 147, 185] and in fact the function *lhsdesign* from the Matlab Statistics Toolbox, which generates Latin hypercubes, attempts to maximize this metric.[120]

The Euclidean distance between two points x_1 and x_2 in P dimensions is calculated as follows:

$$d\left(x_1, x_2\right) = \sqrt{\sum_{i=1}^{P} \left(x_1^{(i)} - x_2^{(i)}\right)^2} \tag{39}$$

Here, $x_1^{(i)}$ is the i^{th} component of x_1.[132]

Each member of the population was a particular combination of 16 hypercubes from the pool. The members were evaluated with the Euclidean maximin distance; the larger this value, the farther apart the two closest points in that set of cases, and the better those cases were spread out throughout the space.

After 100 iterations, the best stacked hypercube had a maximin distance of 1.68. To evaluate this result, a comparison was made using Latin hypercubes. A set of 50 Latin

hypercubes, each of which had 49 dimensions and 8,000 cases, was generated using Matlab's *lhsdesign* function and evaluated using the maximin metric. The average maximin distance for these hypercubes was 1.56 and the best hypercube had a maximin score of 1.61. These results showed that the stacked Latin hypercube approach could produce a combined set with better space-filling characteristics than a single Latin hypercube designed to have good maximin spacing. In light of this performance, the genetic algorithm was curtailed at that point. The entire optimization took roughly 17 hours on a shared computing resource with four 2.66 GHz processors and 4 gigabytes of RAM, although there is no way to determine whether any other users were accessing the resource at the time.

Good performance was demonstrated for the full stacked set, but as yet there was not evidence that a subset of the full stacked set would also have desirable space-filling characteristics. A sorting algorithm was developed which took the 16 hypercubes which make up the stacked Latin hypercube and determined the best order in which to use them to maximize the maximin distance at each level. Thus, the first hypercube would be the one that has the largest maximin score out of the 16 hypercubes in the stack. The remaining 15 hypercubes were one-by-one combined with the first; the one which produced the best 2-hypercube maximin score was designated the second hypercube. The third hypercube is selected by combining the 14 remaining hypercubes with the first and second, and so on.

6.2.4.3 Evaluating the Quality of Competing Space-Filling Approaches

The resulting stacked Latin hypercube was then compared against standard Latin hypercubes and nested Latin hypercubes. Each space-filling approach was assessed for various numbers of cases, from 500 up to 8,000 cases, in steps of 500. For each number of cases, at least 500 standard Latin hypercubes were generated using Matlab's *lhsdesign* function and assessed using the Euclidean maximin distance. The standard deviation of the resulting distances was less than 0.025 for all levels.

The nested Latin hypercube cases had a minimum size of 500 cases and grew by a factor of 2, so each NLHC had space-filling levels of 500, 1,000, 2,000, 4,000, & 8,000 cases. At least 500 NLHCs were generated; each NLHC was then assessed at each level of cases. If a

particular number of cases did not correspond to a space-filling set, such as 1,500 cases, no assessment was made. The average Euclidean maximin distance at each level was calculated.

Table 9 gives the Euclidean maximin distance for the stacked Latin hypercube at each level, as well as the average results for the standard and nested Latin hypercubes. The most obvious aspect of this table is the gaps that result from the geometric growth of NLHCs, which are relatively small at first but become more significant at higher levels. Another interesting observation is the degree of similarity between the nested & sliced Latin hypercubes. Unlike the other methods, these two had maximin distances consistent to at least two decimal places for every point of comparison. It is believed that these results stem from the fact that both methods use random permutations of the smallest space-filling set to generate larger data sets. The use of permutations, rather than the creation of independent & unique hypercubes, may limit the potential of the designs with respect to optimal space-filling characteristics. It must be said, however, that both methods generate sample sets quite rapidly.

To determine the time required to generate a given sample design, each method was used to generate a set of 8,000 cases over 49 dimensions; this was repeated 1,000 times for each method, with the exception of the stacked Latin hypercube. The average time per standard Latin hypercube was 8.8 seconds, while the average sliced Latin hypercube time was a mere 0.38 seconds. Nested Latin hypercubes were generated in an average of 5.3 seconds. It was curious that the nested Latin hypercubes could be generated more rapidly than standard hypercubes despite the complexity of the nesting method. Upon review of the function description for *lhsdesign*, used to create the standard hypercubes, it was found that the function may iterate up to five times to improve the distribution of points according to the space-filling criterion, which by default is the Euclidean maximin distance. This may also explain why the standard hypercubes have a larger maximin distance than the NLHC cases, even at the smallest pool size.

Lastly, it was observed that the stacked Latin hypercube did in fact offer better space-filling characteristics as measured by Euclidean maximin distance. This indicated that the

Table 9: Minimum Spacing of Hypercubes (HCs) of Various Sizes

	500 Cases	1,000 Cases	1,500 Cases	2,000 Cases	2,500 Cases	3,000 Cases	3,500 Cases	4,000 Cases
Latin HC	1.83	1.76	1.72	1.69	1.67	1.65	1.63	1.62
Nested Latin HC	1.77	1.70	–	1.63	–	–	–	1.57
Sliced Latin HC	1.77	1.70	1.66	1.63	1.61	1.59	1.58	1.57
Stacked Latin HC	1.87	1.84	1.81	1.77	1.77	1.74	1.73	1.73
	4,500 Cases	5,000 Cases	5,500 Cases	6,000 Cases	6,500 Cases	7,000 Cases	7,500 Cases	8,000 Cases
Latin HC	1.61	1.60	1.59	1.59	1.58	1.57	1.57	1.56
Nested Latin HC	–	–	–	–	–	–	–	1.51
Sliced Latin HC	1.56	1.55	1.54	1.54	1.53	1.52	1.52	1.51
Stacked Latin HC	1.70	1.69	1.69	1.69	1.69	1.69	1.68	1.68

stacked Latin hypercube had very good space-filling characteristics while enabling progressive sampling with a linear rate of data set growth. The negative aspect of this approach was evident in the amount of effort required to assemble a good stacked Latin hypercube.

Although a stacked Latin hypercubes was computationally intensive to generate, the effort was rewarded with a set of samples that allowed smooth linear growth while retaining good space-filling qualities. This technique would form the null hypothesis, which asserted that space-filling samples would be the most effective sample distribution method for surrogate modeling.

6.2.4.4 Applying the Stacked Latin Hypercube

The cases from the stacked Latin hypercube were analyzed using Cart3D and grouped into the appropriate data pools. Recall that roughly 70% of the cases that were analyzed for the nine-dimensional problem were suitable for modeling, i.e. had converged for all 3 flight conditions. For this larger problem, roughly 90% of the analyses were found to be converged for all flight conditions, which suggests one of two primary explanations: either the Mach 2.5, α 0° flight condition accounted for the bulk of unconverged cases, since it was replaced by the Mach 2.5, α 0 & 40° conditions for this experiment, or the default values used to reduce the design space to 9 dimensions produced configurations that were more likely to exhibit poor computational convergence.

The various data pools were used to train Kriging models with linear trends and anisotropic Gaussian correlation functions. These surrogates were then evaluated using the test cases identified in Section 6.2.2. By comparing the predicted C_M values against Cart3D results, the Root Mean Squared Error (RMSE) would be calculated for each model. Those RMSE values are presented in Figure 47. For the most part, the results showed very limited improvement or mild degradation with increasing data pool size, which was unexpected. The model for C_M at Mach 2.5, α 15° showed an 11% improvement in RMSE for 4,000 cases compared to 500 cases, as seen in Figure 47c; all other C_M models became less accurate as more cases were added, at least with respect to the test cases.

The C_M surrogate models for the Mach 0.3 flight condition were investigated to identify why model improvement was minimal or negative. For those models, the $\hat{\beta}$ values for the underlying trend model – which are similar to the coefficients in a response surface model – showed mild convergence behavior as the available data pool grew, but no significant

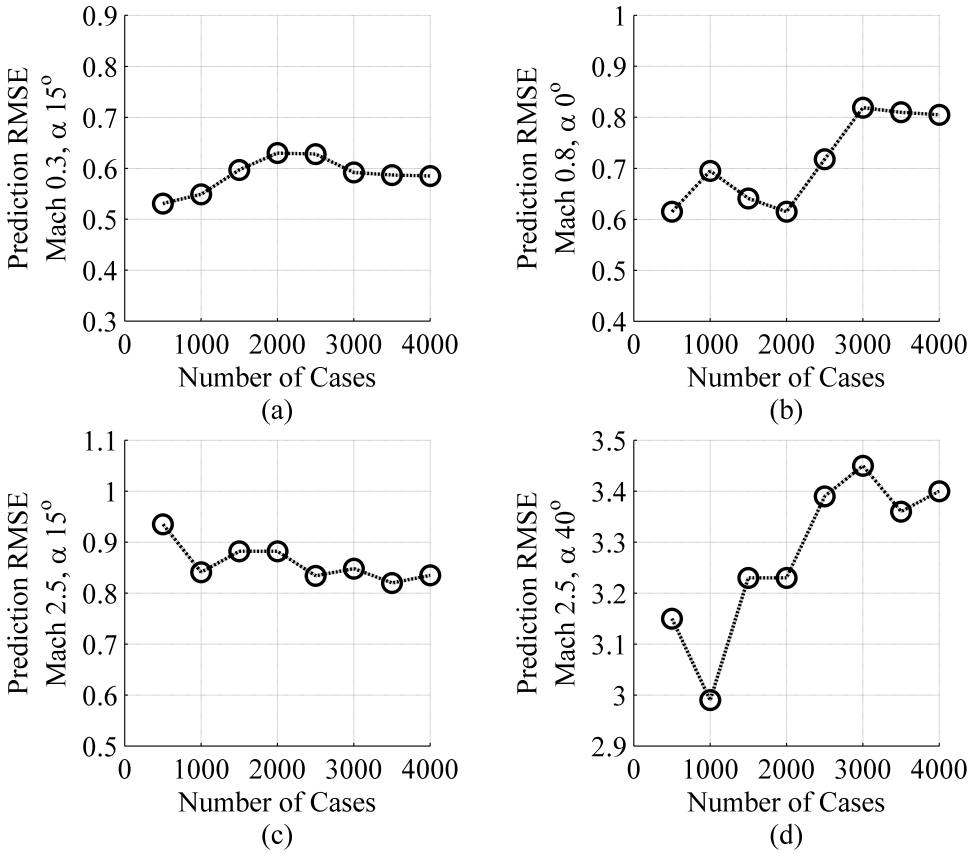

Figure 47: Prediction RMSE for Stacked Latin Hypercube Sampling

changes overall. Some weights progressively approached zero, indicating that as more data is available, the corresponding parameters were found to be less important in determining the response, but those weights were small to begin with and thus did not strongly affect the models. This observation indicated that the underlying trend that was fit to the training data remained approximately constant as more training data was added.

Recall that Kriging predictions depend on two components: the underlying trend model, which is similar to a response surface model, and the correlation-based correction term. The correction term is used to capture divergences from the underlying trend based on the degree to which other nearby data points deviated from that trend. When the Kriging surrogate model is fit to the training data, the coefficients for the trend and the correction term are optimized to best fit the observations.[108] The Kriging surrogate produced by the DACE toolbox utility **dacefit** is represented by a data structure which includes the trend model

coefficients as well as various other parameters relevant to the Kriging model.

When the coefficients for the correlation term were investigated, they were found to be unchanged from the initial guess values. This can occur when the algorithm which optimizes the correction coefficients could not identify any new values that would improve the model accuracy. It was believed that this behavior was due to the relative sparsity of the overall sampling. Even for the larger data pool sizes, the points may not have been close enough together that the correction term is producing any consistent improvement.

This was tested by evaluating the correlation between the test cases and the training data sets. The Kriging models in this study all used an anisotropic Gaussian correlation function, which was previously defined in Equation 3 on page 80. The implementation of this function within the DACE toolbox uses a slightly different notation:

$$k(u,v) = \prod_{i=1}^{n} exp\left[-\theta_i \left(|u_i - v_i|\right)^2\right] \tag{40}$$

In this version, the θ_i term, which indicates the correlation coefficient for dimension i, is multiplicative rather than divisive. u_i and v_i represent the i^th component of points u and v, respectively. When the Kriging models were fit to the data, the θ_i values were initialized at a value of 10 and allowed to vary between 0.1 and 20. Smaller θ_i values would indicate that there is correlation between cases which are father away.

It should be noted that the points u and v must be normalized before correlation can be calculated. The DACE toolbox function **dacefit** transforms the input cases and responses so that the normalized variables have a mean of 0 and a standard deviation of 1.[108] Each variable is normalized by subtracting the mean of the known values for that variable and then dividing by the observed standard deviation of that variable:

$$S_{:,j} = \frac{\left(\bar{S}_{:,j} - \mu\left(\bar{S}_{:,j}\right)\right)}{\sigma\left(\bar{S}_{:,j}\right)} \quad j = 1, \cdots, n \tag{41}$$

Here, $\bar{S}_{:,j}$ is the set of observed values for the j^{th} variable in the set of points S. $\mu\left(\bar{S}_{:,j}\right)$ and $\sigma\left(\bar{S}_{:,j}\right)$ represent the mean and standard deviation of the observed values, respectively. The response values are normalized in the same manner.

Once the test cases had been normalized, the correlation between each test case and the training cases could be calculated. These correlation values can range from 0 to 1, with 0

indicating no correlation and 1 indicating perfect correlation. For the 500-case data set, the largest correlation value between any test case and any training case was 1.6×10^{-170}. When the 4,000-case set is used instead, this correlation increases to 4.2×10^{-151}. Effectively, the training data did not provide any knowledge about how far the test cases were likely to deviate from the underlying trend. As a result, the correlation term of the Kriging model went to zero for all test cases, and the surrogate became a simple least-squares linear fit.

Based on these results, it appears that the sampling is too sparse for the correlation term to affect the Kriging prediction. This reduces the Kriging model to the underlying trend model which is fit to the data using the generalized least-squares solution.[108]

Note that this observation did not invalidate the alternative hypothesis that the proposed approach would improve model accuracy. When a model is fit using a least squares method, it attempts to fit every data point equally well. If the model is fit to space-filling samples, the entire design space is given equal weight. Conversely, if regions of desirable response behavior can be identified and emphasized by adaptive sampling, the resulting models would exhibit better accuracy in those regions because the over-representation of such cases artificially weights those regions as being more important. The evaluation of the alternative hypothesis therefore went ahead as planned.

6.2.5 Alternative Hypothesis: Multi-Fidelity contour-based Sampling

Opposing the null hypothesis in this experiment was the alternative hypothesis, which asserted that the multi-fidelity contour-based sampling approach would produce surrogate models with better predictive accuracy for the cases of interest. Once again, a linear underlying trend model and an anisotropic Gaussian correlation function were used. This illuminated the effects of increased space-filling sampling while ensuring that both the space-filling and adaptively-sampled methods began from an equal footing.

As previously mentioned, the time and computational effort required to train or apply a Kriging model grows geometrically as the number of data points increases.[136] In the previous experiment, Kriging models were trained using no more than 1,000 cases. In deference to the larger data sets that would be modeled for this study, the number of

cases per batch was reduced: whereas for the nine-dimensional study 70 cases were selected per batch, here only 15 cases would be selected per batch. This decision was intended to reduce the amount of time per batch. Smaller batches also meant that the algorithm would be updated more often, potentially leading to better selection of samples as each response was modeled more accurately.

To evaluate how the size of the warm start would affect the performance of the proposed approach, the algorithm was initialized using various levels of the stacked Latin hypercube, which contained space-filling sets of data in multiples of 500 cases. The first 6 of these sets, up to a maximum of 3,000 cases, were used to investigate how the integrated algorithm would be affected by increasing the quantity of data available. It was expected that the integrated algorithm would be progressively more effective as more space-filling data were used, due to the greater information about response behavior.

6.2.5.1 Effects of Initial Sampling Size

A simple study was performed to determine how the size of the initial data pool, or "warm start," would affect the performance of the adaptive sampling algorithm. A larger initial data pool might improve the surrogate model that would be used by the adaptive sampling algorithm, leading to a better assessment of the available candidates and thus a larger improvement in predictive accuracy. However, a larger data pool corresponds to higher surrogate model training costs. As a result, evaluating a set of candidates would take longer. This study was intended to determine the point at which the increased evaluation costs began to outweigh the improved candidate selection performance.

Five different sizes of warm start were investigated. The sizes were based on the first 5 levels of the stacked Latin hypercube, and corresponded to training data sets of around 500, 1,000, 1,500, 2,000 and 2,500 cases. Using each warm start, 30 new samples were selected (in batches of 15). To ensure a fair comparison, all warm starts were provided with the same candidates and test points. New candidates and test points were used for each round.

The adaptive sampling algorithm featured 3 parameters that could be adjusted by the user: the number of candidates to be evaluated, the number of test points to be used in

the evaluation, and the required probability of interest (POI) that would be used to screen out uninteresting candidates. Each of these parameters affected the computational effort required to select a new sample. Identifying optimum settings for these parameters was left for future research, especially as the "optimum" settings were likely to be problem-dependent. Instead, 3 different sets of values were selected for use with the 49-dimensional problem. Each set of values would be used to select 5 samples out of every batch of 15 points.

The first 5 points out of every batch were intended to be exploratory while still keeping sample selection times low: 1,000 candidates, 1,500 test points, and a required POI of 0%. Because the POI criteria was a "greater-than" and not a "greater-than-or-equal," a POI of 0% would still eliminate some candidates. The number of candidates eliminated would depend strongly on the estimated prediction uncertainty: when uncertainty was larger, more candidates would have a POI value greater than zero.

The second 5 points were selected from a larger set of candidates (3,500), which would be evaluated with a greater number of test points (3,500). If the POI requirement were not adjusted, this would have led to a significant increase in analysis time. To mitigate this effect, a higher POI requirement (1%) was used. Depending on the prediction uncertainty, this POI requirement would typically disqualify 65-85% of the candidates for this problem, so the time required per sample selection did not increase excessively.

The final 5 points in each batch were also selected using 3,500 candidates and 3,500 test points. The 1% POI requirement of the previous 5 selections was fairly restrictive, so that requirement was relaxed partially to 0.5% for these selections. This reduced POI requirement would still disqualify 55-80% of the candidates, again depending on the estimated prediction uncertainty.

Beginning with each warm-start set, these parameter schedules were used to select batches of 15 samples at a time. Those samples would be analyzed, and the new samples would be appended to the associated warm-start set. This procedure was repeated twice for each warm start, augmenting the initial space-filling cases with a total of 30 new samples. The sample selections were performed in Matlab 2011b on a shared computing

Table 10: Effects of Data Pool Size on Time Per Sample Selection (in minutes)

	500 Cases	1,000 Cases	1,500 Cases	2,000 Cases	2,500 Cases
1,000 Candidates 1,500 Test Points POI 0%	31	74	165	220	332
3,500 Candidates 3,500 Test Points POI 1%	48	103	233	265	392
3,500 Candidates 3,500 Test Points POI 0.5%	62	130	296	332	475

resource with 4 gigabytes of RAM and two Intel Xeon E5440 2.83 gigahertz processors, each of which had 4 cores. It should be noted that the computing resource was shared and other users may have been active during this effort, reducing the resources that were used for this effort.

The first quantity evaluated was the average time per sample selection. The reader may recall that the computational cost to build a Kriging surrogate model grows as $\mathrm{O}(N^3)$, where N is the number of training data points. As a result, larger sets of warm-start data were expected to take longer to select each sample. The time required to select a sample was grouped based on the associated settings for the algorithm parameters (number of candidate points, number of test points, & POI requirement). The average time required to select a sample under each set of algorithm parameters was then calculated for each warm start size. The results are displayed in Table 10.

As Table 10 shows, the average time required to select each new sample increased drastically as the size of the warm start grew. There was something of an aberration: the increases in average sample selection times between the "1,500 Cases" and "2,000 Cases" columns are

much smaller than the increases between any other pair of consecutive columns. It is possible that other users were accessing the shared computing resource during part of this study. A surge in competition for resources during the 1,500-case portion of the calculation and reduced competition during the 2,000-case portion could explain the observed results; however, the distribution of resources at any given time was not recorded, limiting the author's ability to substantiate this theory. In general, however, the observed results did show that the time & effort required to select a new sample grew rapidly.

The other aspect that was being investigated was whether the extra information led to better sample-selection behavior, evidenced by surrogates with improved predictive accuracy. A better initial surrogate would evaluate candidates more accurately and thus might do a better job of selecting candidates, resulting in more rapid improvement in surrogate model accuracy.

After 2 batches of points were selected using each of the initial data sets, the resulting Kriging models were evaluated using the test cases in order to assess their predictive accuracy. The resulting RMSE values are compared in Table 11. It was found that, as with the stacked Latin hypercube, there was not a strong or consistent relationship between the number of space-filling cases and the predictive accuracy of the model for sets of up to 2,500 cases.

More importantly, new samples could be selected more rapidly for the smallest data set than for the larger sets, which more than compensated for the reduction in available information. For example, the surrogate models based on 1,000 space-filling cases and two adaptively-selected batches had better predictive accuracy for the 2 supersonic flight conditions than those based on 500 cases and 2 adaptive batches. However, in the time required to select 2 batches of adaptive cases for the 1,000-case set (roughly 38 hours for 2 batches), it was possible to select *4* batches for the 500-case set (roughly 35 hours for 4 batches); when the extra 2 batches were included, models based on the 500-case set had better predictive accuracy. In light of these results, the larger data sets were abandoned and future sampling efforts would use the set of 500 space-filling samples as the warm start.

Table 11: Prediction Accuracy (RMSE) After Two Batches of Adaptive Samples

	500 Cases	1,000 Cases	1,500 Cases	2,000 Cases	2,500 Cases
Mach 0.3 α 15°	0.395	0.412	0.412	0.410	0.433
Mach 0.8 α 0°	0.793	0.886	0.754	0.717	0.675
Mach 2.5 α 15°	0.252	0.235	0.256	0.263	0.264
Mach 2.5 α 40°	0.925	0.719	0.800	0.805	0.791

6.2.6 Probability of False Positives

Although Root Mean Squared Error, or RMSE, is a relatively easy way to quantify the predictive accuracy of each surrogate, the result may be difficult to interpret beyond "smaller is better." To more clearly illustrate the accuracy of a surrogate, a new metric was developed. This metric uses RMSE to calculate the likelihood that a poorly-performing configuration will be mistakenly predicted to have good performance. This likelihood shall be referred to as the "**probability of a false-positive**" or **POFP**.

Root Mean Squared Error is an approximation of the variance of the observed prediction error. The "true" response for a given configuration, as calculated by Cart3D, can be approximated by a normal distribution with the predicted response value (0 in this case) as its mean and the RMSE of the surrogate as its standard deviation. Using the mathematical technique presented in Section 4.3.2, the portion of the distribution that lies outside of ± 0.1 can be calculated. Specifically, the user can calculate the likelihood that, if the surrogate predicts that some configuration has a pitching moment coefficient of exactly 0, the *true* pitching moment coefficient as calculated by Cart3D would be found to fall outside the range of $-0.1 < C_M < 0.1$. In other words, POFP estimates the likelihood that the

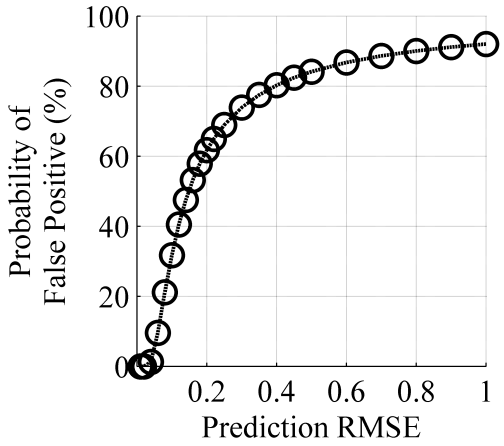

Figure 48: Probability of a False Positive for $|C_M|$<0.1

surrogate model would evaluate an uncontrollable configuration and pronounce it controllable. It stands to reason that smaller POFP values would correspond to better predictive accuracy.

The relationship between prediction RMSE and POFP for a constraint of $|C_M| < 0.1$ is shown in Figure 48. For prediction RMSE values below 0.05, the POFP is effectively zero. There is a large increase in POFP as RMSE grows, although the rate of increase is reduced once RMSE exceeds roughly 0.3. Clearly the objective would be to minimize POFP, and by extension minimize prediction RMSE. Large gains can be made once prediction RMSE falls below 0.3. As predictive accuracy increases, the likelihood of a false positive – that a poorly-performing configuration will be identified as having good performance – is reduced. It must be noted that POFP will depend, not only on RMSE, but on the response range of interest, and as a result Figure 48 is only valid for the current problem.

6.2.7 Evaluation of Accuracy for Pitching Moment

6.2.7.1 Mach 0.3, α 15°

Once the predictive accuracy was quantified for models of pitching moment coefficient, the results for each flight condition were plotted for visual comparison. The results for Mach 0.3, α 15° are plotted in Figure 49. The black squares mark the predictive accuracy of single-fidelity Kriging models that were trained using the space-filling stacked Latin hypercube.

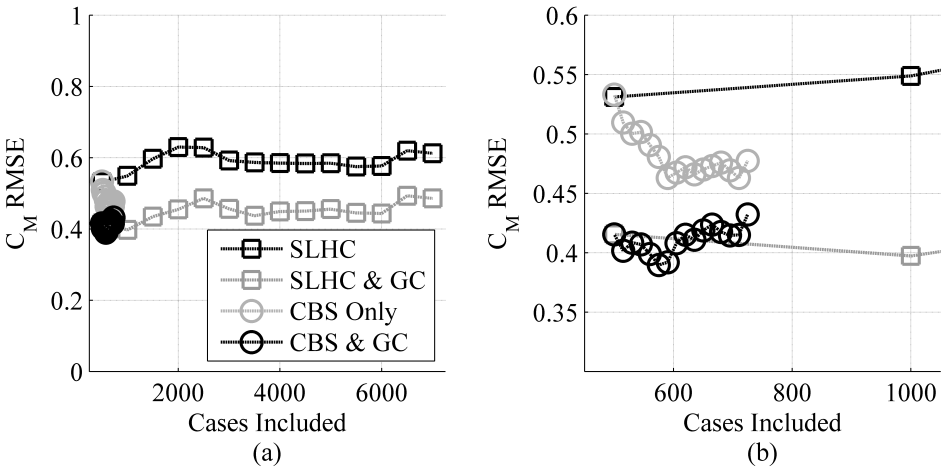

Figure 49: Predictive Accuracy for Pitching Moment at Mach 0.3, α 15°

Note that predictive accuracy did not improve as more samples were included; instead, as the size of the training data pool increased from 500 to 7,000 points, the predictive RMSE got *worse*, from 0.53 to 0.61.

The grey squares denote Kriging models that were trained using the same stacked Latin hypercube cases, but which also made use of APAS data via Ghoreyshi cokriging. Like the single-fidelity results, the accuracy of the models did not improve when more space-filling cases were included. The predictive RMSE values for the smallest and largest training sets were 0.42 and 0.49, respectively. On average, the incorporation of cheaper data reduced RMSE by about 0.13, or 25%.

The grey circles mark the performance of surrogate models trained with the adaptively-selected samples; these models were single-fidelity, trained only with Cart3D results. Note that the samples were *selected* using multi-fidelity contour-based sampling. The single-fidelity models were trained solely for the purpose of comparison. Initial rounds of sampling did demonstrate improved accuracy, from 0.53 to 0.46, although this improvement leveled off after six or seven rounds of sampling.

Lastly, the models produced by the proposed method are marked with black circles. These models showed some variation in behavior, but overall the results were the same as the space-filling cases: predictive accuracy degraded slightly as the training data set grew larger.

Table 12: Predictive Accuracy for Pitching Moment Coefficient at Mach 0.3, α 15°

	Number of Converged Cases	RMSE	POFP
Stacked LHC 500-Case Design	458	0.533	85%
Proposed Method 725 Samples	**651**	**0.432**	**81%**
Stacked LHC 1,000-Case Design	922	0.549	86%
Stacked LHC 7,000-Case Design	6,430	0.612	87%
Neural Network with Nested Latin Hypercube Design	11,417	1.47	95%

Over 15 rounds of adaptive sampling, the training data set grew from 500 to 725 samples and the prediction RMSE grew from 0.42 to 0.43. To translate this RMSE value into probability of false positive, if the latest surrogate model predicted some untested configuration to have a pitching moment coefficient of exactly 0 at this flight condition, there would be a 81% chance that an analysis with Cart3D would reveal the actual pitching moment coefficient to be $> \pm 0.1$. This is larger than would be preferred, but still an improvement over the space-filling approach: after 7,000 samples, the stacked Latin hypercube approach achieved an RMSE of 0.612 and the POFP would be 87%.

The prediction RMSE and POFP scores for a number of surrogate models are given in Table 12 to allow quick comparisons. The proposed method had smaller RMSE and reduced chance of a false positive. In contrast, single-fidelity surrogate models trained using stacked Latin hypercube cases grew mildly less accurate as larger data sets were used.

These results illustrated how the POFP metric could at times behave in an un-intuitive manner: although the prediction error was reduced by 15-20%, the POFP was only reduced

by 5-6%. The prediction error was large compared to the response range of interest, and so even a relatively large reduction in error did not correspond to a large impact on POFP. For smaller RMSE values, a similar reduction in RMSE would produce a larger reduction in POFP.

The neural network that was trained to emulate this response during the previous RBS effort[40] was also evaluated. This network was trained using results from a nested Latin hypercube, and was based only on Cart3D data. It was interesting to note that the neural network produced a prediction RMSE of 1.47 for these test cases – a value much larger than the prediction RMSE of 0.395 when the surrogate was tested with space-filling cases. This suggested that the neural network fit other regions of the space much more accurately than the current region of interest.

Overall, the results were mixed. The inclusion of cheaper data did produce a consistent improvement in predictive accuracy of roughly 25%. However, with regard to sample selection, neither space-filling nor adaptive samples were clearly more effective. Both approaches produced *reduced* accuracy as more samples were added. No matter which sampling approach was used, the DACE toolbox was unable to identify correlation weights which improved model accuracy over that of the underlying trend alone.

6.2.7.2 Mach 0.8, α 0°

Like Mach 0.3, the surrogate model that stood in for APAS at this flight condition had fairly large prediction error. The standard deviation of the Model Representation Error was roughly 0.65, which was large with respect to the width of the response region of interest.

The single-fidelity space-filling approach, using 7,000 cases, produced a prediction RMSE of 0.864 and a POFP of 91%. Combining space-filling samples and data fusion produced a mild improvement, particularly for larger data sets, but it was not as effective as it was for Mach 0.3, α 15°. Likewise, single-fidelity surrogates trained with the adaptively-sampled cases were almost indistinguishable from those trained with the adaptive samples and Ghoreyshi cokriging. After the end of sampling, the final surrogate model created by the proposed method had a prediction RMSE of 0.762, which corresponds to a POFP of

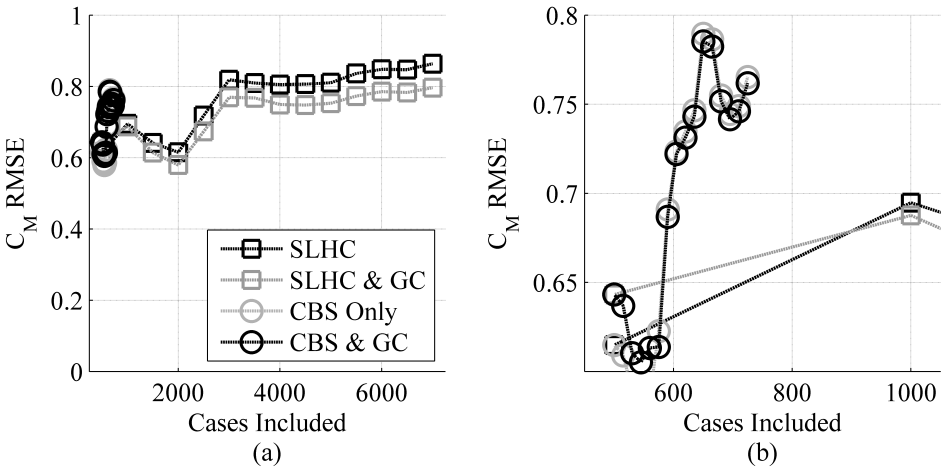

Figure 50: Predictive Accuracy for Pitching Moment at Mach 0.8, α 0°

Table 13: Predictive Accuracy for Pitching Moment Coefficient at Mach 0.8, α 0°

	Number of Converged Cases	RMSE	POFP
Stacked LHC 500-Case Design	458	0.615	87%
Proposed Method	**651**	**0.724**	**90%**
Stacked LHC 1,000-Case Design	922	0.695	91%
Stacked LHC 7,000-Case Design	6,430	0.864	96%
Neural Network with Nested Latin Hypercube Design	11,159	0.688	88%

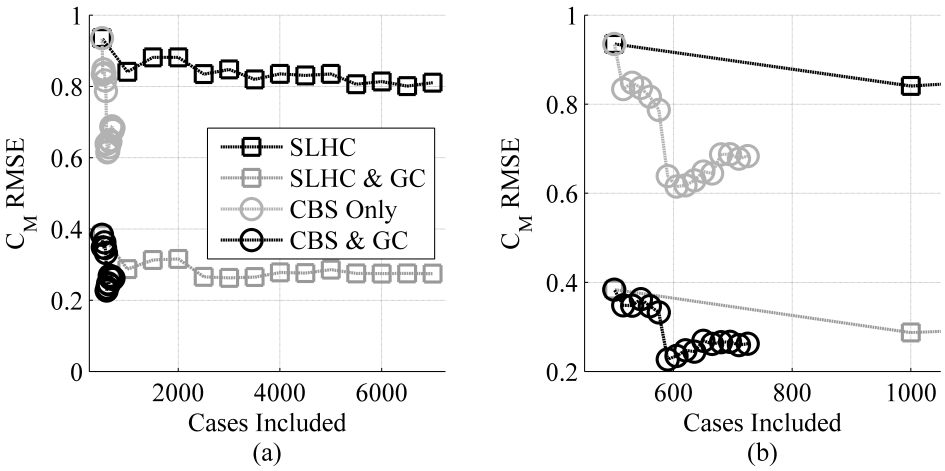

Figure 51: Predictive Accuracy for Pitching Moment at Mach 2.5, α 15°

90%. Note that as the number of samples increased, both sampling approaches became markedly less accurate. For further comparison, the RBS neural network used 11,159 cases and produced an RMSE of 0.688 for these test cases; for space-filling test cases, the RMSE was 0.274. These values are presented for easier comparison in Table 13.

The surrogates based on 2,000-4,000 cases, which demonstrated a sharp increase and plateau of prediction error, were investigated to determine the source of this degradation. A few patterns were observed: the trend coefficients for Loft Start and Loft End shifted from $O(10^{-4})$ to $O(10^{-2})$, while those for Vertical Tail Maximum Thickness Location and the Inboard Starboard Elevon Deflection decreased from $O(10^{-2})$ to $O(10^{-4})$ and $O(10^{-3})$, respectively. It should be noted that the magnitudes of these coefficients remained small at all times, and that identification of these coefficients as the cause for the change in predictive accuracy was tentative at best.

6.2.7.3 Mach 2.5, α 15°

The prediction accuracy results for Mach 2.5, α 15° are given in Figure 51. Here, data fusion made a very significant difference with regard to predictive accuracy, reducing the RMSE value from 0.935 to 0.384. Adaptive sampling improved the results from there, bringing the RMSE down to 0.262. The rate of improvement due to additional sampling was not as rapid as for Mach 0.3.

Surrogates trained with space-filling samples and Ghoreyshi cokriging are marked with grey squares. It was clear that data fusion made a powerful contribution for this response, reducing RMSE by roughly 60% in many cases. The grey circles denote single-fidelity surrogates trained with adaptively-selected samples. It bears repeating that these samples were selected using multi-fidelity contour-based sampling; the single-fidelity surrogates were only trained for performance comparison. The resulting surrogates did improve in accuracy much more rapidly than those trained with space-filling samples, reducing prediction error by about 30%.

The proposed approach, combining both data fusion and adaptive sampling, was the most effective for this response. With an RMSE value of 0.262, the probability of a false positive (POFP) for the proposed approach was 70%. The space-filling model based on 7,000 samples produced a prediction RMSE value of 0.810, resulting in a POFP value of 90%, while the surrogate based on 7,000 space-filling samples and Ghoreyshi cokriging produced a prediction RMSE of 0.274 (equal to a POFP of 72%). This was a much larger improvement in RMSE than at Mach 0.3, although most of the improvement appears to be due to the incorporation of APAS data. The neural network generated during the RBS project was trained with 10,458 space-filling cases and exhibited a prediction RMSE of 0.887 when applied to the test cases with small pitching moment coefficients. When evaluated for space-filling test cases, the RMSE of that neural network was 0.262. These results are given in Table 14.

6.2.7.4 Mach 2.5, α 40°

The prediction accuracy results for Mach 2.5, α 40° are given in Figure 52. Data fusion was very powerful for this flight condition as well, improving prediction RMSE from 3.15 to 1.23 for the 500-case training set as shown by the grey squares. There was still ample room remaining for improvement. The grey circles show the performance of single-fidelity surrogates trained using adaptively-sampled cases; improvement was *much* more rapid than for space-filling cases, although these models did not attain the predictive accuracy of the multi-fidelity surrogates. When both techniques were combined (indicated by the black circles), the prediction RMSE of a model incorporating data fusion was improved to 0.724

Table 14: Predictive Accuracy for Pitching Moment Coefficient at Mach 2.5, α 15°

	Number of Converged Cases	RMSE	POFP
Stacked LHC 500-Case Design	458	0.935	91%
Proposed Method	**651**	**0.262**	**70%**
Stacked LHC 1,000-Case Design	922	0.841	91%
Stacked LHC 7,000-Case Design	6,430	0.810	90%
Neural Network with Nested Latin Hypercube Design	10,458	0.887	91%

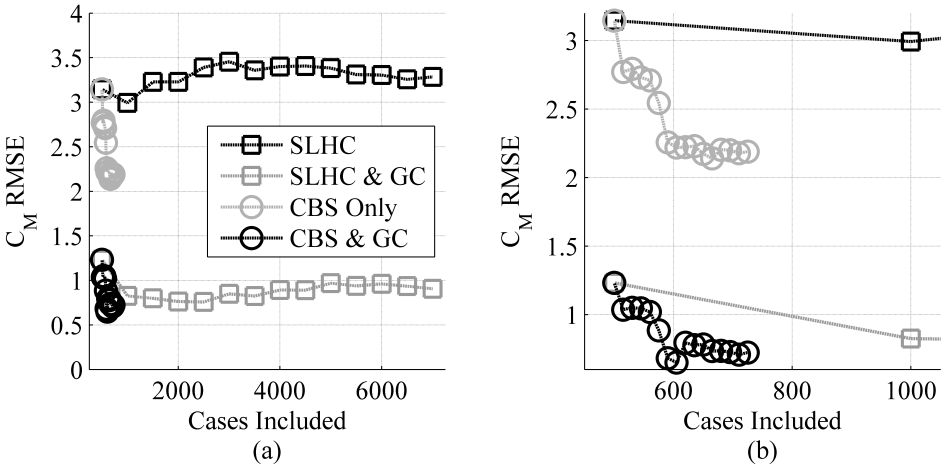

Figure 52: Predictive Accuracy for Pitching Moment at Mach 2.5, α 40°

Table 15: Predictive Accuracy for Pitching Moment Coefficient at Mach 2.5, α 15°

	Number of Converged Cases	RMSE	POFP
Stacked LHC 500-Case Design	458	3.15	97%
Proposed Method	**651**	**0.724**	**89%**
Stacked LHC 1,000-Case Design	922	2.99	97%
Stacked LHC 7,000-Case Design	6,430	3.28	96%
Neural Network with Nested Latin Hypercube Design	10,991	1.98	91%

after fifteen rounds of samples.

The final prediction RMSE for the proposed method using 710 cases was 0.724, which corresponded to a POFP of 89%. Using 7,000 cases, the space-filling approach produced a prediction RMSE of 3.28, which was equivalent to a POFP of 98%. Similar to the Mach 2.5, α 15° flight condition, the incorporation of multi-fidelity modeling produced a substantial improvement in predictive accuracy. Note that for space-filling sampling, the predictive accuracy stagnated quickly and did not improve as more samples were added. In contrast, surrogates based on the adaptive sampling approach using fewer than 800 samples outperformed surrogates based on nearly ten times as many space-filling samples.

For comparison, the neural network for this flight condition from the RBS project used 10,991 cases and, when applied to the present set of test cases with small pitching moment coefficients, produced an RMSE value of 1.98; its RMSE for space-filling samples was 0.456. These results are organized for easier comparison in Table 15.

Table 16: Evaluating Effects of Low-Fidelity Surrogates On Overall Accuracy

	Prediction RMSE Using Surrogates	Prediction RMSE Using APAS Directly
Mach 0.3 α 15°	0.535	0.525
Mach 0.8 α 0°	0.356	0.394
Mach 2.5 α 15°	0.277	0.245
Mach 2.5 α 40°	0.979	0.794

6.2.7.5 Evaluation of Surrogate Models as the Low-Fidelity Data Source

The very poor performance for the method at the Mach 0.8 flight condition was investigated to determine the likely cause of the error. Given the relatively poor accuracy of the surrogate model for APAS at that flight condition (the prediction error had a standard deviation of around 0.65), it seemed possible that the poor behavior stemmed from bad low-fidelity estimates. To test this possibility, multi-fidelity surrogates were trained using data directly from APAS rather than from surrogates of APAS data. Multi-fidelity surrogates were trained based on the 500-case level of the stacked Latin hypercube. One set of surrogates was trained using surrogate model predictions of APAS results as the low-fidelity data source, while the other set was trained using actual APAS results as the low-fidelity data source. Both sets of models were then tested to assess predictive accuracy. The results of these tests are presented in Table 16. When APAS was used directly, there were slight improvements at Mach 0.3, α 15° and Mach 2.5, α 15°, as well as a more substantial improvement at Mach 2.5, α 40°. Unfortunately, there was no improvement at the Mach 0.8 flight condition, which was the motivation for this test.

It would appear that a linear model, even when augmented by the low-fidelity response value, was a poor match for the behavior of the pitching moment coefficient at Mach 0.8,

Table 17: Overall Comparison of Predictive Accuracy for Longitudinal Responses

	Error Reduction Vs. Space-Filling Single-Fidelity	Error Reduction Vs. Space-Filling Multi-Fidelity	Reduction In No. of Analyses
Mach 0.3, α 15°	29%	11%	90%
Mach 0.8, α 0°	12%	4.4%	90%
Mach 2.5, α 15°	68%	4.4%	90%
Mach 2.5, α 40°	78%	20%	90%

α 0°. Based on the example of the two-dimensional Sphere function in Section 4.8, it is expected that if enough samples were available the algorithm would eventually achieve an accurate understanding of the response. However, there is no way of knowing whether that accuracy would be obtained after 10 samples or 10,000.

Note that the pitching moment coefficient at Mach 0.8, α 0° was predicted well by a multi-fidelity model with a linear underlying trend, as shown earlier in this chapter. In that demonstration, the **dacefit** utility was able to identify correlation parameters which improved the prediction accuracy of the model. When those parameters can be identified, Kriging can better estimate how the response diverges from the linear underlying trend. Although **dacefit** was unsuccessful at identifying useful correlation parameters in this enlarged problem, it was expected that once such parameters could be identified, the predictive accuracy of the Kriging model would increase substantially – at least in the neighborhood of the observed samples.

6.2.7.6 Review of Performance for Longitudinal Responses

The relative gains of the proposed method against space-filling sampling – using either single-fidelity or multi-fidelity surrogates – are presented in Table 17. For most of the

responses, multi-fidelity modeling produced the lion's share of the accuracy improvement. This was unsurprising given that the adaptive sampling algorithm depends on a reasonably accurate understanding of the behavior of each response; samples are selected based on this understanding, so if the understanding is poor, the samples may not be very helpful. Adaptive sampling still produced 5-20% improvement over the use of multi-fidelity modeling alone.

Overall, although the gains were substantial, more progress would be necessary before such surrogate models could be used for design space exploration with confidence: none of the surrogates, whether based on the null or alternative hypotheses, achieved a prediction RMSE score below 0.1, which would be equivalent to a POFP less than 33%.

For most responses, the proposed approach was moderately effective, producing significant reductions in prediction RMSE. The proposed approach was also much more efficient with respect to high-fidelity analyses: the improved predictive accuracy was obtained with a *reduction* in the number of expensive analyses by almost 90% compared to the space-filling approach (725 analyses per flight condition compared to 7,000). Despite these achievements, further improvements would still be required before surrogate models such as these could be used for engineering purposes.

6.2.8 Evaluation of Accuracy for Lateral Moments

Although the RBS project found that pitching moments were difficult to model accurately, there were also substantial difficulties in modeling the lateral responses. Because the four flight conditions evaluated for this effort only included symmetric conditions, all of the lateral responses were likely to be near zero. However, asymmetric deflections of control surfaces would still produce nonzero lateral responses that should be modeled as accurately as possible.

Recall from Section 5.6 that the APAS geometry definition did not capture control surface deflections, and thus the low-fidelity model could not capture any asymmetric effects. Any lateral responses calculated by APAS would therefore be known in advance to be spurious. Although it was shown in Section 5.6.1 that multi-fidelity models with nuggets could

Table 18: Correlation Between Uncertainty & Response Magnitude: Rolling Moments

	Mach 0.3, α 15°	Mach 0.8, α 0°	Mach 2.5, α 15°	Mach 2.5, α 40°
Correlation Between Mean μ and Standard Deviation of Uncertainty σ	-0.021	0.16	0.047	0.036

Table 19: Correlation Between Uncertainty & Response Magnitude: Yawing Moments

	Mach 0.3, α 15°	Mach 0.8, α 0°	Mach 2.5, α 15°	Mach 2.5, α 40°
Correlation Between Mean μ and Standard Deviation of Uncertainty σ	-0.097	0.23	0.12	-0.012

overcome unhelpful low-fidelity data, it was decided that APAS results would not be used in these tests. Without low-fidelity surrogate models, the only remaining source of uncertainty in the data was the iteration noise produced by Cart3D. This uncertainty was tracked and incorporated via nuggets when Kriging models were trained.

In Section 5.6.2, it was shown that capturing uncertainty via nuggets was most effective when the uncertainty in the response value was correlated with the magnitude of that response. The correlation values for the lateral responses for the 49-dimension problem were calculated to determine how effective nuggets were likely to be at reducing prediction error for those responses. The results for rolling moments are presented in Table 18, and the results for yawing moments are presented in Table 19.

For the most part, the correlation between response magnitude and uncertainty was small. The exception to this was at Mach 0.8, α 0°, where the correlation coefficient was 0.16 for the rolling moment coefficient and 0.23 for the yawing moment coefficient. These correlation values were still relatively small but indicated that some moderate degree of noise was present in the data. Incorporating this noise using nuggets could therefore improve

Table 20: Correlation Between Lateral Responses & Control Surface Deflections

	Outboard Elevon (Starboard)	Outboard Elevon (Port)	Rudder (Starboard)	Rudder (Port)
Mach 0.3, α 15° C_{Roll}	-0.86	0.78	-0.37	-0.27
Mach 0.8, α 0° C_{Roll}	-0.88	0.73	-0.38	-0.23
Mach 2.5, α 15° C_{Roll}	-0.78	0.83	-0.31	-0.36
Mach 2.5, α 40° C_{Roll}	-0.85	0.77	-0.26	-0.36
Mach 0.3, α 15° C_{Yaw}	0.51	-0.75	0.52	0.15
Mach 0.8, α 0° C_{Yaw}	0.54	-0.76	0.53	0.14
Mach 2.5, α 15° C_{Yaw}	0.50	-0.85	0.33	0.35
Mach 2.5, α 40° C_{Yaw}	0.66	-0.87	0.28	0.41

the accuracy of surrogate models for those two responses. In general, though, these results indicated that the bulk of the observed response behavior was due to actual flow phenomena and not spurious.

To determine whether the observed response behavior was due to legitimate aerodynamic effects or numerical noise, additional tests were performed to determine the correlation between rolling and yawing moments and the design variables. It was found that the lateral responses were strongly correlated with the control surface deflections, as shown in Table 20. Such deflections would in fact produce nonzero rolling and yawing moments, suggesting that the observed behavior was driven by legitimate effects rather than by noise in the data.

Two sets of models were trained: the first set embodied the null hypothesis and was made up of the first 4,000 points of the stacked Latin hypercube. These models were fit with no nuggets, treating all observed response values as perfectly deterministic. The second set embodied the alternative hypothesis and consisted of the first 500-point space-filling level of the stacked Latin hypercube plus 9 batches of adaptive samples which had been selected to improve the pitching moment surrogates. These surrogates were trained using nuggets based on the observed iterative noise in each response. Both sets of models were fit using

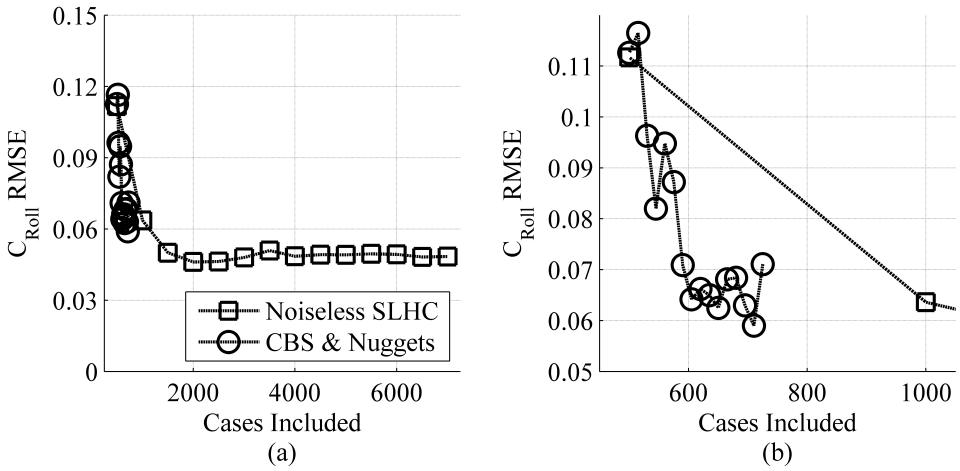

Figure 53: Predictive Accuracy for Rolling Moment at Mach 0.3, α 15°

only Cart3D results.

These models were not evaluated using POFP, the probability of false positives, because unlike pitching moment coefficient, there was no rule of thumb available to set bounds on the lateral control authority of a reusable booster vehicle.

6.2.8.1 Results for Rolling Moment

The prediction accuracy for rolling moment at Mach 0.3, α 15° is shown in Figure 53. Figure 53a shows the full range of results, indicating that the space-filling sampling produced diminishing returns after perhaps 2,000 samples. A cropped view is shown in Figure 53b to more clearly illustrate the behavior of the models based on adaptive sampling. After the most recent set of adaptive cases, the proposed approach with nuggets produced a prediction RMSE of 0.0711 using 725 cases. The space-filling cases produced prediction RMSE values of 0.112 and 0.636 based on 500 and 1,000 cases, respectively.

These results indicated that for this response, incorporating noise did not substantially reduce prediction error. This agreed with the observation that iteration noise had minimal correlation with the response magnitude. For this response, the proposed method was approximately equivalent to the baseline approach.

The results for rolling moment coefficient prediction accuracy at Mach 0.8, α 0° appear in Figures 54a & 54b. At this flight condition, the space-filling approach started out with a

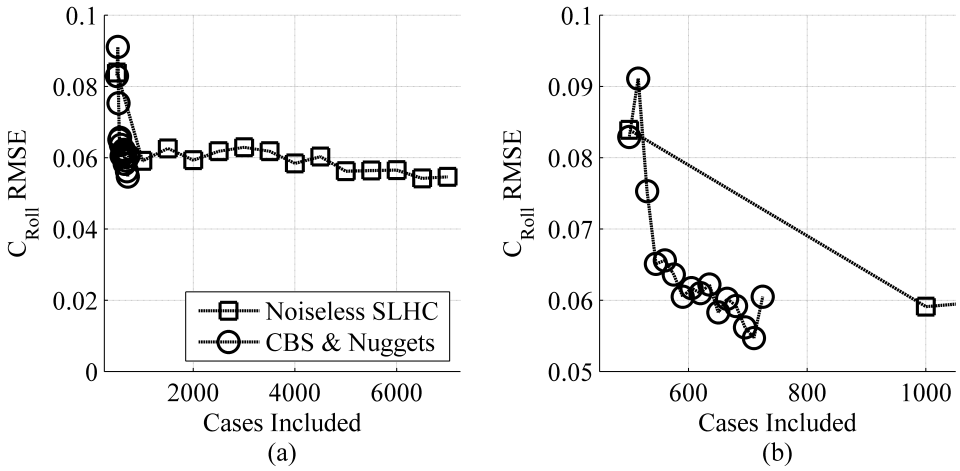

Figure 54: Predictive Accuracy for Rolling Moment at Mach 0.8, α 0°

prediction RMSE of 0.0839 based on 500 cases but improved to 0.0591 for 1,000 space-filling cases. For larger numbers of cases, the average prediction RMSE was 0.0588.

The surrogate models based on the proposed approach demonstrated rapid initial improvement, similar to the noiseless surrogates, and *appeared* to show more rapid improvement than the baseline as the training data pool grew larger. After fifteen rounds of adaptive sampling (725 samples total), the prediction RMSE was 0.0605. The prediction RMSE after *fourteen* rounds of sampling (710 cases) was 0.0547, better than any noiseless model that was trained with fewer than 6,500 cases.

In general, there was mild evidence that the proposed method might have produced more-accurate surrogates if sampling had continued; based only on the available evidence the two approaches are effectively neck-and-neck. These results were consistent with the calculated correlation between the iteration noise for this response and the response magnitude (0.16), which would suggest mild improvements at best.

Figure 55 shows the results for Mach 2.5, α 15°. As was observed for Mach 0.8, the space-filling approach did not obviously improve for sampling sizes above 1,000 cases. The average prediction RMSE for those models was 0.0177, while the RMSE for the model based on 500 cases was 0.0279.

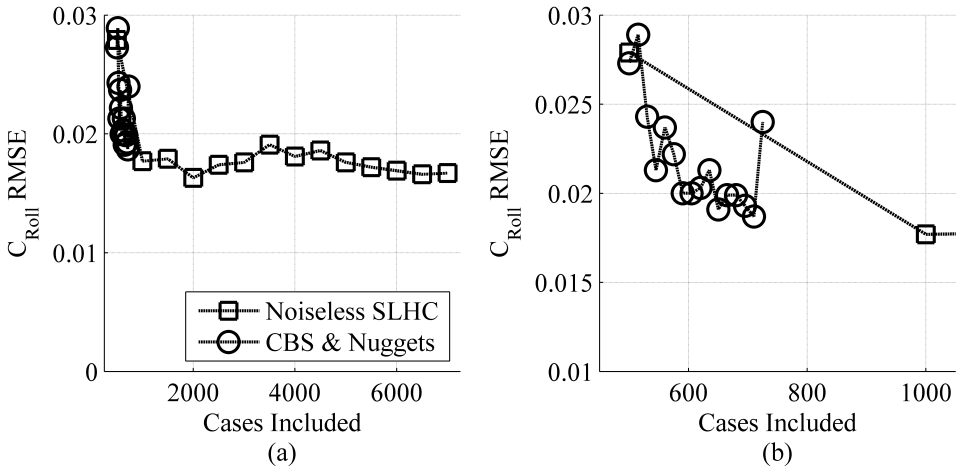

Figure 55: Predictive Accuracy for Rolling Moment at Mach 2.5, α 15°

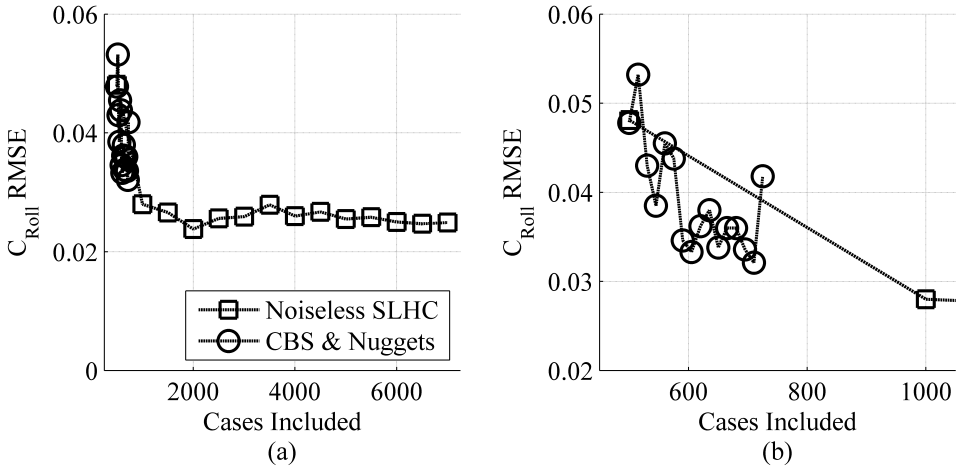

Figure 56: Predictive Accuracy for Rolling Moment at Mach 2.5, α 40°

Once again, the proposed approach with nuggets appeared to improve prediction accuracy slightly more efficiently than the space-filling approach, but the surrogate models based on the proposed approach were never clearly superior. The prediction RMSE values after the fourteenth and fifteenth rounds of adaptive sampling were 0.0187 and 0.0240, respectively. The correlation coefficient between iteration noise and response magnitude for this response was 0.047, which suggested that capturing iteration noise would not produce much improvement - a deduction consistent with the numerical results.

Lastly, the prediction performance for rolling moment coefficient at Mach 2.5, α 40°

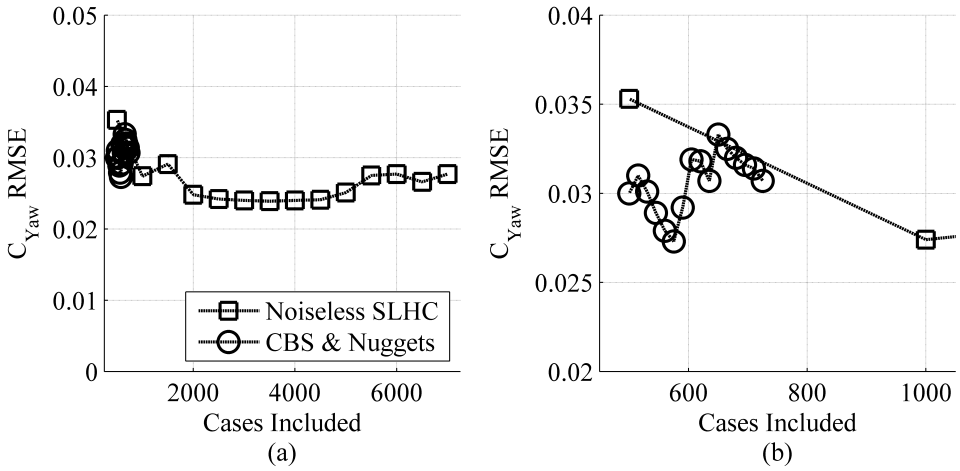

Figure 57: Predictive Accuracy for Yawing Moment at Mach 0.3, α 15°

may be seen in Figure 56a & 56b. As with the other flight conditions, models based on the proposed approach appeared to reduce RMSE using fewer cases than the space-filling approach, but fifteen batches were not sufficient to demonstrate this one way or the other. The average RMSE for models based on at least 1,000 space-filling cases was 0.0256. The RMSE values after the fourteenth and fifteenth batches of adaptive samples were 0.0321 and 0.0418, respectively.

With respect to rolling moment coefficient, the proposed approach did not produce surrogate models that were significantly more accurate than the baseline approach. This was not entirely surprising, as the correlation calculations in Table 18 indicated that iteration noise was not strongly affecting those responses. The strongest correlation between iteration noise and a lateral response, 0.23, was for the yawing moment coefficient at Mach 0.8, as noted in Table 19. The following section will quantify the effects of nuggets when predicting yawing moment coefficients.

6.2.8.2 Results for Yawing Moment

Figure 57 illustrates how prediction RMSE for the yawing moment at Mach 0.3, α 15° varied with sampling and modeling approach. The space-filling approach seemed to converge to an RMSE value of around 0.024 after roughly 1,500 cases. The models using nuggets did demonstrate slightly improved performance – RMSE for the 500-case set when nuggets

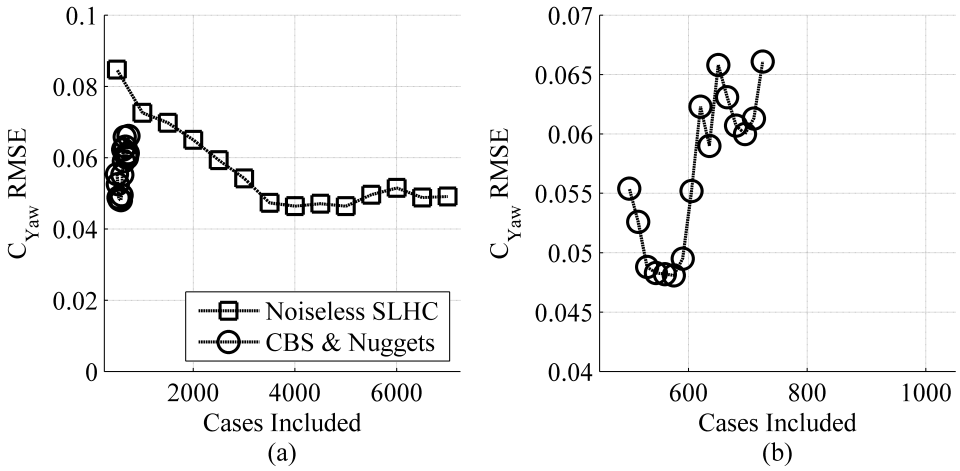

Figure 58: Predictive Accuracy for Yawing Moment at Mach 0.8, α 0°

are used was 0.0300, versus 0.0353 without nuggets – but the results did not demonstrate *consistent* improvement as more samples were added. After the fifteen sets of samples, the prediction RMSE for models with nuggets was 0.0307, slightly worse than the surrogate trained with 500 samples.

The results for yawing moment at Mach 0.8, α 0° are shown in Figure 58. For noiseless models trained with the space-filling samples, there was a clear improvement as more samples are used: the model based on the set of 500 samples had a prediction RMSE of 0.0847, while the one based on 2,000 samples had a prediction RMSE of 0.065 and that based on 4,000 samples produced a value of 0.0464. This improvement leveled off when more than 4,000 samples are available.

For this response, there was a clear benefit to the use of nuggets. Incorporating uncertainty due to iteration for the 500-case sample set reduced prediction RMSE from 0.0847 to 0.0554. Models based on later data sets produced prediction RMSE values between 0.0481 and 0.0661. Even the least-accurate surrogate that used nuggets was more accurate than every noiseless surrogate trained on fewer than 2,000 samples.

The use of nuggets allowed a surrogate trained with 725 samples to out-perform a noiseless surrogate trained with more than twice as much data. The correlation between response magnitude and iteration noise was 0.23, which had led to the expectation that surrogates

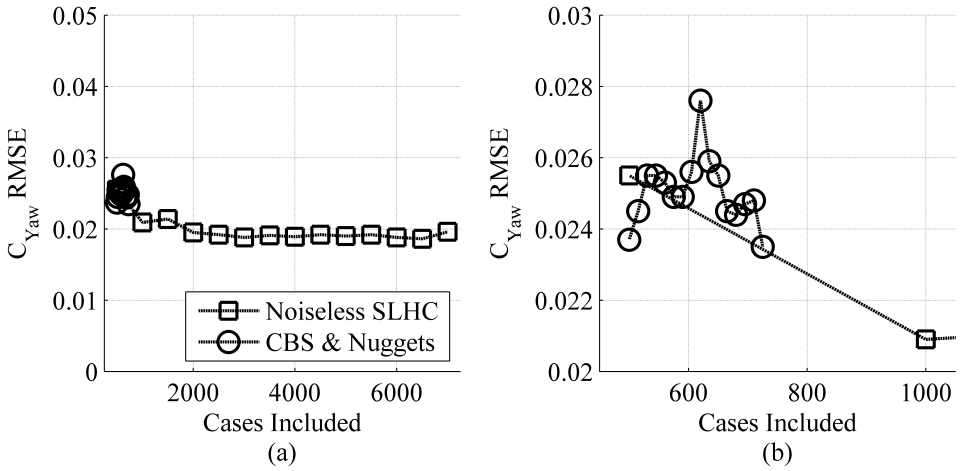

Figure 59: Predictive Accuracy for Yawing Moment at Mach 2.5, α 15°

using nuggets would be more accurate for this response. These observations supported that conclusion.

All surrogate models for yawing moment coefficient at Mach 2.5, α 15° had very similar performance. Surrogates without nuggets based on space-filling samples demonstrated slight improvement as more cases were added, starting at a prediction RMSE of 0.0255 and averaging an RMSE of 0.0191 when more than 1,500 samples were incorporated. Surrogates which incorporated nuggets did not perform any better: the prediction RMSE values ranged from 0.0235 to 0.0276. The correlation coefficient between response magnitude and iteration noise for this response was 0.12, indicating that iteration noise was unlikely to be a major factor in prediction error.

Finally, the results for yawing moment predictions at Mach 2.5, α 40° are depicted in Figure 60. These results were very similar to those for Mach 2.5, α 15°. Noiseless surrogates based on space-filling samples showed mild improvement as more samples were added, eventually settling down to an average RMSE of 0.0205 when more than 1,500 cases were available.

The proposed method produced a slight improvement for the initial sample set of 500 cases, reducing prediction RMSE from 0.0226 to 0.0209, but subsequent sets of data produced worse prediction RMSE values than were produced by the noiseless models. The

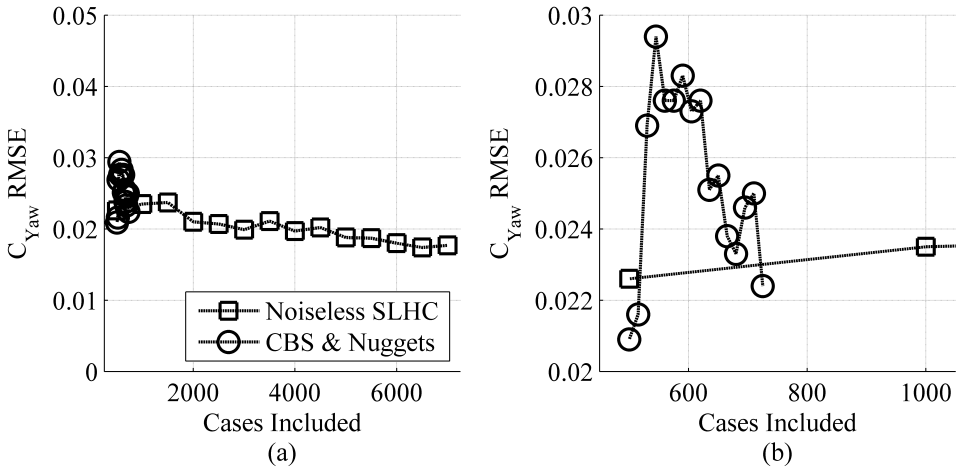

Figure 60: Predictive Accuracy for Yawing Moment at Mach 2.5, α 40°

correlation coefficient between response magnitude and iteration noise was 0.012, which had indicated that nuggets would not substantially improve prediction error for this response.

In general, the proposed method – and in particular the use of nuggets to capture data uncertainty – demonstrated only minor improvements at best compared to noiseless models for most flight conditions. However, at Mach 0.8 a significant benefit was observed for the yawing moment coefficient, and noiseless surrogates required 4-5 times as many samples to equal the performance of the surrogates which captured uncertainty via nuggets. It was found that the impact of nuggets closely matched the degree of correlation between the response magnitude and the iteration noise, which indicates that the user may only wish to use nuggets when that correlation is relatively large (>0.2). The causes for the observed behaviors will be discussed in the next section.

6.2.9 Interpretation of Longitudinal & Lateral Results

With regard to pitching moment, the proposed method was quite effective for most flight conditions. After 15 batches of adaptive samples, the prediction RMSE at Mach 0.3 was 30% smaller than the single-fidelity surrogate trained with 7,000 space-filling samples. At Mach 0.8, the surrogate based on 15 batches of adaptive samples was 12% more accurate than the baseline approach. For the two Mach 2.5 flight conditions (α 15 & 40°), RMSE was reduced by 67% and 78% respectively. Note that these improvements in accuracy were

achieved while *reducing* the number of training samples by nearly 90% compared to the single-fidelity, space-filling approach.

The pitching moment coefficient at Mach 0.8, α 0° proved to be the most difficult to predict accurately. At this flight condition, the use of multiple sources of data provided marginal improvements at best, and it appeared that the response behavior could not be easily modeled with a linear underlying trend. As more training data was used, the model's performance became progressively worse. It was expected that this trend would reverse itself eventually, as was observed with the two-dimensional Sphere Function example in Section 4.8, but it was not known *when* that reversal might occur.

The proposed method demonstrated no significant improvements when used to predict rolling moment coefficients. When applied to yawing moment coefficients, the proposed method produced clear improvements for Mach 0.8 but was approximately equivalent with the baseline approach at other flight conditions. The proposed method primarily distinguished itself for lateral responses through its use of nuggets. The other specialized aspects of the proposed method – use of lower-fidelity data and an iterative sampling strategy – were not relevant because the cheaper APAS data did not capture the phenomena which drove the lateral responses observed in Cart3D, and the iterative strategy focused purely on the pitching moment coefficients. This left the use of nuggets as the distinguishing feature between the standard method and the proposed method.

Section 5.6.1 showed that the use of nuggets produced significant improvements in fit accuracy when the response was correlated with the amount of noise in that response – in essence, when any large values that were observed were more likely to be spurious than representative of actual response behavior. Observations from this demonstration mirrored those results. The lateral response with the strongest correlation between the response magnitude and iteration noise was the yawing moment coefficient at Mach 0.8, α 0°, and it was this response that showed the largest improvement in predictive accuracy when the proposed method was applied.

6.2.10 Shortcomings of Full-Scale Test

The biggest shortcoming of the full-scale test was the inability of the Kriging surrogate models – *all* the Kriging surrogate models – to identify useful correlation coefficients. Lacking such coefficients, the Kriging surrogates would only deviate from the underlying linear trend models in the very close vicinity to training points. Given that the response behaviors were unlikely to be linear over all 49 input parameters, this led to relatively surrogates.

In Section 3.3, it was noted that a sparse correlation matrix would lead to a heavy dependence on the underlying trend model. That generalization was made in the context of applying sparse methods to the problem in the event that the correlation matrix became so large and ungainly that it approached the memory limits of Matlab, the program being used to create the Kriging surrogates. Although no sparse methods were applied to the problem at hand, the same effect – surrogate models which depended strongly on the underlying trend, with correlation parameters affecting the predictions only rarely – was observed in the results. This indicated that sparseness effects might have played some role in the difficulties that were experienced in the full-scale test. To test this possibility, another study was designed to determine whether the difficulties were related to sample sparseness.

6.3 Investigation of Sparsity Effects

Investigating this possibility for the full-scale problem was daunting: the most direct approach would have been to increase the sample density, and by extension increase the number of samples. This was a difficult proposition, since training sets of up to 7,000 cases had difficulty identifying useful correlation coefficients; training sets with *more* than 7,000 cases produced "Out of Memory" errors from Matlab. Instead, based on the recommendations of the research committee, the smaller 9-dimensional problem was used. Instead of progressively increasing the sample density for the larger problem, the sample density would be progressively *decreased* for the smaller problem. As surrogate models were trained with smaller and smaller data sets, it was expected that they would eventually behave in the same manner as the surrogates for the large-scale test.

A stacked Latin hypercube was generated using eighty 25-case Latin hypercubes for a

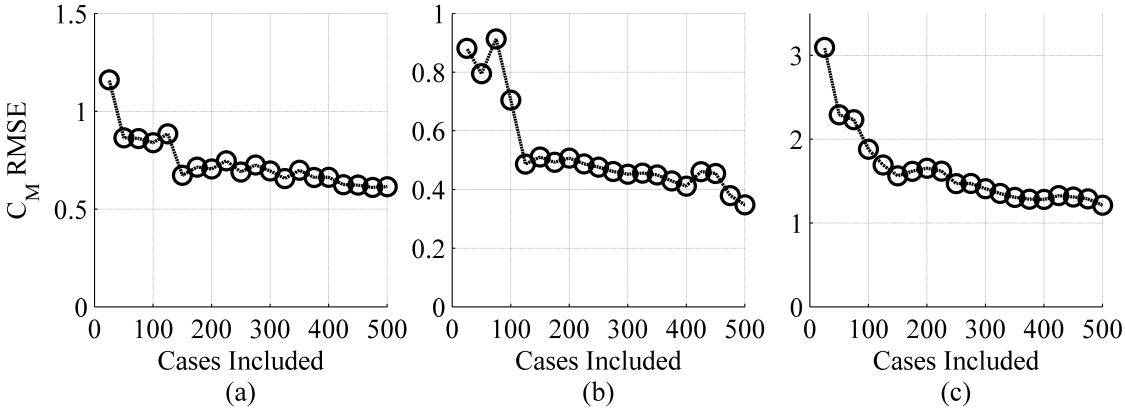

Figure 61: Change in Predictive Accuracy as Training Set Grows

total of 2,000 cases. When every dimension was normalized to a zero-to-one range, the maximin distance between two points was 0.272. For comparison, fifty 2,000-case standard Latin hypercubes were generated; the best-spaced hypercube had a maximin distance of 0.233, indicating that the stacked Latin hypercube cases were well spread out throughout the design space.

Surface meshes for the cases in the stacked Latin hypercube were generated with the PaceLab geometry tool and analyzed with Cart3D at the three relevant flight conditions: Mach 0.3, α 15°; Mach 0.8, α 0°; and Mach 2.5, α 0°. Single-fidelity surrogate models were trained to emulate the pitching moment coefficient at every flight condition using progressively larger levels of the stacked Latin hypercube (SLHC), starting with the smallest set of 25 space-filling points. All Kriging models were trained using an anisotropic Gaussian correlation function and a linear underlying trend. This was different than the previous surrogate models made of these responses (in Section 6.1.1), which used quadratic underlying trends. The change was made because a lower-order trend would require fewer samples – a 9-dimensional quadratic trend would require 91 samples to fit a model, while a linear trend would only require 10 – and the objective was to investigate effects related to sparsity.

The predictive accuracy of the resulting surrogate models was then evaluated using the test cases from Section 4.10.6. The results are plotted in Figure 61. Figure 61a shows the results when predicting C_M at Mach 0.3, α 15°; Figure 61b shows the results for Mach

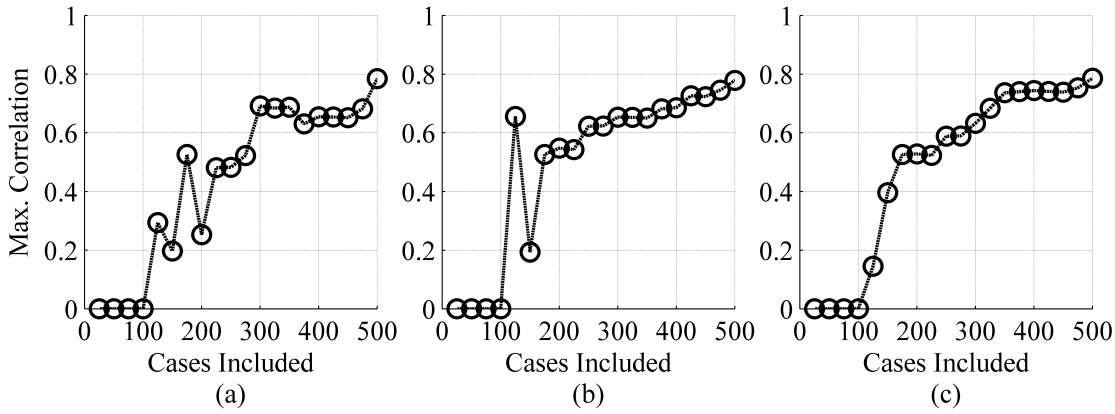

Figure 62: Maximum Correlation Between Any Training Case & Any Test Case

0.8, α 0°; and Figure 61c shows the results for Mach 2.5, α 0°. Note that for all three responses, the predictive accuracy improves rapidly until 100-125 cases are included, and then the improvement is more sedate. This change in behavior indicates that a different phenomenon is at play for the smaller training sets.

To further investigate this behavior, the correlation between training points and test points was calculated for each training set. The correlation was calculated with the same anisotropic Gaussian correlation function used in the Kriging models, while the correlation coefficients were estimated when the Kriging surrogate model was trained. The maximum correlation between any one training case and any one test case is plotted in Figure 62. Figure 62a shows this maximum correlation at the Mach 0.3 flight condition, Figure 62b shows the maximum correlation at the Mach 0.8 flight condition, and Figure 62c shows the maximum correlation at the Mach 2.5 flight condition.

The largest correlations between training and test points for the surrogates trained on the four smallest data sets were less than 1^{-5} for all responses. This is similar to the behavior observed in the large-scale test problem where there was minimal correlation between training and test data, resulting in a total dependence on the underlying trend when predicting responses for the test data points. When the correlation coefficients for these Kriging models were investigated, the surrogates trained on the smallest data sets were unable to identify any coefficient values that would improve surrogate model accuracy, which

was also observed in the large-scale test problem. These lines of evidence indicated that the phenomenon which caused such trouble in the large-scale test problem was present in this investigation, affecting the surrogate models trained on the smallest data sets.

The DACE toolbox was first able to identify useful correlation coefficients for the 125-case level of the SLHC. For this data set, 125 cases were analyzed at each flight condition. After stripping out cases which did not converge at all flight conditions, 74 cases were left in the data set. This would have been too few to fit a quadratic underlying trend, supporting the decision to use a linear trend for this investigation.

It was hypothesized that the DACE toolbox was having difficulty identifying useful correlation coefficients because the training data points were too far apart. If this were the case, the model-fitting utility would not find any useful difference between different coefficient values because no two training points would be close enough to each other to have a non-negligible correlation over the ranges being evaluated. To test this hypothesis, a number of new data points were analyzed at various distances from one of the training points in the smallest, 25-case level of the SLHC. This "seed point" was selected from the available converged data set.

6.3.1 Generating Nearby Samples

New data points were generated using a user-specified distance-limiting parameter and a set of random numbers to perturb the new point away from the initial point. Each new point was perturbed in every dimension from the initial point. To perturb the new point, the user would set a value for the distance-limiting parameter, between 0 and 1, which indicated how far the new point was allowed to move from the initial point in each dimension. Two random numbers on the range 0–1 were then drawn for each of the 9 dimensions. If the first random number in each pair was greater than 0.5, the new point would have a higher value in that dimension than the initial point; otherwise, it would have a lower value. The second random number was used to determine the magnitude of the perturbation in that dimension.

$$posneg = \begin{cases} -1 & \text{if } R_1 < 0.5 \\ 1 & \text{otherwise} \end{cases}$$

$$\delta_i = posneg \times R_2 \times Range_i \times Limit \tag{42}$$

Here, δ_i is the perturbation that is applied in the i^{th} dimension; *posneg* is a parameter that determines whether the perturbation will be in the positive or negative direction; R_1 and R_2 are the first and second random numbers that were generated between 0–1; $Range_i$ is the size of the design space in the i^{th} dimension (i.e., if the i^{th} parameter ranged from 5 to 25, $Range_i$ would be 20); and *Limit* is the user-specified distance limit. A small value for *Limit* would produce new samples that were very close to the initial point, while a larger *Limit* value would allow the new point to move farther away. This perturbation process was repeated for every dimension to make each new point. If the new point lay beyond the edge of the design space in any dimension, it was moved to the edge of the design space in that dimension.

6.3.2 Evaluating the Use of Nearby Samples

Twelve groups, each with 20 new points, were generated and analyzed with Cart3D. The results were combined with the results for the first level of the SLHC, and new Kriging surrogates were trained. Each of the new samples was added to the training set separately, and the combination of the original training set and this new sample were used to train a new surrogate model. These surrogates were then inspected to determine whether the DACE toolbox had been able to identify useful correlation coefficients.

It was observed that, when *Limit* values were small, adding even a *single* nearby sample to the training set led to Kriging surrogates with non-trivial correlation coefficients. For larger values of *Limit*, it became less likely that one new sample would be so useful. For each *Limit* value, the fraction of samples that led to useful correlation coefficients was calculated; the results are shown in Figure 63.

For *Limit* values larger than 0.2, it became progressively less likely that the new sample will lead to useful correlation coefficients, indicating that the closeness of the training data

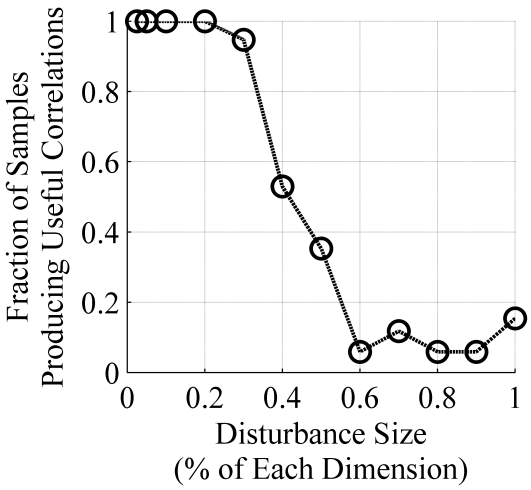

Figure 63: Fraction of Samples That Led To Useful Correlation Coefficients

– or at least, the closeness of two or more points within the training data – was of great importance when the DACE toolbox is attempting to fit a Kriging model. It should also be noted that, by deliberately placing a single new sample close to an existing sample, useful correlation coefficients could be identified using only the 25-case level of the SLHC. In contrast, when space-filling samples were added instead (i.e., using larger levels of the SLHC), no useful coefficients were identified until the 125-case level.

This suggested that, although spreading samples out through the design space is helpful for developing an understanding of the overall response behavior, *some* amount of clustering is desirable when training Kriging models. If the samples are spread out far from each other – as was the case in these experiments, where an optimizer was used to maximize the distance between any two points – the DACE toolbox may not be able to fit a useful model to the data. It would appear that stacked Latin hypercubes are therefore inappropriate for use with Kriging models, particularly for large design spaces or when only a limited number of samples will be used.

6.4 Review & Summary

In general the proposed method was successful: through use of data fusion and adaptive sampling, surrogate models for pitching moments were more accurate even when trained

with fewer cases, a result that was very clear in the 9-dimensional test problem. For lateral responses, the use of nuggets improved the accuracy of surrogates when iteration noise was an important factor.

Some shortcomings were observed, however. Neither the proposed method nor the baseline approach performed well when applied to the full-scale test problem. The surrogate models for pitching moment coefficient for C_M at Mach 0.8 grew increasingly and rapidly inaccurate as more samples were added to the training data, indicating that the effectiveness of the proposed approach is dependent on the difficulty of modeling the response. When the estimated behavior of the response is inaccurate, the samples that are selected may not improve the accuracy of the model. Evidence indicated that the sheer amount of distance between samples was a major reason why surrogate models failed to identify useful correlation coefficients; for future efforts using Kriging, less effort should be spent maximizing the space-filling characteristics of the sampling plan. Instead, some mild clustering was shown to be beneficial in cases of sparse data sets, as it led to improved estimation of correlation coefficients. If the initial set of samples is too sparse, the adaptive sampling algorithm can be directed to cluster samples by setting the POI requirement very high (as in Section 4.9), although the effectiveness of this strategy will depend on how close the candidate samples are to existing training samples.

Additionally, nuggets were introduced as a response to the relatively large iteration noise that was observed in the data during the RBS project. For responses where the response magnitude was strongly correlated with the standard deviation of the iteration noise, the use of nuggets was shown to improve predictive accuracy. However, when the uncertainty was *not* well-correlated with the amount of uncertainty in the data, no significant improvement was observed.

The resulting surrogate models, although often improved with respect to the baseline approach, were not sufficiently accurate for engineering purposes. However, sufficient evidence was gathered to assess the effectiveness of the proposed approach. The effectiveness of this approach was found to increase under certain conditions:

- When the responses being modeled could be approximated effectively with simple

surrogates despite a limited quantity of data;

- When the cheaper data source provided useful insight into the behavior of the response as calculated by the high-fidelity, expensive data source; and
- When uncertainty was correlated with response magnitude.

For problems which did not meet one or more of these conditions, the effectiveness of the proposed approach was reduced or negated. In addition, when samples were too spread out from one another, sparsity effects could prevent the identification of useful correlation parameters for Kriging models. This limited the ability of the Kriging surrogate models to fit the responses well, which in turn handicapped the adaptive sampling algorithm and the overall performance of the approach. These sparsity effects could be mitigated by mild clustering, which could be achieved in the initial set of samples or by setting high POI requirements for some of the adaptive samples. Investigations into the degree of clustering required for good performance was left for future work.

CHAPTER VII

SUMMARY, CONTRIBUTIONS & CONCLUSIONS

This dissertation attempted to identify, evaluate and demonstrate a new sampling and surrogate modeling procedure to help users to create accurate surrogate models of expensive, high-fidelity analysis tools at a more reasonable cost than was previously possible. Such surrogate models would allow users to have greater confidence in the decisions made during a design effort, reducing the risk that inadequate modeling fidelity would lead to dead ends and backtracking. Previous efforts demonstrated that, although valuable, such surrogate models could be exceedingly expensive to train to a useful level of accuracy.

The proposed approach therefore emphasized efficiency, using an adaptive sampling algorithm to identify the experiments that would best improve the surrogates, in order to minimize the number of analyses required while maximizing the improvement produced by each analysis. In addition, cheaper sources of data were leveraged when possible, and uncertainty in data values was quantified to reduce the likelihood of over-fitting a noisy response.

At the end of Chapter 3, this research effort was framed in the context of a sampling and modeling methodology. The methodology included multiple steps, from the initial exploratory analyses to the evaluation of surrogate models. In particular, the research questions and hypotheses that drove this research were formulated to identify the most effective ways to carry out each step of the methodology. Note that the methodology was deliberately designed to be agnostic with respect to the analysis tools used. The following section will review the research questions and the hypotheses that were formulated to address these research questions. After that review, it will show how the experimental results addressed the hypotheses and the research questions in turn.

7.1 Review of Research Questions & Hypotheses

The primary research question that drove this effort was:

How can high-fidelity modeling be feasibly applied earlier in the design process, despite the computational expense?

Based on observations made from the first attempts to create aerodynamic surrogate models of reusable boosters (Chapter 2), three factors (adaptive sampling, multi-fidelity modeling, and capturing uncertainty) were identified that were expected to be a more effective way to approach the problem. These factors led to focused research questions, which scoped the problem and drove the literature search (Chapter 3). The literature search, in turn, identified methods which could address those focused research questions; these expectations were expressed in the form of hypotheses, and experiments were designed to test those hypotheses.

7.1.1 First Focused Research Question & Hypothesis

The first observation was that the pitching moment coefficient for many configurations was so extreme for at least one flight conditions that they were unlikely to be feasible designs. If samples could be placed to emphasize feasible configurations, surrogate models could be created that were accurate for the feasible regions of the design space while minimizing the number of infeasible configurations analyzed. The fact that the objective was a particular range of each response, rather than a maximum or minimum, meant that common sample selection approaches were not appropriate. This led to the first focused research question:

When "good performance" refers to responses within desirable ranges rather than maxima or minima, how can regions of good performance be identified and emphasized during the sampling process?

A review of available sample selection methods led to the identification of contour-based sampling as a promising approach. This formed the first contributing hypothesis, Hypothesis 1:

Contour-based sampling will balance the selection of cases with good performance and the reduction of prediction uncertainty in promising regions, identifying samples that efficiently improve surrogate accuracy for configurations with small aerodynamic moments.

Chapter 4 detailed the implementation of the method and the experiments that tested the hypothesis. A number of test problems were used, each of which had one or more responses for which only a particular range of response values were of interest. Surrogate models were then trained with data sets generated with both the proposed approach (using contour-based sampling to augment a set of initial space-filling samples) and with the baseline approach (using only space-filling samples).

In most cases, contour-based sampling led to surrogate models that were more accurate than those based on space-filling samples. The exception to this rule was when the underlying trend of the Kriging surrogate model was a poor representation of the response behavior, such as the Sphere Function in Section 4.8. In that case, contour-based sampling had relatively poor performance due to a mismatch between the *perceived* response behavior (in the form of the surrogate model used by the sample-selection algorithm) and the *actual* response behavior. In general, however, the tests demonstrated that contour-based sampling did lead to surrogate models that were more accurate, even when based on smaller training sets.

Section 4.10 demonstrated contour-based sampling for multiple flight conditions. Surrogate model accuracy was evaluated using test cases that had small aerodynamic moments at each flight condition. Creating a training data set with contour-based sampling led to surrogate models that were more accurate than if only space-filling samples were used, even if a larger number of space-filling samples were available. Thus, contour-based sampling was shown to "efficiently improve surrogate accuracyfor configurations with small aerodynamic moments," supporting this hypothesis.

Contour-based sampling effectively identified regions of good performance (i.e., regions with response values within a specified range) and placed samples in those regions. This behavior produced surrogate models that were more accurate over that specified response range. In fact, contour-based sampling was so effective that in most cases, surrogate model prediction accuracy could be improved while *reducing* the number of cases used to train the surrogates. In light of those results, the research question was considered to have been addressed satisfactorily.

7.1.2 Second Focused Research Question & Hypothesis

The second observation was that, although simpler analysis methods such as APAS were not sufficiently adequate to be the sole source of data, such methods could still shed light on the overall trends in response behavior. If so, this could substantially reduce the number of expensive analyses necessary to train accurate surrogate models. This led to the second focused research question:

> ***How can cheaper analyses be integrated with high-fidelity models to reduce the overall cost of design space exploration or exploitation?***

The process of combining information from multiple data sources is known as data fusion or, in situations where some data sources are more accurate than others, multi-fidelity modeling. Many alternative methods for data fusion exist; it can be difficult to know which one is best-suited for the problem at hand. The second focused hypothesis, therefore, did not identify one data fusion method in particular:

> ***Data fusion techniques will allow results from high-fidelity analyses to be augmented with cheaper sources of data to produce surrogate models that are more accurate yet require less computationally-expensive data.***

This hypothesis was tested in Chapter 5. Four data fusion methods – additive correction, proportional correction, Ghoreyshi cokriging, and data harmonization – were implemented and applied to various test problems that were similar to the intended application of reusable booster aerodynamics. Based on those tests, one of those techniques (Ghoreyshi cokriging) was selected since it produced the most accurate surrogate models for the test problems.

Here, the experiments served two purposes: they compared data fusion techniques against each other to identify the most effective approach, and they compared those techniques against the standard single-fidelity approach to confirm that data fusion would produce better surrogates. To be precise, the hypothesis was tested by the latter comparison; the evaluation of Ghoreyshi cokriging against other data fusion techniques served simply to determine which technique was the most promising.

Surrogates created using Ghoreyshi cokriging were more accurate than the standard single-fidelity surrogates for all test problems. Those results supported the hypothesis

that data fusion would lead to surrogates that were "more accurate" yet required "less computationally-expensive data." In particular, surrogates created with Ghoreyshi cokriging were more accurate than single-fidelity surrogates that had been trained with much more data from the expensive data source. These results were strong enough that the second focused hypothesis was considered to be supported. The results also demonstrated that data from multiple sources were being blended together to produce surrogate models that were more accurate while requiring fewer expensive analyses, allowing design space exploration and exploitation to be conducted at reduced cost. Thus, the second focused research question had been addressed satisfactorily.

7.1.3 Third Focused Research Question & Hypothesis

The third observation was that predictive accuracy for lateral responses was very poor, and this poor accuracy was due in part to the relatively noisy behavior of the responses. By ignoring that noise and treating the responses as deterministic, the surrogate models were actually *less* accurate than if the surrogate were replaced with the response mean. Identifying which data points were accurate and which were spurious might lead to a more accurate surrogate. However, most surrogate modeling techniques could not take such information into account, leading to third focused research question:

> ***How can information about uncertainty in the data be captured effectively?***

Although most engineering applications of Kriging assume that the training data is deterministic, an alternative formulation using nuggets was identified that could account for uncertainty in the data. Critically, this approach allowed the user to specify a different uncertainty magnitude for each data point. This led to the third hypothesis:

> ***When creating a Kriging model, the use of nuggets will capture uncertainty in the data, improving predictive accuracy for noisy responses.***

This hypothesis was also tested in Chapter 5, particularly Section 5.6. The experiment demonstrated that predictive accuracy for yawing moment coefficient could be substantially improved when sources of uncertainty (especially iteration noise) were captured via nuggets

when training the Kriging model. This effect was significant for noisy data but was negligible when there was not much uncertainty present in the data.

The results showed that the use of nuggets did in fact improve predictive accuracy for noisy responses, giving support to the hypothesis. This indicated that nuggets serve to answer the focused research question, allowing the user to capture uncertainty in the response in an effective manner, such that the resulting surrogate models would be more accurate than the deterministic surrogates that are common in engineering.

7.1.4 Primary Research Question & Final Hypothesis

Each of the three research questions were formulated to address a factor that made it difficult to make surrogate models of expensive high-fidelity data sources, and thus each question addressed an aspect of the primary research question:

> ***How can high-fidelity modeling be feasibly applied earlier in the design process, despite the computational expense?***

Each supporting hypothesis highlighted a technique to address this primary research question: contour-based sampling optimized the selection of samples; data fusion incorporated cheaper data sources; and nuggets captured noise or uncertainty in the observed data. It was asserted that, by combining these techniques into a coherent approach, the primary research question might be answered. This assertion was itself the final hypothesis of the research effort:

> ***By placing samples intelligently, reducing dependence on the expensive models, and accounting for any uncertainty in the data, the selected methods will enable improved surrogate model accuracy with significantly reduced data requirements, such that high-fidelity modeling becomes a feasible option earlier in the design process.***

This hypothesis was tested by applying it to a series of representative problems of increasing complexity. These tests were described in depth in Chapter 6. In the first test, the combined techniques were applied to a problem with 3 responses (all of which were used for adaptive sampling) and 9 free parameters. This test focused on predicting pitching

moment coefficient, which exhibited relatively low uncertainty; as a result, only data fusion and adaptive sampling were used. The combined techniques were shown to be quite effective at improving predictive accuracy for this problem, supporting the hypothesis.

As a final test, the combined techniques were applied to a problem with 12 responses (4 of which were used for adaptive sampling) and 49 free parameters. This test would quantify how the combined techniques performed when applied to the motivating problem of reusable booster design. The test showed that all Kriging models – whether produced by the baseline approach or by the proposed approach – had difficulty fitting the responses. Evidence indicated that this was due to the large distances between the samples. When surrogates were fit to the lateral responses, the proposed approach out-performed the baseline approach for responses where noise was significant, but had roughly equivalent performance where noise did not significantly affect the response value. With regard to the longitudinal responses, data fusion provided the bulk of the observed improvements, which was unsurprising given that the effectiveness of contour-based sampling was shown to depend on the accuracy of the available surrogate models when evaluating new samples.

The large-scale problem gave weak evidence to support the final hypothesis: although the combined techniques out-performed the baseline approach, countour-based sampling was not particularly effective since it was difficult for the algorithm to assess candidates with any accuracy. However, the smaller-scale problem did demonstrate that when the Kriging models were moderately accurate, the combined techniques could produce significant improvements in predictive accuracy. The combined techniques were shown to improve predictive accuracy while reducing dependence on the expensive data source, supporting the final hypothesis and in turn addressing the primary research question: the selected techniques reduced the computational expense of high-fidelity modeling by a substantial margin, enabling design space exploration or optimization at a more reasonable cost than was possible before.

The specific steps of the proposed method will be reviewed in the next section.

7.2 Review of Steps in the Method

Section 3.6.1 described the generic approach to creating surrogate models. This section will review those steps in light of the experimental results that have been observed, clearly identifying the way that certain steps have been updated in light of observations made during this research. The updated steps are illustrated in flowchart form in Figure 64.

Step 1: Generate an initial set of samples to be analyzed.

The use of data fusion will affect the way that the initial samples should be selected, as well as the way that the surrogate models are trained. Sample distributions such as nested Latin hypercubes,[152] sliced Latin hypercubes,[153] and stacked Latin hypercubes (Section 6.2.4.2) allow the user to generate sample distributions that can fill multiple roles simultaneously: the overall set of samples is large and space-filling, suitable for the data source that has low per-analysis costs, while subsets are identified that are much smaller yet still retain good space-filling characteristics, suitable for the higher-fidelity data source with its greater per-analysis costs.

Step 2: Analyze the samples using the appropriate data sources.

Step 3: Train Kriging surrogate models using the resulting data. *Nuggets* can be used to capture the relevant uncertainties in the data, while *Ghoreyshi cokriging* will produce surrogates which are more accurate but require less investment in training data.

It should be noted that the effectiveness of a data fusion technique will depend on the problem being addressed. For the applications described in this work, Ghoreyshi cokriging produced the largest improvement in predictive accuracy, but this may not be the case for every application.

Step 4: Evaluate the resulting surrogate models to quantify the predictive accuracy.

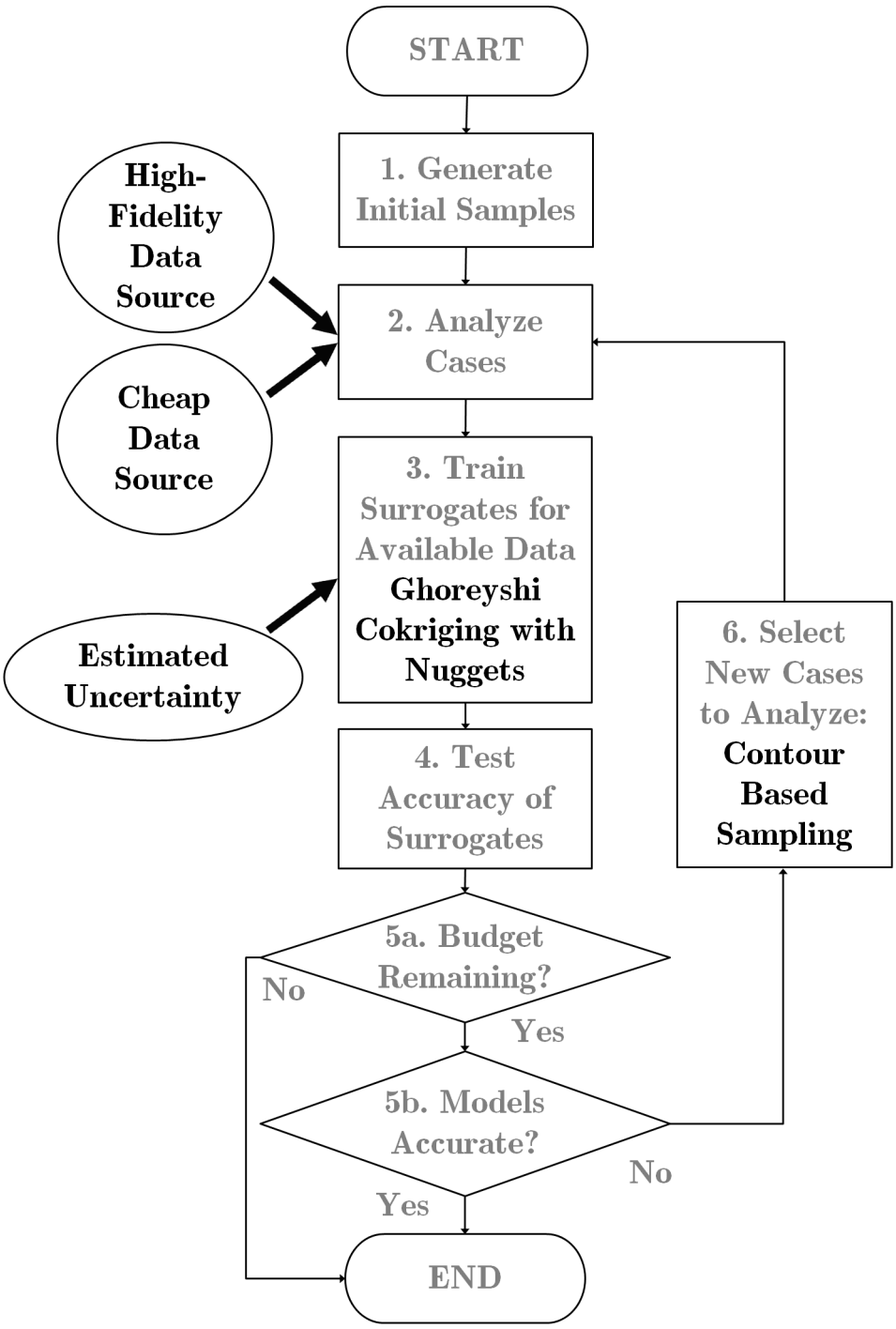

Figure 64: Updated Methodology for Sample Selection & Surrogate Model Creation

In these experiments, a separate set of test samples were used to evaluate the predictive accuracy. Cross validation was not used because data was (relatively) cheap, allowing some of it to be used purely for test purposes, and because of the particular characteristics of the problems being addressed: for some responses, predictive accuracy was only important for cases with response values within a certain range. Had cross validation been used with the data sets that were available, only a handful of points would have been fallen within that range, leading to large uncertainty in the error estimates.

Steps 5a & 5b: If the surrogate models are sufficiently accurate or the project resources have been consumed, terminate the process.

Step 6: Otherwise, select new samples for analysis using *contour-based sampling* and go to Step 2.

Contour-based sampling was selected as the adaptive sampling method of choice in light of the experimental results. Before applying contour-based sampling, the user should refer to validation test results for all data sources to determine what the response range(s) of interest may be, accounting for any observed biases or uncertainty in the results.

The method defined by these steps was shown to perform well for problems which match the conditions for which it was designed: design spaces which are large but contain only a small feasible space constrained by the allowable values of one or more responses; responses which can be approximated moderately well using a cheaper source of data; and responses with a degree of uncertainty that is large relative to the scale of the response being modeled. When one or more of those qualities is not present, the proposed method may not have an advantage over the baseline approach of space-filling samples and deterministic single-fidelity modeling.

In the course of developing and demonstrating this method, a number of contributions to the field of advanced design methods were made. These contributions will be reviewed in the next section.

7.3 Contributions

The contributions that resulted from this research effort may best be expressed by grouping them by topic. On the topic of sample selection techniques:

- Contour-based sampling has been extended to address multiple responses simultaneously.
- The Probabiliy of Interest (POI) requirement was introduced to allow the user to determine the algorithm's behavior and balance exploration of regions that may be of interest against exploitation of regions that are known to be of interest.
- The effects of varying the POI requirement were demonstrated for a representative aerospace problem, illustrating the changes in algorithmic behavior that resulted.
- The computational savings produced by the use of Schur's Complement rather than direct matrix inversion were quantified for a problem with dozens of variables.
- Stacked Latin hypercubes were proposed and demonstrated. Tests showed them to have superior space-filling qualities when compared against other progressive sampling approaches found in the literature, although the costs of creating the sample design were much higher.
- It was shown that these well-spread-out sample sets could be difficult to fit using Kriging models; when the design space is large or very few samples will be available, it may be preferable to allow moderate clustering so that the Kriging surrogate can better identify correlation in the data.

On the topic of the creation of surrogate models:

- Data harmonization was demonstrated for the first time for a problem outside its field of origin, geostatistics.
- A variant of Ghoreyshi cokriging, in which all low-fidelity response values were incorporated rather than only the one which corresponded to the high-fidelity response

being modeled, was implemented and evaluated. It was not found to offer improved performance.

- Although the use of nuggets to capture uncertainty for Kriging models was shown to improve the accuracy of those models when fitting noisy responses, it was demonstrated that such information is not preserved in the prediction variance of the model.

7.4 Future Work

The work presented in this document was intended to be thorough, but by no means exhaustive. There are a number of avenues for further work through which the proposed method might be improved or augmented:

Most importantly, at the present time the user must set values for the probability of interest (POI) requirement, the number of candidate points to evaluate, and the number of test points to use for evaluation when using contour-based sampling. The values selected can significantly affect the behavior of the algorithm, as shown in Section 4.9 and Appendix C, but at present it is difficult to determine what the most effective values should be.

In addition, it is expected that as the number of candidate and test points are increased, some point of diminishing returns would be encountered – for example, beyond some density of test points, adding an additional test point might not increase the accuracy of the algorithm. Although the effectiveness of the additional test point would be reduced or negated, there would still be an incremental cost to evaluate how that test point is affected by each candidate point. Similarly, although a larger number of candidates would be expected to increase the chance that a highly informative candidate might be identified, a similar point of diminishing returns without diminishing costs might be expected.

Also of interest is the other end of the spectrum: how few test points is *too* few? At what point does the decreased accuracy when evaluating candidates outweigh the computational savings? How small can the pool of candidates become before the effectiveness of the sampling algorithm becomes handicapped?

It is possible that the optimal value (or schedule of values) for each of these parameters could be highly problem-dependent. For example, in Section 4.9 it was shown that POI

requirements much above 10% served to make the algorithm overly-cautious, limiting its ability to identify the region of interest. This behavior stemmed from the fact that, especially early on, the surrogate models were poor representations of the responses being emulated. As a result, cases which appeared promising were of little interest and vice versa. Without low POI requirements, the algorithm could not do the exploration necessary to build an accurate understanding of the responses. Once the models had improved, a higher POI requirement might have led to sampling that better clustered cases near the region of interest. If this behavior could be anticipated, an efficient schedule of POI requirements and candidate & test pool sizes could be developed, whether that schedule be universal or tunable to various types of problems.

Lastly, it was shown that sparse sample sets led to difficulty in fitting accurate Kriging surrogates, and that clustering helped to address that difficulty. However, the details of that clustering have yet to be addressed rigorously. Presently the only way that an overly-sparse data set may be identified is when the DACE Kriging toolbox fails to find useful correlation coefficients. After this occurs, any attempts to address the problem – typically by clustering more samples near existing samples, whether manually or by setting a high POI requirement for the sampling algorithm – would be by their nature reactive rather than proactive. An investigation into the effects of sparsity, as well as ways to predict and/or mitigate those effects, would be greatly beneficial when the data set may be sparse.

7.5 Final Remarks

If more accurate information could be made available to decision-makers early in the design process, trade studies and optimizations could be carried out with greater confidence. By ensuring that decisions are made using sufficiently accurate understanding of all consequences, the risk that later analysis will reveal a previously-unsuspected deficiency in the design may be reduced or eliminated.

The process of selecting experiments to acquire the most useful data is a perennial problem, and often a thorny one. It becomes even more challenging when resources are in short supply and no analysis can be wasted. In such situations, special care must be taken

to ensure that each analysis that is performed is as informative as possible. This process can be complicated when several responses must be taken into account at once.

This research effort attempted to contribute to those goals by identifying: techniques for selecting the minimum set of samples while maximizing the useful information obtained; techniques for leveraging cheaper sources of data to reduce dependence on accurate-but-expensive analyses; and techniques for identifying, tracking and capturing any uncertainty present in the data being modeled. The resulting approach to sampling and modeling was shown to be effective for the most part, improving predictive accuracy while reducing the number of expensive analyses required.

The proposed method is not a silver bullet – for example, the time & effort required to select each sample may be non-negligible. This method is most effective when the per-evaluation cost of the primary data source is high compared to the costs of the sample selection process. Although the problem of expensive analyses is not entirely negated, it is hoped that this effort has contributed a useful step, however small, toward bringing the problem down to manageable size.

APPENDIX A

A PRACTICAL GUIDE FOR EFFICIENT SURROGATE MODELING

> "The result is statements of undue length whose persuasive power is attributable solely to their strangeness and which impress the reader only by the abstract quality of their vocabulary, which moreover is ill-defined." – André Breton[22]

A.1 Overview

The goal of this work is to create accurate surrogate models while minimizing the cost of the necessary data. The surrogate models will emulate analysis methods such as computational tools or physical measurements. The method described in this guide combines three techniques – adaptive sampling, data fusion, and Kriging nuggets – which serve to improve predictive accuracy and/or reduce the cost of acquiring the data required to train useful surrogate models.

This guide will help the reader apply the method. The guide assumes that the reader has already set up the problem, i.e. that reader has already identified the independent and dependent parameters (e.g., the inputs & responses).[96] If possible, screening tests should be performed to minimize the number of independent parameters. The user should also have selected appropriate sources of data (e.g., CFD models or physical measurements) based on the level of accuracy that is required: greater accuracy typically requires more expensive sources of data. The guide will assume that the reader is familiar with advanced design methods such as surrogate modeling and design space exploration; readers unfamiliar with these topics may wish to review the survey articles of Shan & Weng[174, 189] and the text *Response Surface Methodology* by Myers, Montgomery, and Anderson-Cook.[134]

This method may not be appropriate for problems that are highly resource-constrained. A common rule of thumb for the amount of data necessary for accurate surrogate model is at least $10d$ applications of each data source, where d is the number of dimensions (i.e. free

parameters) being investigated simultaneously.[58, 106] If resource limits will not allow the user to carry out the recommended $10d$ analyses using the desired data source, it is unlikely that an accurate surrogate model can be trained. There are exceptions – if the response behavior is very simple, if the ranges of the input parameters are very small, etc. – but any problem which requires such expensive analyses is unlikely to be so accommodating. If the problem is resource-constrained in this manner, a surrogate model may be trained to match cheaper data sources, while the expensive data source is used directly for verification and/or correction of the cheaper data in a manner similar to the Pegasus booster design process.

During the design of the Pegasus booster in the early '90s, project resource limitations meant that only a few expensive Navier-Stokes simulations could be performed. The designers used cheaper sources of data to design the vehicle & trajectory. The expensive simulations were used to investigate phenomena that the cheaper simulations wouldn't capture, such as interactions between shock waves and boundary layers, to determine whether those phenomena would significantly affect the vehicle's performance. The expensive results were also used to confirm some of the predictions of the cheaper data sources. This gave the designers confidence in the cheaper simulations while minimizing the use of expensive simulations.[130] If such an approach is not acceptable, the user is urged to reduce the scope of the problem by eliminating free parameters (thus reducing the $10d$ samples recommended) or using an alternative, cheaper data source.

This remainder of this guide will assume that the problem at hand has at least enough resources to perform $10d$ analyses with each data source. This guide describes a method for making the best use of those analyses. The steps of the method are given in detail beginning in Section A.3. In some sections, example commands are given to clarify how to accomplish certain tasks, such as creating a surrogate model that combines multiple sources of data. These example commands are written for Matlab, as this was the environment used for the original implementation.

The method being described was developed for predicting aerodynamics, so the description will be in those terms. Note that, although it was developed for an aerodynamics problem, the method is not restricted to that field and can be applied to any problem that

has the appropriate characteristics. Those characteristics are specified in the next section.

A.2 Intended Applications

First, the problem of interest must be **complex** enough that rapid analysis methods, such as handbook methods or panel methods, are not sufficiently accurate. Instead, more complex methods, such as computational fluid dynamics (CFD), are necessary to capture all the relevant phenomena. Use of these complex methods means increased effort per analysis, whether physical or computational. Due to the increased cost of data, it may not be possible to construct adequate surrogate models using standard techniques. For simple problems, where phenomena such as viscous or nonlinear effects are insignificant, there is no incentive to use any special techniques to reduce the computational effort. Simple problems can often be analyzed by quick-to-execute tools in less than a second. It is usually more effective to analyze a large number of than it is to spend time calculating which single sample would be the most informative to obtain.

The term *sample* in this context refers to the combination of inputs to, and outputs from, one analysis. If a computational tool is used, a sample is obtained when the data source is used to determine the response(s) for a particular set of input values. The term *analysis* refers to the process of determining the response values that correspond to a particular set of input values. An analysis may be the application of a computer model or the measurement of a physical system. Part of the goal of the technique is to identify the most effective & informative samples to analyze.

Secondly, this technique is most effective when **only a certain range of the response is of interest.** The technique seeks to improve predictive accuracy for samples likely to have a response value within that range. If there are multiple responses, the technique seeks to improve predictive accuracy for samples which fall within the specified ranges for *all* responses.

As an example, the technique was developed to create accurate models of vehicle performance at multiple flight conditions. It was found that the pitching moment coefficient was a critical response when evaluating a possible design: if the pitching moment coefficient

at any flight condition was too large , the design would probably not be controllable. As a result, this technique helped to identify designs that were likely to have small pitching moment coefficients at all flight conditions. Those designs, once analyzed, could be used to improve the surrogate models. In this case, the response values of interest were defined as a range.

Alternatively, this technique has been used for predicting structural failure. The user would define the response threshold(s) of interest, e.g. the limit strength of a beam, and the technique would identify analyses that would enhance the surrogate model's accuracy when predicting whether or not the response for a given sample would exceed the specified limit. By performing those analyses and adding them to the data set used to train the surrogates, the surrogate models became better predictors of system reliability than when this technique was not applied. In that case, the response values of interest were defined using one threshold value for each response.

Thirdly, this technique is intended for problems where **multiple sources of data are available**. Typically the sources of data will have different levels of expense associated with them: a computational model might take a few minutes or hours to complete one analysis, while a flight test might require months of preparation and millions of dollars. When there is a large discrepancy in the expense associated with the available data sources, a well-designed set of analyses can offer significant savings over a more haphazard approach. It is possible to incorporate any number of data sources,[83] but for the sake of simplicity this guide will assume that only two sources of data are available and that one data source has much greater per-analysis costs than the other.

Most likely, the cheaper data source is a computational model. This data source is unlikely to be accurate enough for use as the main source of data (or else why use the more expensive source at all?). Still, the cheaper source should be accurate enough to capture trends in the response behavior. For an aerodynamics analysis, the main source of data might be wind tunnel analyses while the cheaper source of data might be a computational tool based on the Euler or Navier-Stokes equations.

Lastly, the technique is intended for problems where **significant uncertainty may be**

present in one or more responses. This uncertainty can stem from many sources, such as difficulty in measuring the response, random noise, or errors introduced by the numerical solver of a computational tool.

The technique can be applied to problems which do not have all four of the characteristics described in this section. If one or more characteristics are not present, the technique can be adjusted to take advantage of that fact; likely adjustments will be called out during as needed in the following sections.

A.3 Setting Up the Problem

As previously stated, this guide assumes that the user has already identified appropriate data sources, input variables and responses. A number of techniques exist to help the user minimize the number of input variables, such as screening, parameter mapping, and decomposition.[174] If the data sources have not been validated for the problem at hand, validation tests should be performed to quantify the accuracy of each data source for the relevant applications: in vehicle aerodynamics, for example, validation tests should be performed at each flight condition of interest, as aerodynamics analysis tools may be accurate for some flight conditions yet perform poorly for others. In a sense, **validation** may be considered **Step Zero**, as it must be completed before the technique can be applied. Validation compares the response values predicted by the data source in question against the "true" values, which come from some higher-fidelity data source such as physical measurements. This validation data may be purpose-generated or compiled from the available literature.[32, 59, 188, 190]

"Fidelity" in this context refers to "the degree to which a model is an accurate representation of the real world for the intended uses of the model".[8] For an aerospace vehicle, the highest-fidelity data source would be a full-scale test of the vehicle at every relevant flight condition. Naturally, generating that data can be quite expensive. Instead, simpler data sources such as wind tunnels may be used, although it is important to make sure that the simplification does not leave out any important effects such as those dependent on Reynolds number.[17]

A.4 Selecting Initial Analyses

Once the data sources have been validated, it is time to choose the analyses that will be performed. This **selection of the initial analyses** constitutes **Step One**. The process of selecting these analyses will depend heavily on the number of analyses that will be performed using each data source.

This guide is intended for the situation where each data source can provide dozens or hundreds of analyses. In such a situation, a common rule of thumb for the number of initial analyses is $10d$ for each data source, where d is the number of dimensions (i.e. free parameters) being investigated simultaneously.[58, 106]

Once the number of initial analyses has been selected, the user must determine *which* analyses to run. The analyses should be chosen to maximize the knowledge gained. If expert knowledge is available for the problem at hand, it may be easy to identify the most useful analyses. On the other hand, if the response behavior is not known well, it may be better to take a more universal approach: space-filling sampling.[170]

Space-filling sampling is a way of choosing analyses which attempts to spread those analyses out as much as possible. This approach will roughly illustrate how the response varies throughout the design space. Latin hypercubes are the most common form of space-filling sampling, but other techniques such as Sobol sequences are becoming more popular. Reviews of many available techniques can be found in the works of Chen et al. and Shan & Weng.[27, 174, 189]

If multiple data sources will be available and the user plans to apply a multi-fidelity method (see Section A.5.3), it may be worthwhile to take this into account when selecting the analyses to be performed. Researchers have proposed a variety of ways to design an set of samples to make it highly compatible with multi-fidelity modeling. These approaches to analysis design include methods such as nested or sliced Latin hypercubes.[95, 151, 152, 153]

Alternatively, it may *not* be plausible to expect dozens or hundreds of results from each data source. This is common when physical measurements are made, as acquiring those results is often *much* more expensive than a computational analysis. In this case, another approach is necessary. Instead of spreading the expensive samples out, emphasis should be

placed on identifying the most useful samples to be analyzed. The process of identifying useful analyses will depend on two sources of knowledge: *validation results*, and *data from cheaper analyses*.

The results from the validation tests in Step Zero are useful for a number of reasons. First, the user can identify the situations where the cheaper data source can meet the user's accuracy requirements. In those situations, the more expensive data source may not be necessary at all. Conversely, the user can identify situations where the cheaper source of data is expected to have particularly poor accuracy, and thus where the cheaper source should not influence the selection of expensive analyses. Expert knowledge – in particular, knowing which phenomena affect the response in each situation and comparing those against the phenomena that each data source can capture – may also help in this regard.

The validation results can also help the user to identify any consistent biases that may be present in the predictions of the cheaper data source. These biases are then used to partially "correct" the predictions of the cheaper data source. Once this is done, the cheaper data source is used to explore the design space and identify promising regions. For example, if the goal is to find regions of the design space where the "true" response is close to zero, and the cheap data source over-estimates the response by an average of 5 units, the first few expensive analyses should be placed in regions where the cheap data source predicts the response is close to 5. This is a very rough approach to sample selection, but it can be a good way to get started.

A.5 Training Surrogate Models

Step Two is to **train one or more surrogate models**. Surrogate modeling is way to estimate the behavior of a response using mathematical techniques. Once a surrogate model is trained, it is very computationally cheap to predict the value of the response for some new set of input values. This is particularly important for design space exploration, which is the process of investigating the response value for different combinations of input values, as exploration can require a large number of analyses.

A number of different surrogate modeling techniques have been described in the literature, including Response Surface Methods, Kriging, Artificial Neural Networks, Radial Basis Functions, and Gaussian Process Models. Reviews of many available techniques and their relative merits can be found in the works of Chen et al. and Shan & Weng.[27, 174, 189] This guide will assume the use of Kriging. DACE, a useful Kriging toolbox for Matlab, can be downloaded from `http://www2.imm.dtu.dk/~hbni/dace/`.

Kriging is used in this guide primarily because it can explicitly account for uncertainty in the data. When fitting a Kriging model, a covariance matrix is used to represent the relationships between the training samples. If uncertainty is present in the data, that uncertainty can be captured by adding "nuggets" to the diagonal of the covariance matrix. The DACE toolbox presently does not allow user-defined nugget values, but that is easy to remedy.

If there is no significant uncertainty present in the data, the reader can skip Sections A.5.1 and A.5.2.

A.5.1 Implementing Nuggets in DACE

This implementation allows the user to specify nugget values while minimizing changes to the existing DACE functions. As a result, the command sequence to make a Kriging model with nuggets is inelegant but functional, as will be demonstrated in Section A.5.2.

First, create a duplicate copy of **dacefit.m** named **dacefitNugget.m**. Change line 1 of **dacefitNugget.m** from

```
function [dmodel, perf] = dacefit(S, Y, regr, corr, theta0, lob, upb)
```

to

```
function [dmodel, perf] = dacefitNugget(S, Y, regr, corr, theta0, lob, ...
   upb, nug)
```

This tells the modified function to expect an extra input parameter, which will be the nugget value or values. Next, change line 93 from

```
  'D', D, 'ij',ij, 'scS',sS);
```

to

```
'D', D, 'ij',ij, 'scS',ss, 'nuggets',nug);
```

This incorporates the nugget parameter into the structure *par* which contains the data necessary to evaluate how well the Kriging model fits the data. Next, modify line 114 from

```
'C',fit.C, 'Ft',fit.Ft, 'G',fit.G);
```

to

```
'C',fit.C, 'Ft',fit.Ft, 'G',fit.G, 'nuggets',nug);
```

This adds the nugget values to the structure *dmodel*, which is the trained Kriging model that is returned by the function. Adding the nugget values that were used to train the model is not strictly necessary, but may be helpful when documenting the Kriging model. Lastly, change line 129 from

```
[r(idx); ones(m,1)+mu]);
```

to

```
[r(idx); ones(m,1)+mu+par.nuggets]);
```

This adds the nugget values to the diagonal elements of the covariance matrix. No other functions need to be changed, as nuggets only have a direct effect on the surrogate training process. Predictions made with the resulting Kriging model will take the nuggets into account without any further modifications.

A.5.2 Using the Modified DACEFIT Function

With the standard **dacefit.m** function, the user can fit a model using the command

```
dmodel = dacefit(S, Y, TrendType, Correlation, theta, lob, upb);
```

Here, S is the matrix of input values; Y is the vector or matrix of response values; $TrendType$ specifies whether the underlying trend of the model is constant, linear or quadratic (or in terms of the functions supplied with the DACE toolbox, *@regpoly0*, *@regpoly1* or *@regpoly2*);

and *Correlation* specifies the desired correlation model (e.g. exponential, Gaussian, linear, etc., again referring to the functions included with DACE such as *@correxp*). The remaining parameters represent the initial guess for correlation coefficient values (*theta*) and the limits on those coefficients: *lob* is the lower bound while *upb* is the upper bound.

The first step of creating a Kriging model using nuggets is to determine the value of the nugget. The nugget can be a scalar or a vector. If the uncertainty is constant for every training sample, the nugget will be a scalar equal to the variance of that uncertainty. If the uncertainty is different for each training sample, the nugget will be a vector of size $(n \times 1)$, where n is the number of training samples, and the i^{th} entry will be the variance of the uncertainty in the response for the i^{th} training sample.

Note that before it can be used to fit a Kriging model, the nugget must be scaled by the process variance of the Kriging model. The process variance is calculated by DACE, so the set of commands that fit a Kriging model using nuggets is almost recursive:

```
dmodel_noiseless = dacefit(S, Y, TrendType, Correlation, theta, lob, upb);
scaled_nugget = original_nugget/dmodel_noiseless.sigma2;
dmodel = dacefitNugget(S, Y, TrendType, Correlation, theta, lob, upb, ...
   scaled_nugget);
```

The resulting *dmodel* is a Kriging model which captures the uncertainty specified in the nugget value(s). The commands to use *dmodel* to estimate response values and prediction confidence are identical to those for a nugget-less Kriging model, and are given in the manual that is included with the DACE toolbox download.

A.5.3 Multi-Fidelity Surrogate Modeling

If the reader will only be using one data source, this section may be safely skipped. If multiple data sources will be available, it may be possible to combine data from each source into a single surrogate model. This new surrogate is often more accurate than if only the cheaper data source were available, yet less expensive than if only the more expensive data source were used. The methods used to create such surrogates are known as multi-fidelity methods.

Multi-fidelity methods enable the user to train surrogate models that emulate the more expensive data source while reducing the number of expensive analyses necessary to achieve the desired accuracy. Rather than having to infer response behavior purely from the results of expensive analyses, multi-fidelity methods use cheaper analyses to learn how the response varies throughout the design space. Those results are then "corrected" using results from the more accurate (but more expensive) data source.

For example, consider the problem of predicting the lift coefficient of a vehicle at multiple angles of attack. Simple linear aerodynamics tools can often estimate how the lift coefficient *changes* with angle with good accuracy (at least for small angles) but may be less accurate when predicting the lift coefficient at any one particular angle. More complex methods like computational fluid dynamics (CFD) offer better predictions of lift coefficient but require a much larger computational effort. Multi-fidelity methods would use the cheaper tools to estimate how the response (e.g., lift coefficient) changes with angle, and correct the response at each angle based on one or two results from the more accurate CFD analysis.

The savings offered by multi-fidelity methods will depend on how much similarity there is between the behavior of the response as estimated by the two data sources. If there is very little similarity, multi-fidelity methods will not offer much benefit – but if there is so little similarity, there may be no point in using the cheaper source of data at all.

A number of multi-fidelity techniques have been developed and used for engineering purposes. Some create two surrogate models, one to emulate the cheaper source of data (f_{cheap})[1] and one to "correct" the cheaper result to match the more expensive source of data. This type of multi-fidelity technique is very popular, but assumes that there will be enough expensive results ($f_{expensive}$) to create a surrogate model (preferably $> 10d$, where d is the number of independent parameters). This family of methods includes:

- **Additive correction:** one surrogate is trained to emulate f_{cheap}, another is trained to emulate $(f_{expensive} - f_{cheap})$, and the predictions of both are added together to estimate $f_{expensive}$.[73]

[1] If the cheaper source of data is *very* cheap, it may be possible to use it directly instead of creating a surrogate model.

- **Proportional correction:** one surrogate is trained to emulate f_{cheap}, another is trained to emulate $\left(\frac{f_{expensive}}{f_{cheap}}\right)$, and the predictions from both surrogates are multiplied together to produce an estimate of $f_{expensive}$.[9]

- **Hybrid correction:** a combination of additive and proportional correction.[94, 155]

- **Ghoreyshi cokriging:** one surrogate is trained to emulate f_{cheap}; the estimated f_{cheap} value is then treated as an extra input parameter when fitting a surrogate to $f_{expensive}$.[63]

Alternatively, some multi-fidelity techniques create only one surrogate model, using all the available data at once. Because the effort of training a surrogate model increases as more data is used, these methods may become quite computationally-intensive for large data sets. On the other hand, if the user will not have the resources to run $> 10d$ analyses with the expensive data source, these methods can still produce a surrogate that incorporates all available data. These techniques include:

- **Cokriging:** a form of Kriging that can handle multiple responses (or the same response calculated by multiple data sources).[182] Some software tools are available that can perform cokriging, but most can only handle up to three independent parameters.[145, 164]

- **Data harmonization:** similar to additive correction, data harmonization attempts to capture any biases between data sources. The source of each sample is identified using binary columns.[13, 14, 15]

A.5.4 Demonstrating Ghoreyshi Cokriging

Ghoreyshi cokriging was selected for a more in-depth demonstration, as it produced the most accurate surrogate models for a particular test problem. Note that the effectiveness of each multi-fidelity method is strongly problem-dependent, and although Ghoreyshi cokriging produced the most accurate surrogate model for one application, another method might be superior for a different application. If possible, the user should do comparison tests

to determine which method is best for the problem at hand. The process of evaluating surrogate model accuracy is the subject of the next section of this guide.

To understand the implementation of Ghoreyshi cokriging, consider a problem with two input variables, x_1 & x_2. The matrix of inputs for a standard (non-multi-fidelity) Kriging model with a linear underlying trend would resemble:

$$S = \begin{bmatrix} x_{1,1} & x_{2,1} \\ x_{1,2} & x_{2,2} \\ x_{1,3} & x_{2,3} \\ x_{1,4} & x_{2,4} \end{bmatrix} \tag{1}$$

Here, $x_{2,1}$ represents the value of x_2 for the first training sample. The matrix of inputs for a Ghoreyshi cokriging model with a linear underlying trend, on the other hand, would take the form:

$$S = \begin{bmatrix} x_{1,1} & x_{2,1} & \hat{f_{cheap,1}} \\ x_{1,2} & x_{2,2} & \hat{f_{cheap,2}} \\ x_{1,3} & x_{2,3} & \hat{f_{cheap,3}} \\ x_{1,4} & x_{2,4} & \hat{f_{cheap,4}} \end{bmatrix} \tag{2}$$

where $\hat{f_{cheap,i}}$ represents the estimated response (as determined by the cheaper data source) at the i^{th} sample. Thus, a problem with d input dimensions becomes one with $d+1$ input dimensions. The process of fitting the Kriging model (or any other form of surrogate model) is otherwise unchanged. When using the DACE toolbox for Matlab, a Ghoreyshi cokriging model may be created using the following commands:

First, assuming the per-analysis cost of the cheaper data source is not negligible, a surrogate model may be trained to emulate the cheaper data source. Analyses are performed using that data source and a Kriging surrogate model is trained to match the results:

```
dmodel_cheap = dacefit(S_cheap, f_cheap, TrendType, Correlation, ...
   theta, lob, upb);
```

Here, *S_cheap* is the matrix of input values that were analyzed with the cheaper data source, while f_{cheap} is the vector or matrix of response values from the cheap data source. *TrendType* specifies whether the underlying trend of the model is constant, linear or quadratic. *Correlation* specifies the desired correlation model (e.g. exponential, Gaussian, linear, etc.). The remaining parameters represent the initial guess for correlation coefficient values (*theta*) and the limits on those coefficients: *lob* is the lower bound while *upb* is the upper bound.

Next, a smaller set of analyses (*S_expensive*) is performed using the more expensive data source, and the surrogate model of the cheap data source (*dmodel_cheap*) is used to estimate the response values that would be obtained if the same smaller set of analyses were performed using the cheaper data source. The estimated cheaper response values (*fcheap_pred*) are obtained with the command:

```
fcheap_pred = predictor(S_expensive, dmodel_cheap);
```

Finally, a Ghoreyshi cokriging model is created using both the cheap and expensive data by including the cheaper response values as an extra input parameter:

```
dmodel_expensive = dacefit( [S_expensive fcheap_pred], f_expensive, ...
   TrendType, Correlation, theta_extra, lob_extra, upb_extra);
```

Note that, because there is one more input parameter, the vectors for *theta*, *lob*, and *upb* must each be enlarged by one entry. *theta_extra*, *lob_extra* and *upb_extra* represent those enlarged vectors.

To predict the expensive response for some new sample using this Ghoreyshi cokriging model, the user must also predict the cheap response for that sample:

```
fcheap_newsample = predictor(S_new,dmodel_cheap);
fexpensive_newsample = predictor( [S_new fcheap_newsample], ...
   dmodel_expensive);
```

Here, *fexpensive_newsample* is the prediction of the Ghoreyshi cokriging model for the response value for the new sample.

At this point in the process, initial samples have been selected and analyzed using the various data sources. Using the results from those analyses, surrogate models have been trained to estimate the response(s) of interest. The question may then be raised: are those surrogate models accurate enough?

A.6 Evaluating Surrogate Model Accuracy

Step Three is to quantify the predictive accuracy of the surrogate models that have been trained. Evaluation of surrogate model accuracy is typically done in one of two ways: cross validation or test samples.[88, 148]

A.6.1 Test Samples

Evaluation using **test samples** is simpler conceptually, but requires more data than cross validation. To evaluate surrogate model accuracy using test samples, some number of analysis results are set aside and *not* used to train the surrogate model. The more test samples available for this purpose, the more confidence the user can have in the prediction error estimate for the surrogate. If the user is only concerned with the predictive accuracy for *samples with response values within a certain range*, only samples with responses within that range should be used for testing; obtaining useful test samples in this scenario may require extra effort, such as a separate optimization process.

Once a surrogate has been trained, it is used to predict the response value for each of the test samples. These predictions are then compared with the observed results to calculate the prediction error for the test set. For each sample in the test set, the prediction error (e_i) is the discrepancy between the observation (y_i) and the prediction ($\hat{y}_i$):

$$e_i = (y_i - \hat{y}_i) \tag{3}$$

Once the prediction errors for all n samples in the test set have been calculated, they can be used to evaluate the overall accuracy of the surrogate model. First, the average

prediction error ($\bar{e}_{ts}$) is calculated:

$$\bar{e}_{ts} = \frac{1}{n}\sum_{i=1}^{n} e_i \tag{4}$$

This value indicates whether there is any consistent bias in the predictions of the surrogate, and ideally should be close to zero.

Next, the spread of the prediction errors, ϵ_{ts}, is calculated to determine how precise the surrogate model is, i.e. how much variability is present in the prediction error.[10] A model with some bias that is consistently within 5% of the correct value may be more useful than a model that has no average bias but is occasionally off by 50%. To assess this, the spread of the error is first calculated:

$$\epsilon_{ts} = \frac{1}{n}\sum_{i=1}^{n} (e_i)^2 \tag{5}$$

Here, ϵ_{ts} has units of variance, i.e. the units of the response squared, which may be difficult to interpret. The square root of ϵ_{ts} may be easier to work with:

$$RMSE_{ts} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (e_i)^2} \tag{6}$$

This term, $RMSE_{ts}$, is the Root Mean Squared Error and has the same units as the response of interest. A small $RMSE_{ts}$ value indicates that the surrogate model prediction errors did not exhibit large variations.

By combining the average prediction error ($\bar{e}_{ts}$) and the deviation of the prediction error ($RMSE_{ts}$), the user can estimate a confidence interval for a future prediction $\hat{y(x)}$. First, the expected bias is accounted for:

$$\begin{aligned} \hat{y(x)}_{unbiased} &= \hat{y(x)} + \bar{e}_{ts} \\ &= \hat{y(x)} + \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)}{n} \end{aligned} \tag{7}$$

Next, assuming the prediction error is normally distributed, a 95% confidence interval can be estimated:

$$\begin{aligned} \hat{y(x)}_{lower95\%} &= \hat{y(x)}_{unbiased} - 2 \times RMSE_{ts} \\ \hat{y(x)}_{upper95\%} &= \hat{y(x)}_{unbiased} + 2 \times RMSE_{ts} \end{aligned} \tag{8}$$

These values let the user estimate how much uncertainty is present in the predictions of the surrogate model. The user may assume that, in light of the available data, there is a 95% likelihood that the actual response $y(x)$ falls somewhere between $\hat{y(x)}_{lower95\%}$ and $\hat{y(x)}_{upper95\%}$.

The argument against the use of test samples for accuracy evaluation is that some analytical effort – the effort required to analyze the test samples – does not contribute to improving the surrogate model. If the per-analysis cost is very high, this may be considered too wasteful. Iooss proposed a way to distribute the test samples throughout the design space so that the user can get the most accurate assessment of surrogate model accuracy for the least possible number of test samples, although it is difficult to know in advance how many test samples will be required.[87]

A.6.2 Cross Validation

The alternative to using separate test samples is **cross validation**. In cross validation, all the available data is used to create the primary surrogate model. This primary surrogate is then set aside. Then, the available data is split up into k groups, where k is some integer. All but one group of data are used to train a new surrogate model, and that surrogate is used to predict the response values for the samples that were omitted. The resulting predictive errors are used to infer the predictive accuracy of the original surrogate model. This process is known as "k-fold cross-validation".

As with the test-samples approach, if the user is trying to quantify the predictive accuracy of the surrogate for *samples where the response value lies within a specified range*, the cross validation procedure may be performed by progressively omitting only the samples which have response values in that range. If most of the training samples do not have response values in the range of interest, the estimate for prediction error may itself have low confidence.

The best group size to use is a matter of some debate in the literature. The limiting case is to leave out one sample each time. For a data set of N samples, N new models would be have to be created, each of which omits the i^{th} training sample and then attempts to predict

the response at that omitted sample. The prediction error for the i^{th} cross validation model – which omitted the i^{th} sample from its training data set – would be:

$$e_i = (y_i - \hat{y}_i) \tag{9}$$

Here y_i is the actual response value for the i^{th} sample, and $\hat{y}_i$ is the predicted response using the i^{th} cross validation model. The overall cross validation error, ϵ_{cv}, is calculated as:

$$\epsilon_{cv} = \frac{1}{n}\sum_{i=1}^{n}(e_i)^2 \tag{10}$$

Like ϵ_{ts}, ϵ_{cv} has units of variance, so another calculation is done to make the results easier to interpret:

$$RMSE_{cv} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(e_i)^2} \tag{11}$$

This term, $RMSE_{cv}$, is the Root Mean Squared Error and has the same units as the response of interest. A small $RMSE_{cv}$ value indicates that the surrogate model prediction errors were consistently small and did not exhibit large variations.

As with the test samples approach, the user can estimate a confidence interval for a future prediction $\hat{y(x)}$ using the average prediction error ($\bar{e}_{cv}$) and the deviation of the prediction error ($RMSE_{cv}$). First, the expected bias is accounted for:

$$\begin{aligned} \hat{y(x)}_{unbiased} &= \hat{y(x)} + \bar{e}_{cv} \\ &= \hat{y(x)} + \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)}{n} \end{aligned} \tag{12}$$

Next, assuming the prediction error is normally distributed, a 95% confidence interval can be estimated:

$$\begin{aligned} \hat{y(x)}_{lower95\%} &= \hat{y(x)}_{unbiased} - 2 \times RMSE_{cv} \\ \hat{y(x)}_{upper95\%} &= \hat{y(x)}_{unbiased} + 2 \times RMSE_{cv} \end{aligned} \tag{13}$$

As before, the user may assume that, in light of the available data, there is a 95% likelihood that the actual response at x, $y(x)$, falls somewhere between $\hat{y(x)}_{lower95\%}$ and $\hat{y(x)}_{upper95\%}$.

This approach can provide a reasonably good estimate of the prediction error of the surrogate trained with *all* the samples, but there are three points of concern. First, the

user must train N extra surrogate models, which may become very time-consuming for larger data sets.[58] Secondly, if certain training samples have a large effect on the surrogate model, there may be a large spread in the cross validation prediction errors, which means there may be a lot of uncertainty in the cross validation error estimate. Finally, cross validation only evaluates the accuracy of surrogate models trained with *some* of the data, so the true predictive accuracy of the model may be under-estimated.

In some cases, splitting the data up into 5 or 10 larger groups may address some of those concerns. By omitting each group one at a time for cross validation instead of each sample individually, the number of extra surrogate models that must be trained is reduced from N to 5 or 10. If the data pool is large enough this may also result in a smaller spread of prediction errors, reducing the uncertainty in the cross validation error estimate.[100]

However, if the amount of data is limited, the removal of 10-20% of the data for k-fold cross validation may significantly affect the resulting surrogate models, leading to a substantial over-prediction of surrogate error.[75] The user may screen for this effect during cross validation by tracking the model parameters, such as correlation coefficients (e.g., the *theta* parameters used by the DACE toolbox), for each new surrogate produced. If these parameters vary significantly, the user may wish to reduce k to split the data into smaller groups.

A.6.3 Emphasizing Certain Response Value Ranges

If the user is not attempting to create a surrogate model that is accurate over the entire range of the response, but rather wishes to emphasize a certain range of response values, the estimation of predictive accuracy becomes slightly more complex. This was noted briefly in the previous sections.

In order to quantify the predictive error of the surrogate model for samples with response values within a certain range, the user should only use samples that fall within the range of interest when selecting test samples or cross validation groups. By using only those samples, the user ensures that the estimated confidence intervals are as relevant as possible to the expected use of the surrogate. Depending on the behavior of the response and the size of

the design space, however, there may not be many samples available which fall within the desired range.

In such a scenario, the user is faced with tough options:

- **Work With What's Available.** The user may simply go ahead and use the available samples within the range of interest. If very few such samples are available, this approach may result in error estimations that have high uncertainty. However, if the uncertainty of the error estimates are reported along with the predictions, that result may be good enough.

- **Use "Close-Enough" Data.** If samples are available that have response values *close* to the range of interest, the user may also include those samples when performing the predictive accuracy calculations. If the extra samples are close to the range of interest, this may be a very effective option. However, there is no guarantee that good predictive accuracy for samples close to the range of interest will correspond to good accuracy for samples *within* the range of interest: picture a zoologist learning about lions by studying housecats.

- **Get More Data.** The user may attempt to gather more results in hopes some of the new data will fall within the range of interest. This may be expensive or infeasible, depending on the time and effort that would be required.

If the user has the time and resources to obtain more data, the third option is preferable as it offers the greatest reduction in uncertainty. If separate test samples are used for accuracy evaluation, an optimization approach may be the best way to accumulate samples within the range of interest. On the other hand, if cross validation is used, any technique that identifies samples within the range of interest can be used, as those samples will benefit both the accuracy estimation process *and* the accuracy of the surrogate model itself. One such technique is described in Section A.7.

A.6.4 Stopping Criteria

There are multiple reasons for the user to decide that the surrogate model is "good enough" and end the process. The trivial reason is that all resources have been expended – there is no more time or budget left to run more analyses or train new surrogate models. This result probably would not be satisfying for anyone involved.

The literature mostly assumes that the user will know what "good enough" means for themselves for each application. Most of the stopping criteria that have been published are intended for use with optimizations, such as stopping when the expected improvement is less than some threshold or when no improvements have been made after some number of cycles. Fortunately, some guidelines for surrogate model accuracy for non-optimization purposes have been identified.[57]

As stated in the previous section, Root Mean Squared Error has the same units as the response being modeled. The calculated RMSE value can be normalized by the useful range of the response being modeled. This quantifies the prediction error relative to the range of the response. If the user is interested in the global behavior of the response, this range may be calculated using the largest and smallest observed response values. For example, if the response has a maximum value of 20 and a minimum value of 7, the RMSE would be normalized by a factor of 13. If, on the other hand, only a certain range of the response is of interest (for example, the user is only interested in pitching moment coefficients between -0.1 and 0.1), the RMSE would be normalized using this range of interest (0.2, in this example). Note that when normalizing by a range of interest, the RMSE should be calculated using *only* samples that fall within that range of interest.

This normalized RMSE value can give the user a rough estimate of the predictive accuracy of the surrogate. As a guideline, a model with normalized RMSE of less than 10% is considered to be "reasonable," while one with normalized RMSE of less than 2% is considered "very good."[57] These guidelines can help the user determine whether the current model is acceptable or if it will require additional investment, such as the acquisition of more training data. Such acquisition is the subject of the next step in this guide.

A.7 Selecting New Analyses Based On Previous Results

This step in the method, **Step Four**, assumes that the available surrogates are not accurate enough, and that there are sufficient resources available to perform new analyses to acquire more data. To be most effective, the analyses will be selected based on the results of previous analyses, allowing them to emphasize portions of the design space that are of interest to the user. This process of **selecting the most useful new analyses based on previous results** is known as "adaptive sampling."

Most of the research that has been done with adaptive sampling has done so in the context of optimization, i.e. maximizing or minimizing some objective function.[38, 83, 111, 173] The present technique is intended for problems where *a certain range of response values is of interest*, as opposed to maximizing or minimizing something, so a less-common approach to sample selection is necessary. The preferred approach in this case is contour-based sampling.[149]

Contour-based sampling evaluates potential new samples, known as "candidates," by estimating how each candidate would affect the predictive confidence of the surrogate model if it were added to the data pool. Specifically, the contour-based sampling algorithm attempts to identify the candidate which would produce *the largest reduction in prediction uncertainty for samples with responses within the specified ranges of interest.*

To quantify the reduction in prediction uncertainty, test samples are used. Unlike in model validation, the test samples do not need to be analyzed in contour-based sampling. Instead, the surrogate model is modified as if the candidate had been added to the training pool, and then the modified surrogate is used to estimate the prediction uncertainty at each test sample. These uncertainties are then combined in a weighted sum. If a test sample has a high likelihood of having a response value within the specified range of interest, it will be weighted heavily (on the order of 1); if the test sample is unlikely to fall within the range of interest, it will be weighted lightly (on the order of 0). The algorithm is based on Kriging, since that surrogate modeling approach allows the user to quickly and easily estimate the predictive confidence at any sample in the design space.

For this guide, it will be assumed that there are multiple responses of interest, and the

user wishes to accurately identify samples for which the value of every response falls within some user-defined range of interest. This section will use the symbol Q to refer to the total number of responses that are being taken into account when selecting new samples to analyze.

A.7.1 Generating Candidate & Test Samples

The first step in choosing a new sample with contour-based sampling is to *generate candidate samples*. This is done in the same way that the initial samples were chosen in Section A.4: a space-filling distribution is probably the simplest and most universal approach. The number of candidates generated is left up to the user. Evaluating more candidates means that the selection algorithm will have more options to choose from, but will increase the amount of computational effort required to choose a new sample. A good starting value may be $10d$ candidates, where d is the number of free parameters.

Next, *a set of test samples is generated.* These test samples will be used to evaluate the candidates based on the estimated response values and prediction uncertainties. Most likely, the test samples will also be evenly distributed throughout the design space. The more test samples that are used, the more accurately the algorithm will assess the candidates, but the longer the selection process will take. As a rough guide, $15d$ may be a good number of test samples, although the actual best value will depend strongly on the problem being investigated.

A.7.2 Filtering Candidates

If desired, the user may speed up the sample selection process by ignoring candidates that have a low likelihood of having responses within the specified ranges of interest. To accomplish this, the user must calculate the likelihood that each candidate falls within the range of interest for a given response. This is done by using the surrogate models to estimate the response value (y) and the prediction uncertainty (y_{MSE}) for each candidate.

The Kriging prediction uncertainty indicates, roughly, how close to the prediction is expected to be to the actual response value. This distribution is assumed to be a Gaussian distribution. As a result, the probability that the response is larger than some target value

may be calculated analytically. Using that relationship, the likelihood that the response (y) falls within certain limits (y_{lower} & y_{upper}) can be calculated as:

$$P\left(y_{upper} > y > y_{lower}\right) = P\left(y > y_{lower}\right) - P\left(y > y_{upper}\right) \tag{14}$$

$P\left(y > y_{lower}\right)$ is the likelihood that the predicted value y is larger than some specified threshold y_{lower}. Zelen & Severo[1] provide a method for calculating such likelihoods for a standard normal distribution. To use this method, the prediction from the Kriging model must be translated into a standard normal distribution.

The standard normal is a special case of the Gaussian, or normal, distribution for which μ, the mean, is equal to 0 and σ^2, the variance, is equal to 1. This is commonly written as $\mathcal{N}\left(0,1\right)$. Any normal distribution $\mathcal{N}\left(\mu,\sigma^2\right)$ can be transformed into a standard normal distribution using the following equation:

$$Z = \left(\frac{y_{interest} - \mu}{\sqrt{\sigma^2}}\right) \tag{15}$$

Here, $y_{interest}$ is the threshold value, such as y_{lower}; μ is the predicted response value from the surrogate model; and σ^2 is the prediction uncertainty.

To calculate the probability that $y_{interest}$ is larger than μ, Z is plugged into the equation:

$$\begin{gathered}\Phi(x) \approx 1 - \phi(Z)\left(b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 + b_5 t^5\right), \\ t = \frac{1}{1 + b_0 Z}\end{gathered} \tag{16}$$

Here, $\phi(Z)$ is the probability density function (PDF) of the standard normal distribution, which expresses the probability that a random draw from the distribution would produce a value of Z. The b coefficients each take a different value: $b_0 = 0.2316419$, $b_1 = 0.319381530$, $b_2 = -0.356563782$, $b_3 = 1.781477937$, $b_4 = -1.821255978$, and $b_5 = 1.330274429$. This approximation for $\Phi(x)$ is accurate to within $< 7.5 \times 10^{-8}$ as long as $y_{interest} > \mu$.

The PDF of a normal distribution, $\phi(Z)$, can be calculated by:

$$\phi(Z) = \frac{1}{\sqrt{2\pi(1)^2}} exp\left[-\frac{(Z-0)^2}{2(1)^2}\right] \tag{17}$$

Note that this relation for $\Phi(x)$ is only valid when $y_{interest}$ is greater than μ, the predicted response. If $y_{interest}$ is less than μ, this set of equations will give nonsense answers. In

these situations, the user should replace $y_{interest}$ with $-y_{interest}$ and μ with $-\mu$, which will calculate the probability that $y_{interest}$ is *less* than μ ($P(y_{interest} < \mu)$). This must then be converted back to $P(y_{interest} > \mu)$:

$$P(y_{interest} > \mu) = 1 - P(y_{interest} < \mu) \tag{18}$$

Using the predicted response value (y) and prediction uncertainty (y_{MSE}) for a candidate sample, the limits of the range of interest for the response (y_{lower} and y_{upper}), and Equations 14-18, the user can calculate the likelihood that the true response being emulated by the surrogate falls within the range of interest. If there are multiple responses, this process is repeated for every response, and the minimum value is retained as the likelihood score, referred to as the **"probability of interest"** or **POI**. This is then repeated for the rest of the candidates. The user may then choose not to analyze the candidates with too low of a likelihood score, saving time and computational effort.

This raises the question: what constitutes "too low" of a POI score? There is a natural impulse to demand high values, such as 75% or even 90%. This seems reasonable, but may be too restrictive: if the prediction uncertainty is large relative to the response range of interest, it is possible that *no* candidate will meet the requirement. Instead, the user should review the likelihood scores of the candidates and choose a requirement that makes sense in light of those scores.

Higher POI requirement values will eliminate more candidates, reducing the necessary computational effort to select a new sample. This may have the side-effect of eliminating the candidates with higher uncertainties, i.e. those farther away from existing samples, which could result in a very conservative approach that does not explore the design space much. Alternatively, lower requirement values allow more exploration of the design space, but increase the time and effort required to select each sample, resulting in a slower sample selection process. It is recommended that a POI requirement of at least 0.01-0.1% be used to screen out any candidates that are extremely unlikely to fall within the range of interest.

A.7.3 Analyzing a Candidate

The next step is to evaluate how much the predictive uncertainty of the surrogate model would be reduced if a particular candidate were analyzed. Ordinarily, this would require estimating the responses for the candidate sample, adding the sample to the training data set, and re-training the surrogate model to include the new sample. This can quickly become computationally expensive if more than a handful of candidate samples must be evaluated. For Kriging, the computational expense is primarily due to the need to invert a correlation matrix each time a surrogate is trained. However, most of the training data set will be unchanged; only the candidate sample will vary. This simplifies the problem significantly, as will be demonstrated shortly.

The calculations and example code given in this section are for a single-fidelity model, not the Ghoreyshi cokriging model that was described in Section A.5.3. This is because the choice of multi-fidelity approach will depend on the problem being addressed; any multi-fidelity approach could be substituted into this sample selection algorithm.

Some of the terms in the calculations depend only on the current surrogate model, not on any of the candidates. These terms can be calculated before any candidates are evaluated:

```
Cinv = cell(1,Q);
F = cell(1,Q);
for q = 1:Q
  daceCholesky = full(dmodel{q}.C);
  daceCinv = inv(daceCholesky);
  Cinv{q} = daceCinv' * daceCinv;
  F{q} = full(dmodel{q}.C) * dmodel{q}.Ft;
  %  From DACE manual, equation 3.10
end
IMSE = zeros(number_of_candidates, Q);
```

Note that there is a change in notation that occurs in those lines of code: the DACE model uses $dmodel\{q\}.C$ to refer to the Cholesky factorization of the correlation matrix, R, for the

q^{th} surrogate in *dmodel*. However, $Cinv\{q\}$ is the inverse of the correlation matrix, *not* the inverse of the Cholesky factorization. This change was done to align with the notation used by Picheny et al.[149]

$IMSE$ is a matrix that will contain the weighted uncertainty values that are calculated for each candidate. Q is the number of responses that are being modeled; the meaning of $number\_of\_candidates$ should be clear.

All operations after this point must be repeated for each candidate sample – or at least, each candidate sample that meets or exceeds the *probability of interest* requirement described in Section A.7.2, if such requirements were set. The variable j will be used as the index denoting a particular candidate sample from the set of options.

Because DACE normalizes the data when creating a Kriging model, the j^{th} candidate sample must be normalized in the same manner:

```
[n,o] = size(S);
mS = mean(S);
sS = std(S);
Snorm = (S - repmat(mS,n,1)) ./ repmat(sS,n,1);
candidate_norm = (candidateSet(j,:) -mS) ./ sS;
Snorm_can = [candidate_norm; Snorm];
```

Here, n is the number of training samples that are available, and o is the number of free parameters included in the model.

Next, the normalized distances from the candidate sample to the existing training data are calculated, and those results are used to determine c_{new}, the correlations between the candidate and the training data:

```
newS = Snorm;
D_temp = repmaat(candidate_norm,n,1) - newS(1:n,:);
c_new = feval(correlationModel, dmodel{q}.theta, D_temp);
```

Here, *correlationModel* indicates the correlation model that was used to build the Kriging surrogate, such as Gausssian ("@corrgauss"), exponential ("@correxp"), etc. The correlation

model names correspond to the correlation models included with the DACE toolbox.

To evaluate how the candidate would affect the prediction confidence, the new correlation matrix (which includes the candidate sample) must be calculated and inverted. Rather than using brute force, Schur's complement[201] calculates the new inverted correlation matrix (C_{n+1}^{-1}) in terms of inverted correlation matrix *without* the candidate sample (C_n^{-1}).

$$C_{n+1}^{-1} = \begin{bmatrix} 1 & 0 \\ -C_n^{-1}c_{new} & I_n \end{bmatrix} \begin{bmatrix} \frac{1}{\sigma^2 - c_{new}^T C_n^{-1} c_{new}} & 0 \\ 0 & C_n^{-1} \end{bmatrix} \begin{bmatrix} 1 & -c_{new}^T C_n^{-1} \\ 0 & I_n \end{bmatrix} \tag{19}$$

Here, I_n is the $n \times n$ identity matrix, and σ^2 is the estimated process variance of the surrogate model. 0 and 1 represent blocks of all zeros and all ones, respectively. c_{new} is a $n \times 1$ vector containing the correlation between the candidate sample and each existing training data samples.

The augmented F matrix must be calculated as well:

```
f_candidate = feval(dmodel{q}.regr, candidate_norm);
F_nplusone = [f_candidate; F{q}];
to_be_inverted_term = (F_nplusone' * C_nplusone_inv * F_nplusone);
```

The $dmodel\{q\}.regr$ term indicates the underlying trend model that was used for the Kriging surrogate $dmodel\{q\}$. These trend models, such as "@regpoly1," are included with DACE.

The rest of the calculations depend on the test samples as well as the candidate sample. The variable k will be used as an index to denote individual test samples. First, each test sample must be normalized, just like the training data and the candidates:

```
test_norm = (testSet(k,:) - mS) ./ sS;
```

Then, using the normalized test sample, the distances to the training data (including the current candidate) and the correlation between the test sample and the training data are calculated:

```
D_test = repmat(test_norm, n+1, 1) - Snorm_can(1:n+1,:);
c_test = feval(correlationModel, dmodel{q}.theta, D_test);
```

Next, the f value for the test sample is calculated, which is the last term required to calculate the prediction uncertainty for that sample:

```
f_test = feval(dmodel{q}.regr, test_norm);
Res_MSEs(k,q) = dmodel{q}.sigma2 * (1 - c_test' * C_nplusone_inv * ...
   c_test + (f_test - c_test' * C_nplusone_inv * F_nplusone) * ...
   (to_be_inverted_term \ (f_test - c_test' * C_nplusone_inv * ...
   F_nplusone)' ) );
```

Once the new prediction uncertainty has been estimated, the weighting value for that test sample can be evaluated. Like the probability of interest calculations, the weighting function depends on the cumulative distribution function of the normal distribution (Equation 16), so first some logic is introduced to make sure that the predicted response value for the test sample (μ) is less than the threshold of interest (y_{lower} or y_{upper}):

```
predictedValue1 = predictor(testSet(k,:),dmodel{q});
predictedValue2 = predictedValue1;
rev1 = 0;
if predictedValue1 > y_upper
   predictedValue1 = predictedValue1 - 2*(predictedValue1 - y_upper);
   rev1 = 1;
end
rev2 = 0;
if predictedValue2 > y_lower
   predictedValue2 = predictedValue2 - 2*(predictedValue2 - y_lower);
   rev2 = 1;
end
```

After making sure that the requirements for Equation 19 have been satisfied, $P(y > y_{lower})$ and $P(y > y_{upper})$ can be calculated:

```
Znorm1 = (y_upper-predictedValue1) / sqrt(Res_MSEs(k,q)+0.0000000001);
```

```
t1 = 1 / (1 + b0 * Znorm1);
standard_normal1 = (1/sqrt(2*pi*1^2)) * exp( -((Znorm1-0)^2) / (2*1^2));
prob1 = 1 - standard_normal1*( b1*t1 + b2*t1^2 + b3*t1^3 + b4*t1^4 ...
  + b5*t1^5);
if rev1 == 1
  prob1 = 1 - prob1;
end

Znorm2 = (y_upper-predictedValue2) / sqrt(Res_MSEs(k,q)+0.0000000001);
t2 = 1 / (1 + b0 * Znorm2);
standard_normal2 = (1/sqrt(2*pi*1^2)) * exp( -((Znorm2-0)^2) / (2*1^2));
prob2 = 1 - standard_normal2*( b1*t2 + b2*t2^2 + b3*t2^3 + b4*t2^4 ...
  + b5*t2^5);
if rev2 == 1
  prob2 = 1 - prob2;
end
```

Note that some division operations include an extra factor of 10^{-10}. These additions were made to account for situations where the uncertainty at a test sample is close to zero, which may be the case if the test sample is very close to a sample in the training set. By adding this factor of 10^{-10}, the division operation avoids any divide-by-zero errors.

The weighting function is equal to $P(y_{upper} > y > y_{lower})$, which can be transformed using Equation 18 into:

```
W(k,q) = (prob1-prob2);
```

Lastly, the contribution of this test sample to the overall weighted uncertainty for the current (j^{th}) candidate is calculated as a weighted sum:

```
IMSE(j,q) = IMSE(j,q) + W(k,q) * Res_MSEs(k,q);
```

This sequence is repeated for each test sample. Then, the $(j+1)^{th}$ candidate is evaluated in the same manner. This continues until all candidates have been evaluated. If necessary,

the algorithm then moves on to the next response (from q to $q+1$) and the analysis begins again.

If the sample-selection algorithm is evaluating only one response, the best sample can be selected at this point. The sample that will most reduce predictive uncertainty in the response range of interest is the one with the lowest IMSE value.

If more than one response has a specified range of interest, it is likely that the IMSE values for each response will have very different magnitudes. Unless this is addressed, the response with the largest IMSE values will dominate the sample selection process. To account for this factor, the IMSE values for each response are normalized to have a mean value of 0 and a standard deviation of 1:

```
avgIMSE = ones(Q,1);
stdIMSE = ones(Q,1);
norm_IMSE = zeros(size(IMSE));
for q = 1:Q
  avgIMSE(q) = mean(IMSE(:,q));
  stdIMSE(q) = std(IMSE(:,q));
  norm_IMSE(:,q) = ( IMSE(:,q)-avgIMSE(q) ) / stdIMSE(q);
end
```

The normalized IMSE values for each candidate are averaged to quantify the overall effect that is expected if that candidate were to be analyzed. The results are then sorted and the most effective sample is identified:

```
net_IMSE = zeros(number_of_candidates,1);
for j = 1:number_of_candidates
  net_IMSE(j) = mean(norm_IMSE(j,:));
end

results = sortrows([net_IMSE candidateSet]);
new_sample = results(1, 2:number_of_dimensions+1);
```

The sample that is selected, *new_sample*, is the candidate that is expected to be the most useful new sample to analyze.

This algorithm is not infallible. It can only select one of the candidates that are presented to it; it does not choose the best sample possible within the design space. Additionally, the algorithm assumes that the current surrogate model is fairly accurate with respect to the behavior of the response. If the surrogate model is poor, the candidates will not be evaluated accurately, and it is unlikely that the best candidate will be selected.

Once the new set of inputs has been selected, it can be analyzed to determine the true response values. Once the response values are available, an updated surrogate model should be trained and evaluated – essentially, returning to **Step Two** of this guide. The process continues until the surrogate models are deemed acceptable or the available resources run out, as described in Section A.6.4.

A.7.3.1 Non-Sequential Analyses

The strategy of analyzing each new sample as soon as it is selected may not be the most efficient one. This is especially true if some or all of the analysis (including both the setup and the analysis) can be done in parallel. In such a scenario, it may be more efficient to select a batch of samples and analyze them all at once before updating the surrogate model.

Selecting multiple samples introduces the risk that the second sample in a batch might be very close to the first, reducing its usefulness. To avoid this, each sample is added to the training data pool (using the response values estimated by the current surrogate models) before subsequent samples are selected. This has the effect of reducing the prediction uncertainty around the "new" sample, diminishing the incentive to place later samples in the same region of the design space.

Because estimated response values are used rather than true response values, the later sets of input values in the batch will be chosen based on information that is not entirely up-to-date. This may lead to sub-optimal selections in some cases. Still, if the analysis time is significant and multiple samples can be analyzed in parallel, the overall execution time may be substantially reduced by this approach.

To select multiple cases without executing the analysis, the user should run the sample selection algorithm repeatedly, adding the selected sample to the model after each round. The augmented data set is then used when selecting the next sample. By taking this approach, subsequent samples in each batch will naturally spread out, with no risk of clumping or clustering.

After a batch of samples is selected, they can then be analyzed all at once, taking advantage of any opportunities to perform the analyses in parallel. This approach can significantly reduce the total time to select and analyze a given number of samples, while reducing the negative consequences of not analyzing each sample immediately.

APPENDIX B

GEOMETRIC PARAMETER RANGES AND DEFAULT VALUES

A full description of the geometric parameters which define the outer shape of a reusable booster vehicle may be found in the work of Garmendia et al.[61] The details should not affect the validity of the methods described herein, and thus were not reproduced within this document.

Two parameters, Fuselage Radius Fraction and Wing Root Chord Fraction, were independent variables in every experiment and thus default values were not assigned for these parameters. Additionally, some of the experiments featured 9 independent parameters rather than 2. For these experiments, the variables Top Curvature 1, Top Curvature 2, Nose Droop, Nose Fineness Ratio, Wing Half–Span Fraction, Wing Airfoil Camber, and Area Ratio of Vertical Tail to Wing were allowed to vary alongside Fuselage Radius Fraction and Wing Root Chord Fraction. These variables are **bolded** and marked with the symbol [1].

When selecting default values for the parameters that would be inactive in the smaller–scale experiments, the objective was to identify values that would produce consistent and/or interesting response behavior for the purposes of the experiments. In some cases, the default value selected was outside the nominal range for that parameter. This was considered to be acceptable because these default values were used as part of smaller experiments but never as part of the larger–scale experiments, and thus the pre–existing space–filling data sets which *were* constrained to those limits were still usable as the "null hypothesis" for the larger–scale efforts. Parameters for which defaults lie outside the nominal range are marked with the symbol [2].

Parameter	**Default Value**	**Min. Value**	**Max. Value**	**Units**
Scale	50	40	100	Feet
LoftStart	0.6	0.40	0.95	Fraction of Scale
LoftEnd	0.15	0.05	0.20	Fraction of Scale
Fuselage Radius Fraction	–	0.05	0.25	Fraction of Scale
Flare Factor	1.07	1.00	1.15	Fraction of Fuselage Radius
Top Curvature 1[1]	**0.1**	**0**	**1**	Unitless
Top Curvature 2[1]	**0.6**	**0**	**1**	Unitless
Bottom Curvature 1	0.1	0	1	Unitless
Bottom Curvature 2	0.6	0	1	Unitless
Nose Droop[1]	**0.57**	**0.5**	**1**	Fraction of Fuselage Height

Parameter	Default Value	Min. Value	Max. Value	Units
Nose Spatularity	0.15	0	1	Unitless
Nose Fineness Ratio[1]	**1.7**	**1**	**3**	Unitless
Inboard Wing Sweep Angle[2]	75	40	70	Degrees
Outboard Wing Sweep Angle	45	10	60	Degrees
Wing Root Chord Fraction	–	0.3	0.7	Fraction of Scale
Inboard Taper Ratio	0.6	0.5	0.7	Unitless
Outboard Taper Ratio	0.35	0.25	0.95	Unitless
Wing Half–Span Fraction[1]	**0.34**	**0.1**	**1.5**	Fraction of Scale

Parameter	**Default Value**	**Min. Value**	**Max. Value**	**Units**
Wing Crank Location	0.17	0.1	0.5	Fraction of Wing Half–Span
Wing Tip Twist Angle	0	-5	0	Degrees
Wing Incidence Angle	0	0	3	Degrees
Wing Dihedral Angle	0	0	12	Degrees
Wing Airfoil Camber[1]	**0.07**	**0**	**6**	Fraction of Local Chord (in 10% Increments)
Wing Airfoil Position of Maximum Camber	5	2	6	Fraction of Local Chord (in 10% Increments)
Wing T/C Ratio	7	3	8	Fraction of Local Chord (in 10% Increments)

Parameter	**Default Value**	**Min. Value**	**Max. Value**	**Units**
Wing Location of Max. Thickness	3	2	8	Fraction of Local Chord (in 10% Increments)
Leading Edge Radius Factor	6	2	8	Unitless
Area Ratio of Vertical Tail to Wing[1]	**0.4**	**0.1**	**0.5**	Unitless
Vertical Tail Cant Angle	10	0	30	Degrees
Vertical Tail Toe–In Angle	5	0	10	Degrees
Vertical Tail Leading Edge Sweep Angle	45	20	55	Degrees
Vertical Tail Aspect Ratio	1.25	0.5	2.0	Unitless

Parameter	Default Value	Min. Value	Max. Value	Units
Vertical Tail Taper Ratio[2]	0.4	0.5	0.8	Unitless
Vertical Tail Airfoil Camber	0	0	6	Fraction of Local Chord
Vertical Tail Position of Maximum Camber[2]	0	2	6	Fraction of Local Chord (in 10% Increments)
Vertical Tail T/C Ratio	7	3	8	Unitless
Vertical Tail Location of Max. Thickness	3	2	8	Fraction of Local Chord (in 10% Increments)
Inboard Elevon Depth	30	10	40	Unitless
Outboard Elevon Depth	30	10	40	Unitless
Rudder Depth	30	10	40	Unitless

Parameter	**Default Value**	**Min. Value**	**Max. Value**	**Units**
Body Flap Size	0.10	0.08	0.015	Fraction of Scale
Inboard Elevon Deflection (Starboard)	0	-30	30	Degrees
Outboard Elevon Deflection (Starboard)	0	-30	30	Degrees
Inboard Elevon Deflection (Port)	0	-03	30	Degrees
Outboard Elevon Deflection (Port)	0	-30	30	Degrees
Rudder Deflection (Starboard)	0	-30	30	Degrees
Rudder Deflection (Port)	0	-30	30	Degrees

Parameter	**Default Value**	**Min. Value**	**Max. Value**	**Units**
Body Flap Deflection	0	-20	30	Degrees
Wing Longitudinal Position	-0.1	-0.15	0	Fraction of Scale

APPENDIX C

ADDITIONAL RESULTS FROM "PROBABILITY OF INTEREST" STUDIES

This appendix presents the results from the experiment in Section 4.9 in greater depth. This experiment featured 2 free parameters, the Fuselage Radius Fraction and the Wing Root Chord Fraction, and 3 responses. The responses were the pitching moment coefficient of the vehicle at three flight conditions: Mach 0.3, α 15^o, β 0^o; Mach 0.8, α 0^o, β 0^o; and Mach 2.5, α 0^o, β 0^o. Default settings for the remaining geometric parameters are recorded in Appendix B.

The contour-based sampling algorithm, which drew only on results from Cart3D for this experiment, was tasked with reducing prediction variance for cases likely to have pitching moment coefficients within $\pm$ 0.1 at every flight condition. A 50 $\times$ 50 grid of cases spanning the design space was analyzed with Cart3D. Based on the results of those analyses, the cases which met the pitching moment criterion at each flight condition can be seen in Figures 65a, 65b & 65c. The cases which met the pitching moment criterion at *every* flight condition are shown in Figure 65d. These cases will be henceforth referred to as the "cases of interest."

The analysis results were interpolated so that sampling experiments could be conducted without having to analyze each selected case as it is requested. This allowed experiments to be conducted quite rapidly, although the time required for the contour-based sampling algorithm to select the next sample was not insignificant. The algorithm used a 23×23 grid of candidates and a 40×40 grid of test points. The same candidates and test points were used in every round of sampling.

The contour-based sampling algorithm for one response as described by Picheny et al.[149] was extended to identify samples that would improve prediction confidence for multiple responses at once. When this extended algorithm was applied to the two-input, three-response problem described above, it was found that the algorithm would select samples

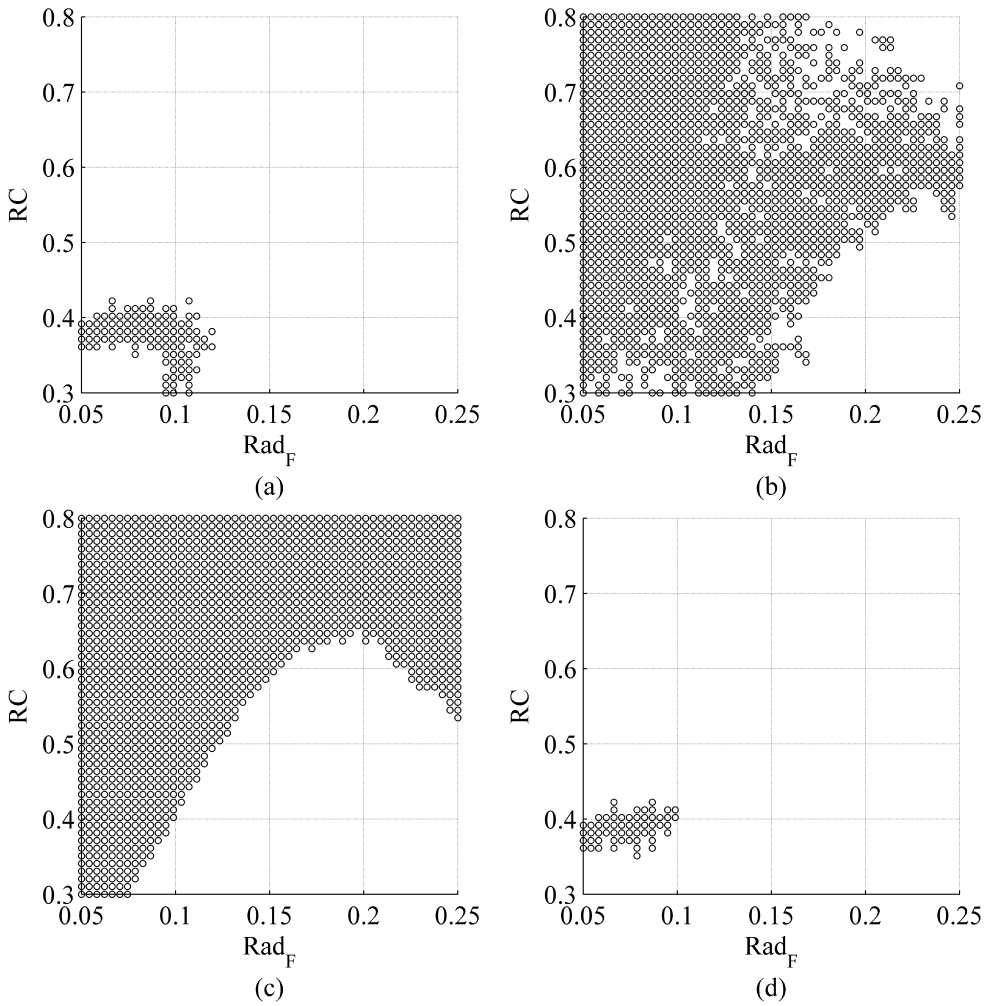

Figure 65: Cases of Interest at All Flight Conditions

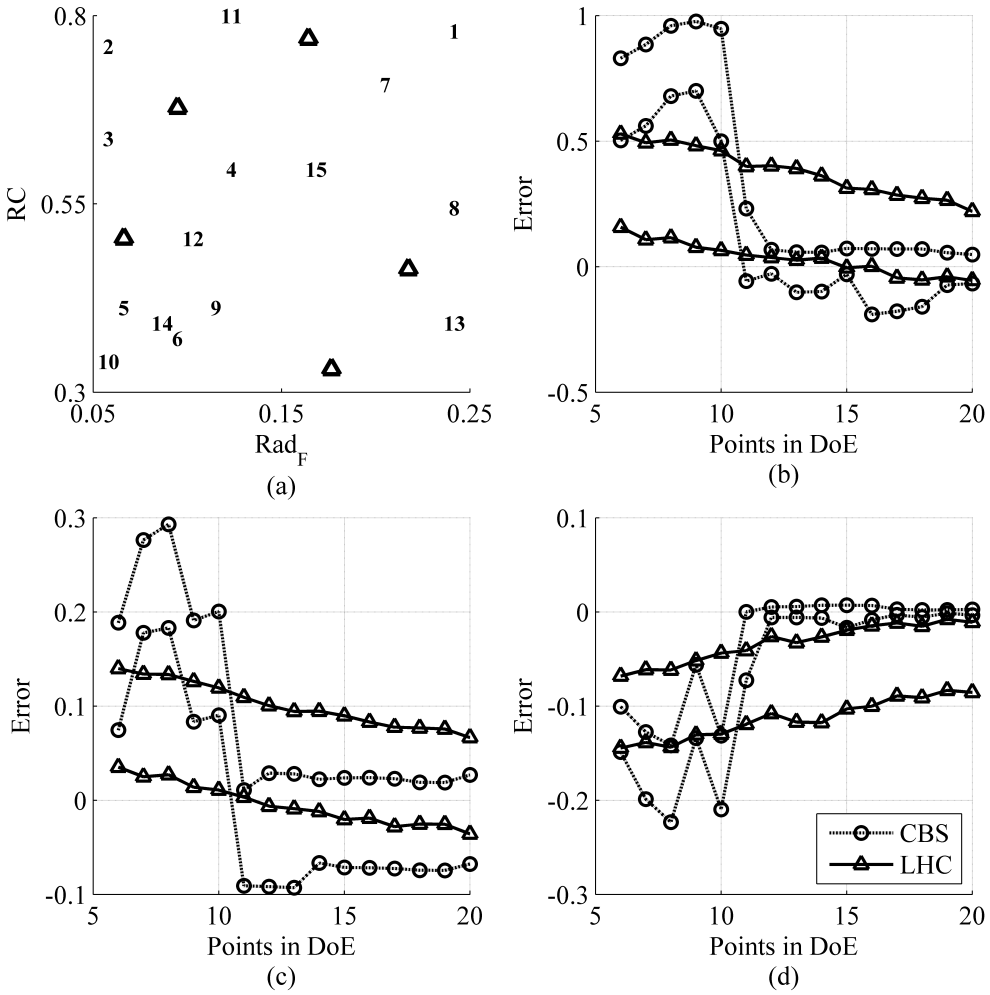

Figure 66: Sampling For Required POI of 0% (a) & 95% Prediction Error Quantiles for Cases of Interest at (b) Mach 0.3, (c) Mach 0.8, and (d) Mach 2.5

that improved prediction confidence for *any* response, even if the selected sample did not benefit *every* response. The samples selected, and the resulting prediction accuracy for the cases of interest, may be seen in Figure 66.

The algorithm was modified to emphasize regions that were likely to contain cases of interest (e.g. cases for which all responses lie within the range of interest) based on the expected Probability of Interest, or POI. This quantity captures the likelihood that every response for a particular sample would fall within the specified range(s) of interest. See Section 4.9 for more information on this topic. After modification, the algorithm would not

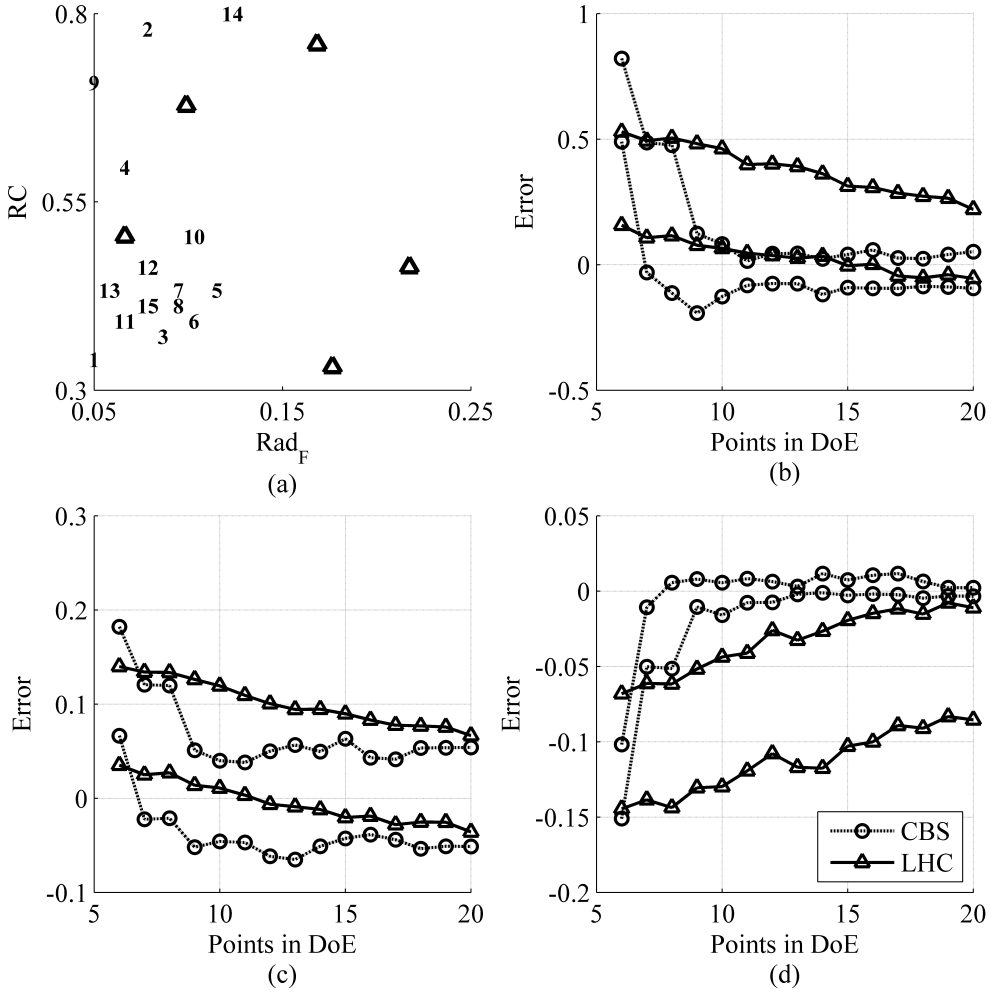

Figure 67: Sampling For Required POI of 1% (a) & 95% Prediction Error Quantiles for Cases of Interest at (b) Mach 0.3, (c) Mach 0.8, and (d) Mach 2.5

evaluate candidates which had POI values lower than the user-defined requirement. If no candidates met the requirement, the algorithm would select the candidate with the highest POI value.

A minimum POI requirement of 1% – that is, only candidates with better than a 1% likelihood of falling in the region of interest for all three responses – produced the sample distribution seen in Figure 67a. The samples exhibited significantly more clustering in the region of interest, which was the desired effect. Additionally, the 95% error quantiles converged more quickly than in the previous case.

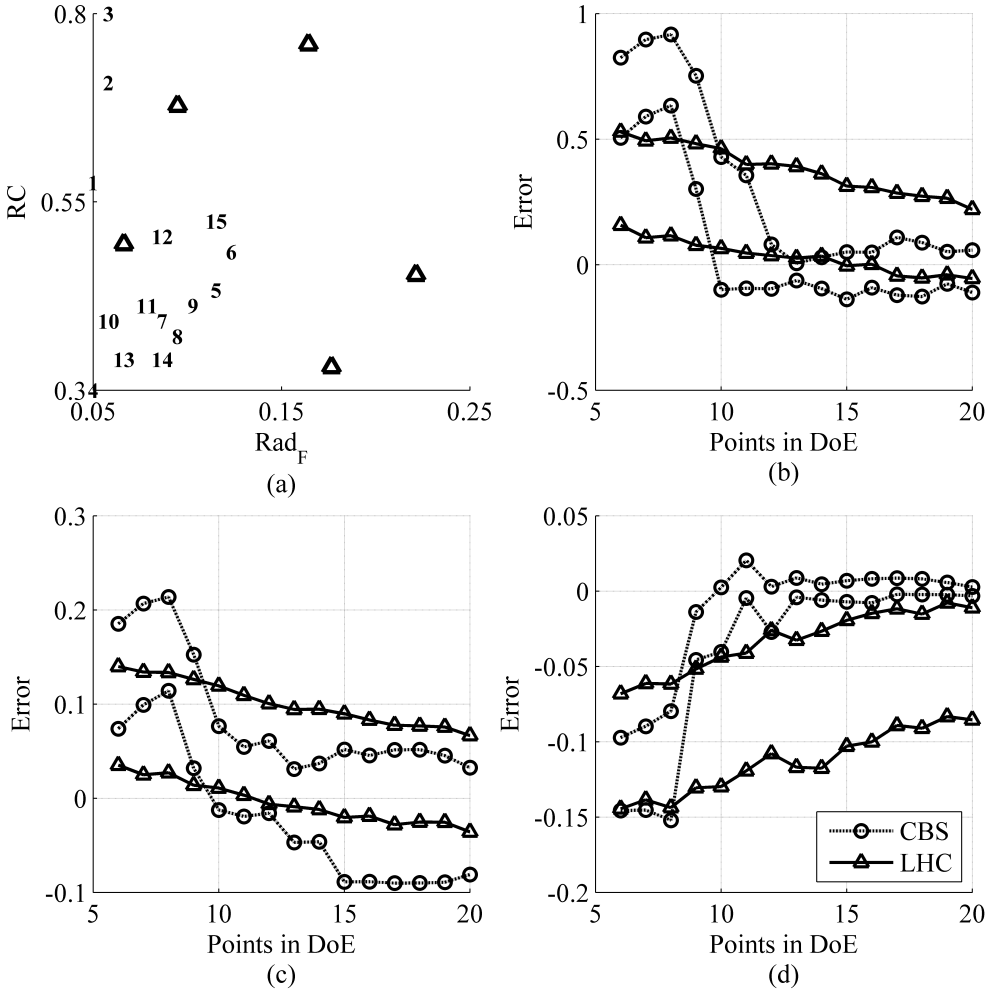

Figure 68: Sampling For Required POI of 5% (a) & 95% Prediction Error Quantiles for Cases of Interest at (b) Mach 0.3, (c) Mach 0.8, and (d) Mach 2.5

Increasing the required POI value to 5% produced the results shown in Figure 68. Importantly, in this scenario the algorithm did not identify the region of interest as quickly as before: the third sample for 1% POI was in or near the region of interest, while at 5% POI the region was not identified until the fifth sample. The prediction accuracy results for cases of interest also showed convergence behavior that was delayed relative to the 1% case, although it was still an improvement compared to the 0% case.

Figure 69 shows the effects of a 10% POI requirement. The samples were even more tightly clustered than in previous scenarios. Of particular importance was the order of

samples: note that the first 5 samples gradually progressed down the left edge of the space. The POI requirement restricted the candidates which the algorithm could consider, allowing only those with higher likelihoods to be sampled.

It should be noted that a low POI value did not necessarily indicate that a given candidate was expected to be far from the region of interest. The POI was estimated using the cumulative distribution function of a normal distribution; even if the predicted response was exactly centered in the region of interest, the POI might still be low if the prediction uncertainty was large relative to the integration limits. Responses that were poorly represented by a linear trend model would have high estimated process variance, and in turn would have high estimated prediction uncertainty. See Section 4.10.9.3 for more details on this topic.

With respect to prediction accuracy, the 10% POI results showed slightly delayed convergence relative to the previous scenarios, although the convergence for the Mach 0.3 response was significantly more smooth than for other scenarios.

A 15% POI requirement produced the results shown in Figure 70. The effect of POI requirements on sample selection was even more obvious here than it was for the case of 10% POI. Because the prediction uncertainty was small in the region around each original sample (represented by triangular icons), the algorithm selected cases close to those samples. This process repeated as the samples march through the space in search of desirable response behavior. After the seventh sample, there was enough confidence in the estimated model behavior to make the jump to the region of interest, which was correctly identified. Subsequent samples remained close to this region for the most part.

Once again the prediction error quantiles converged more slowly than in previous scenarios. The final 95% quantiles for the Mach 0.8 response were actually slightly wider than for the 10% scenario, indicating that after 15 samples the model produced by the 15% POI requirement was slightly less accurate for that response.

Finally, the experiment was repeated with a POI requirement of 25%. The effects of a high POI value are clearly visible in the results, displayed in Figure 71. The sample progression showed tight clustering with occasional jumps; although the left side of the region of interest was sampled by the eighth sample, the extent of that region was not

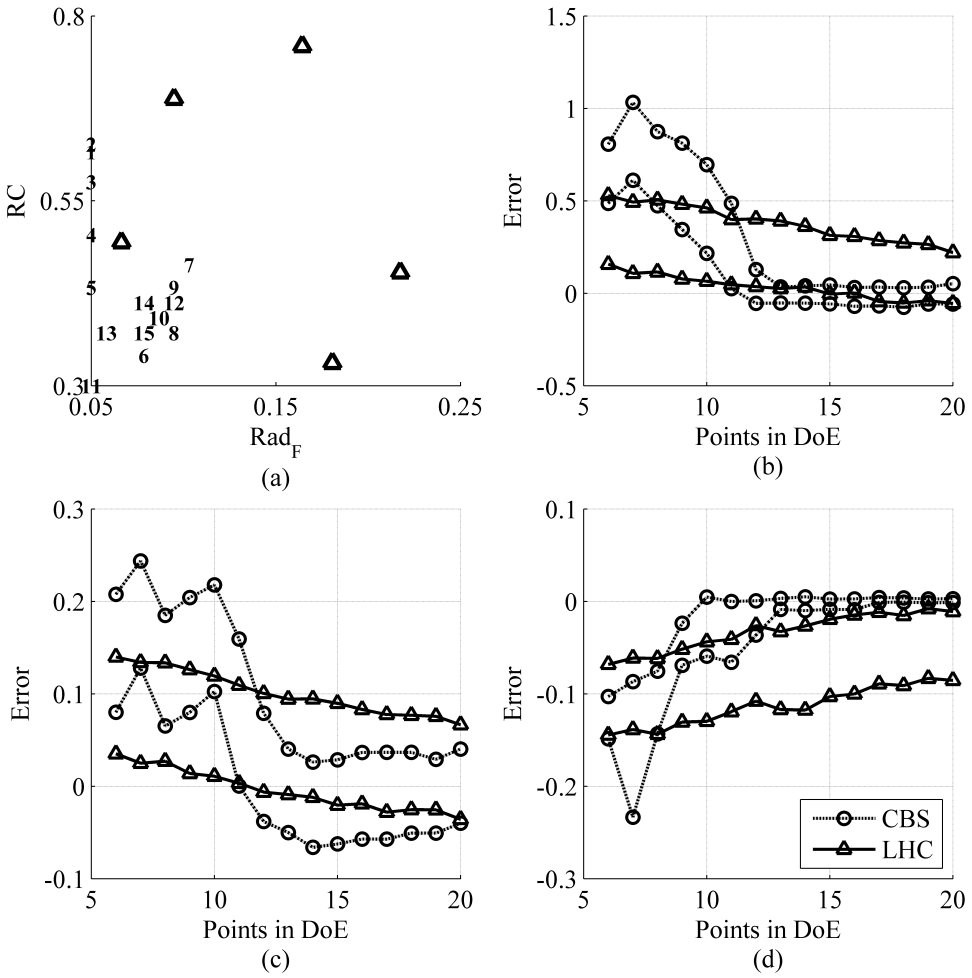

Figure 69: Sampling For Required POI of 10% (a) & 95% Prediction Error Quantiles for Cases of Interest at (b) Mach 0.3, (c) Mach 0.8, and (d) Mach 2.5

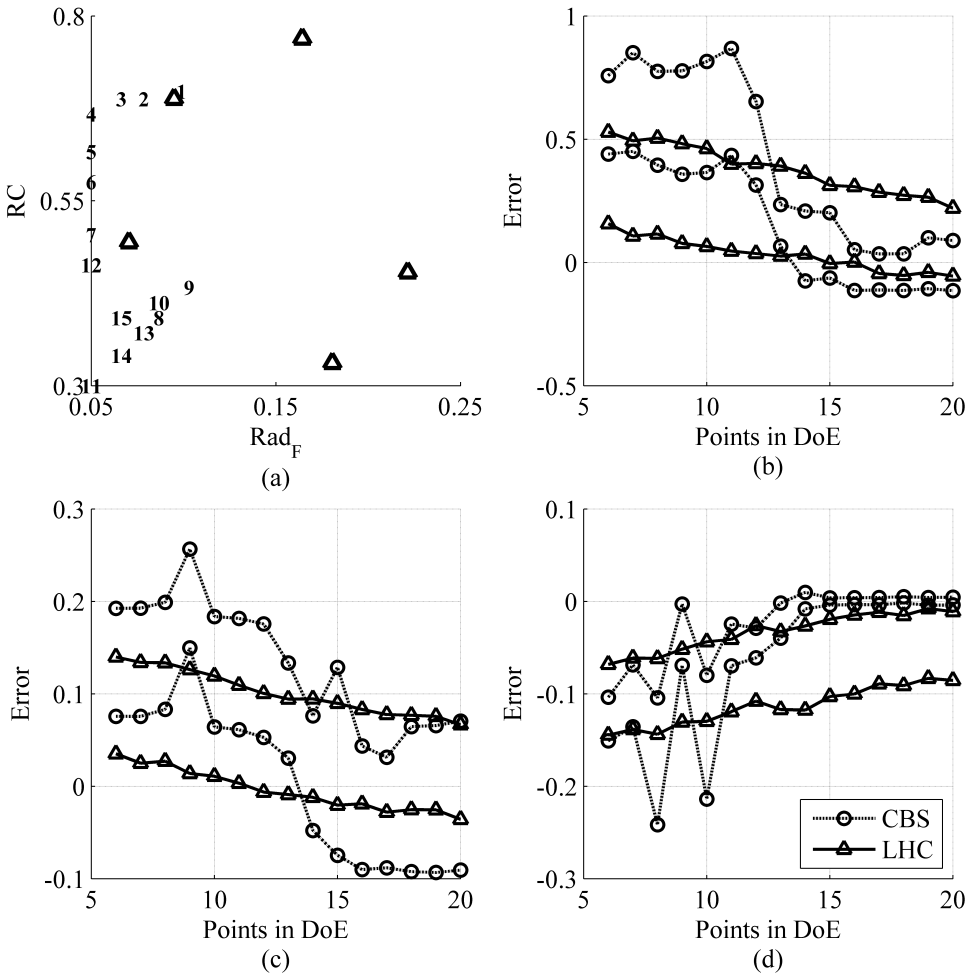

Figure 70: Sampling For Required POI of 15% (a) & 95% Prediction Error Quantiles for Cases of Interest at (b) Mach 0.3, (c) Mach 0.8, and (d) Mach 2.5

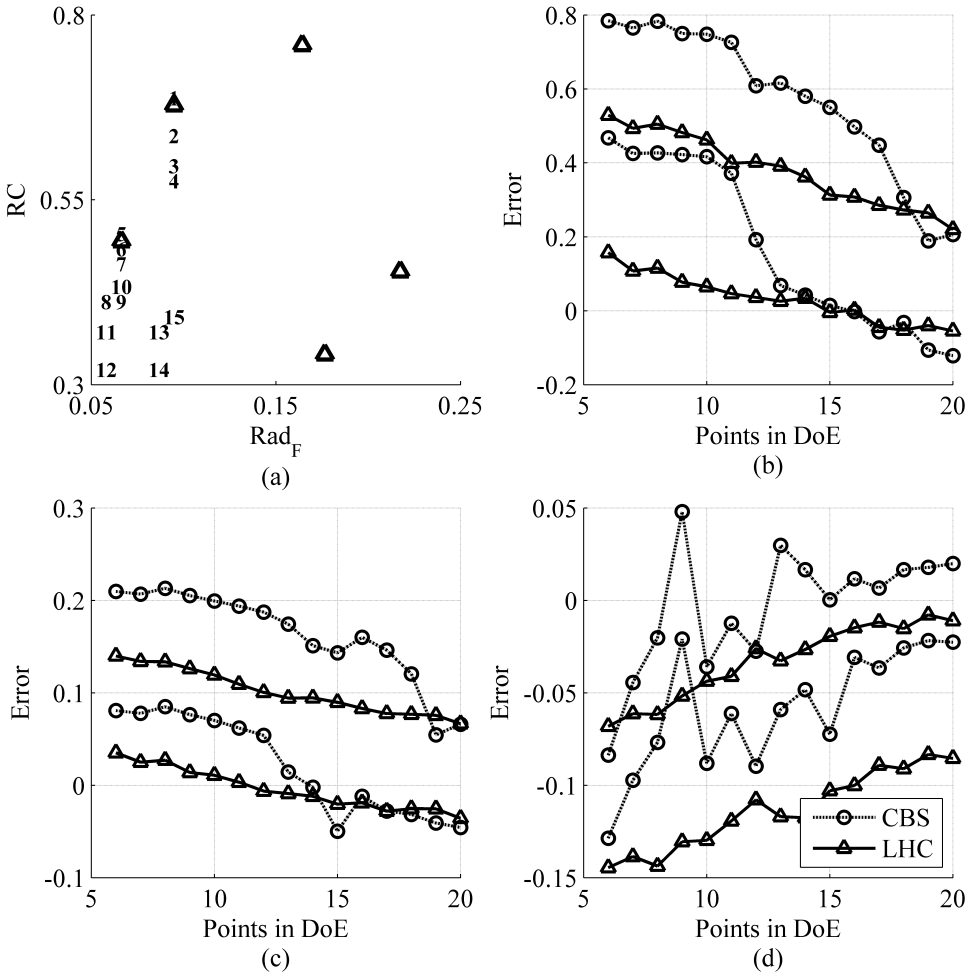

Figure 71: Sampling For Required POI of 25% (a) & 95% Prediction Error Quantiles for Cases of Interest at (b) Mach 0.3, (c) Mach 0.8, and (d) Mach 2.5

discovered until the thirteenth sample.

The slow exploration process that resulted from the high POI requirement is evident in the convergence history of prediction error for each response, as seen in Figures 71b, 71c & 71d. After 15 adaptive samples, contour-based sampling out-performed space-filling sampling only for the Mach 2.5 response.

The results in this Appendix should serve to demonstrate the effect of the POI requirement. Experiments revealed that even an ostensibly reasonable POI requirement such as 15% could handicap the sampling algorithm compared to a more lenient setting. This was due to the way that POI was calculated: candidates which are expected to have desirable

response values might exhibit low POI values if the prediction variance was large. Thus, particularly in the early stages of sampling, it may be best to use a relatively low POI requirement to allow the sampling algorithm to explore and gain a better understanding of response behavior. This requirement could be raised later on in the sampling process when there was more confidence that each response can be approximated with reasonable accuracy.

APPENDIX D

DESCRIPTION OF SCRIPTS FOR OPERATION OF CART3D AND COMPUTING RESOURCES

This appendix will provide details about the way that Cart3D and the High Performance Computing systems were run.

D.1 Preprocessing

This research effort focused on the aerodynamics of reusable booster vehicles during the unpowered return-to-launch-site phase of operation, and in particular the static moments. The aerodynamics were calculated based on the flight condition and the outer geometric shape, or Outer Mold Line (OML), of the vehicle. This shape, including control surfaced deflections, could be defined via a set of parameters using the PaceLab vehicle definition tool.[61, 142] For the purposes of this research, the vehicle definition tool was only used to generate a triangular surface mesh, which could then be analyzed using Cart3D, and a text file which identified the hinge axis of each control surface.

D.1.1 Generation of Surface Meshes

The PaceLab tool generated the wing and fuselage as a halfbody, as well as each control surface in its deflected state. The wing and half-fuselage were converted from .STL format to .TRI format using the **stl2tri.pl** and **off2tri.pl** utilities (packaged with Cart3D) and ADMesh, a utility which processes solid triangular meshes.[2] The two TRI files were then intersected using Cart3D's **intersect** utility to produce a single, watertight triangular surface mesh for half the vehicle. Almost invariably, any failures at this stage were found to be problems with the STL files. Most commonly, the STL file would not describe a closed surface (commonly referred to as "watertight") and when ADMesh was run, that program would attempt to create a closed surface if possible. The resulting triangulation could be of poor quality – i.e., very long thin triangles – and still might not produce a closed surface.

The **cubes** utility, which generates a volumetric grid around the shape, was used to test whether the triangulation for the wing & half-fuselage were of sufficient quality for Cart3D analysis. This was done by generating a volumetric mesh of roughly the resolution that would be used in later analyses, with approximately 2–3 million cells. Any deficiency in the surface triangulation would produce an error message and the triangulation would be considered to have failed. If the triangulation passed this test, it would then be mirrored about its centerline to create a full fuselage and wing using a Matlab-based utility created by Jonathan Sharma. The seven control surfaces would then be intersected with the fuselage and wing, and **cubes** would once again be used to test for gaps or other deficiencies in the mesh.

When a particular case failed to build properly, the root of the problem was sometimes to be found in the wing control surfaces. The PaceLab tool "cuts" control surfaces from the fixed wing using the nearest available surface nodes; a wing mesh with higher resolution was more likely to result in a viable TRI file, but this also produced relatively large files. The Matlab script which drove the geometry generation process was written so that the wing mesh resolution was started relatively low — *ncs*, the number of chordwise cross-sections between the wing root and the wing tip was usually set to 130 and *pcs*, the number of vertices per cross-section, was often set to 140 — and then, if any cases failed to build, these two parameters were increased by 20 and the process repeated. If *ncs* exceeded 290, any cases which had not been built successfully were abandoned.

D.1.2 Defining Cart3D and HPCC Settings

Aside from the surface mesh, a number of other files had to be generated before each analysis could be done. A number of scripts were created which automated the generation these other files.

D.1.2.1 Master Job Description File

A centralized data file was created which encapsulated all of the data required to set up a case. This master data file included:

- Batch name and case number;

- Flight condition ID number;
- Mach number, angle of attack, and sideslip angle;
- Orientation of vehicle within local coordinate frame;
- Reference area, mean aerodynamic chord, and wing half–span for normalization of forces & moments into coefficients;
- Center of mass position in the local coordinate frame, based on mass-estimating relationships developed by Lockheed Martin Space Systems;
- Number of nodes on the HPCC system to be requested for each case;
- Length of time these nodes should be reserved;
- Cart3D baseline CFL setting; and
- Definition of control surface hinge lines, for the calculation of hinge moments.

Most of these items are straight-forward, but some may deserve further explanation.

The **batch name** and **case number** uniquely identified each case and connected it to the input settings used to generate the surface mesh.

The **flight condition ID number** was a two-digit value which corresponded to particular values of Mach number, angle of attack and sideslip angle.

The **local orientation of the vehicle** told Cart3D how the vehicle was oriented with respect to its internal numerical conventions. The surface mesh generated by PaceLab had its origin at the lower corner of the fuselage rear surface, where the engine assembly or assemblies would attach, and the nose extended in the negative-X direction. The right wing of the vehicle was in the positive-Y direction, and the upper surface of the vehicle was in the positive-Z direction. The native orientation of Cart3D is to have the nose of the vehicle be the most positive X-coordinate, and the top of the vehicle be in the negative-Z direction, so any deviation from that orientation must be noted or else the condition simulated may not be the one intended. In Cart3D terms, the PaceLab orientation is described as $(-Xb, Yb, -Zb)$.

The **reference area & lengths** were calculated using the geometric planform of the wing, which was composed of two trapezoidal panels. These values were expressed in millimeters since that was the length scale used by PaceLab when exporting the surface mesh.

The **CFL setting** for Cart3D acts as a sort of pseudo-"time step" similar to the forward-Euler method for initial value problems.[70] Cart3D is a static rather than dynamic analysis so this is something of a false analogy, but the flow solver arrives at a solution by iterating until the flow behavior throughout the space has converged to some steady-state solution. The CFL value determines how much the flow can change between iterations.[37] The larger the CFL value, the more the state of the simulation will change after each iteration. Smaller values correspond to a smaller change per iteration, which slows convergence to a solution but reduces sensitivity to numerical instabilities, such as may occur in transonic or highly-separated flow. The default value in Cart3D is 1.1 and the minimum value used in Cart3D's adaptive grid refinement script **aero.csh** is 0.2. If convergence problems are identified during an analysis, the minimum value is used. CFL numbers used in this effort occasionally ranged as high as 1.3 if the problem was expected to be well-behaved numerically.

Once created, the master job description file would be uploaded to a High Performance Computing system, along with the surface triangulations. Additional scripts would then be used to set up each case to be run.

D.1.3 Setting Up Cases on HPC Systems

An assortment of files were used to define and run each Cart3D analysis. Perl scripts were written to customize these files to the case being run and the desired Cart3D behavior.

D.1.3.1 Config.xml

This file, expressed in XML, gives names to each of the numbered components in the surface mesh. The component numbers are derived from the order they are listed in the **intersect** command during geometry generation. Although not strictly necessary for use of Cart3D, the configuration name in this file is set using the batch name, case number, and flight condition ID from the master job description file.

D.1.3.2 Input.cntl

This is the primary file which controls Cart3D. Information from the master job description file is used to set values for the Mach number, alpha (angle of attack) and beta (sideslip angle). A CFL number is defined here using the specified baseline value, although this value may be overridden by a similar entry in **aero.csh**. The orientation of the vehicle is defined in this file as well, along with the reference area and one reference length. Only one reference length can be specified in this file, so the mean aerodynamic chord is commonly used for normalization of the pitching moment. The reference length for lateral moments will come into play in the CLiC files, described below.

The objective function for the adaptive gridding logic of **aero.csh** is also specified in this file. Although the objective function can incorporate any combination of the aerodynamic forces, it can only include *one* of the aerodynamic moments at a time. For these efforts, the pitching moment coefficient and drag coefficient were given weights of 1, while the lift coefficient was given a weight of 0.5. For flight conditions with nonzero sideslip angles, the side force coefficient was also given a weight of 0.5. No rigorous evaluation of the effect of these values on Cart3D behavior was undertaken.

This file is created based on information from the master job description file during the job creation process using the script **batch_adjoint_multiple.pl**, which in turn calls the script **c3d_input_maker.pl**.

D.1.3.3 clic.cntl

CLiC is a post-processing module which takes as input an annotated triangulation of a body or vehicle (a TRIQ file) and produces force and moment coefficients for the entire configuration and/or for individual components.[31] The module also requires an input script, named **clic.cntl** by default, which defines the relevant parameters. These parameters include the local vehicle orientation, the angle of attack, the sideslip angle, the reference area and reference length, the location of the center of mass, and (if desired) the component number and reference axis for component-level moments. Such component-level moments can be used to calculate the hinge moments necessary to deflect the control surfaces to the specified

positions.

Two sets of CLiC input files are generated for each analysis using Perl scripts. One set uses the mean aerodynamic chord as the reference length for accurate calculation of the pitching moment coefficient, while the other set uses the half-span for calculation of the rolling and yawing moment coefficients. Each set also includes four center-of-mass (C.O.M.) locations. The first location is the one calculated by the mass estimating relationships, and serves as the best guess of the true C.O.M. location of the vehicle. The other three locations are derived by nudging the C.O.M. one meter along each Cartesian axis. The resulting changes in the aerodynamic moments can be used to determine the effect of shifting the C.O.M. of the vehicle, such as different distributions of internal components such as batteries or RCS propellant tanks. The change in aerodynamic moments due to a change in the C.O.M. of the vehicle can also be calculated using the angle of attack, sideslip angle, reference lengths, and aerodynamic force coefficients; data from the extra CLiC results serve as a confirmation and double-check of the analytically-derived results.

These CLiC input files are generated during the job creation process that is carried out by the Perl script **batch_adjoint_multiple.pl**, which in turn calls the scripts **clic_input_maker.pl**, **clic_input_maker_perturbx.pl**, **clic_input_maker_perturby.pl**, and **clic_input_maker_perturbz.pl**, which in turn would instruct CLiC to calculate the baseline forces and moments plus the effects of a 1-meter C.O.M. change in the x-, y-, and z-directions respectively.

D.1.3.4 aero.csh

This script was included as part of the Cart3D distribution. It executed both the adaptive gridding logic and the flow solver, cycling between the two to refine the volumetric mesh in the regions which most affected the solution, as calculated using the weighting function from the **input.cntl** file. Most parameters were left unchanged from the default settings, aside from the CFL value (specified in the master job description file) and the spanwise orientation flag (changed to indicate that the Y dimension corresponded to the wingspan of the vehicle).

Additionally, the mesh growth rates would be tweaked for each flight condition. These rates would be adjusted, following the trend in the default **aero.csh** file of larger rates in later rounds, until the script produced grids with roughly two or three million cells, as evaluated with one or two dozen configurations. Typically, supersonic flight conditions required higher growth rates to produce that number of cells; it is believed that the shock waves present at those conditions resulted in a more rapid convergence of surface pressure distribution.

It is quite possible that volumetric grids with fewer cells would produce results that were equally accurate at those flight conditions. Testing this possibility would require a series of mesh sensitivity studies, repeated for different portions of the design space and different flight conditions. Given the high availability of computing resources, it was concluded that those studies were unlikely to identify sufficient potential savings to justify their execution.

Perl scripts were written to allow the settings in **aero.csh** to be adjusted based on the master job description file. The job creation script **batch_adjoint_multiple.pl** would first call the Perl utility **aeroCSH_maker.pl**, which would write the initial section of **aero.csh** which contained the user-controlled settings. This initial section would then be joined with the remainder of the default **aero.csh** file which was not intended to be modified by the user.

D.1.3.5 setBoxRunAero.pl

This Perl script was created as a wrapper for the actual command to run Cart3D (**./aero.csh**). Although most failed cases resulted from improper geometry triangulations, occasionally a case would fail even though the triangulation was watertight. Watertightness was tested during the building phase, as described in Section D.1.1. Review of the Cart3D User's Group identified other users who had occasionally encountered this problem. One user suggested that a perturbation of the outer mesh boundary settings might address the problem.[168] Another user indicated that Cart3D would sometimes behave poorly if the volumetric domain boundary aligned exactly with the edge of the configuration being analyzed.[178] In light of these suggestions, it was hypothesized that, if a case were found

to fail during Cart3D analysis, it might run successfully if the outer volumetric boundary were shifted.

The Cart3D utility **autoInputs** was used to define the outer volumetric boundary. This utility initializes the volumetric mesh at a certain distance, which is proportional to the maximum Cartesian dimension of an axis-aligned bounding box that encloses the object being analyzed.[124] In short, the representative length L is the maximum extent of the object in any one Cartesian direction, and the volumetric mesh is defined as a cube with sides of length $(n \times L)$, where n is a user-controlled scaling parameter. Cart3D documentation indicates that for subsonic flow, the value of n should be roughly 20–30; this value can be reduced for supersonic flow.[5]

Initially, **setBoxRunAero.pl** runs **autoInputs** with an n value, or box size, of 24. If the case does not run to completion – i.e., it does not generate a file named "entire.dat" in the "adapt08" folder – the **setBoxRunAero.pl** script will increment the box size by 1 and re-run **autoInputs** and **aero.csh**. If the box size exceeds 30, attempts to run the case are aborted and an empty "entire.dat" file is generated in the adapt08 directory to avoid any future attempts to re-analyze the case.

When **setBoxRunAero.pl** is run, it first searches for "entire.dat" in the folder "adapt08". The existence of this file would indicate that the current analysis has already been completed, and thus Cart3D does not need to be run again. If the "entire.dat" file does not exist but "adapt##" folders are present, it is assumed that a previous run ended unsuccessfully. This is also true if a "STOP" file is found in the working directory. If the case did not complete successfully but "adapt##" folders or a "STOP" file is present, the **setBoxRunAero.pl** script will purge all "adapt##" folders and the "STOP" file and re-run Cart3D.

Consultation with HPC systems help desks led to the modification of this script: the "lfs setstripe" command was used to change the number of object storage targets (OSTs) that would be used to write the files to disk. The default value is 6, but due to the large number of (relatively) small files generated by Cart3D, the help desks requested that the OST settings be changed so that only 1 OST was used for each directory to mitigate system loads.

D.1.3.6 PBS_maker.pl

This PBS file creation script would be called by the job creation script **batch_adjoint_multiple.pl**. It would draw upon information in the master job description file to create the PBS job scripts that would be submitted to the queue. These job scripts included PBS commands that controlled the number of processors requested, the duration of the request, and the HPC account which should be charged for the resources used.

Additionally, the scripts contained the commands which would be executed when the job was run. Typically these commands included changing the current directory to that of a case to be analyzed, running **setBoxRunAero.pl** to analyze the case if necessary, and running the **aero_archive.csh** script after completing a job to clean up unnecessary files. Often a single PBS job file would include 10–25 cases. This simplified the bookkeeping of job resources, as it reduced the overall number of jobs submitted to the queue and simplified the workload of the queue control software.

D.1.3.7 submit.pl

This script was fairly simple compared to the others. It would check the current directory for any PBS job files and submit them sequentially with a 10–20 second delay between submissions so as not to overload the queue software. A subdirectory, "submittedPBS," would also be created if it did not exist; after each PBS job was submitted, the job file was moved to "submittedPBS" so that it would not be re-submitted if the script were called again.

D.1.3.8 refreshQueue.pl

This script was used primarily on the U.S. Army Engineer Research & Development Center (ERDC) system called Diamond. That system offered high throughput of Cart3D cases compared to other systems, and it was difficult to manage the queue to maximize the number of analyses completed without negatively impacting the experience of other users of the system. Virtually unlimited jobs could be submitted to the queue at a time, but too many jobs would make it difficult for other users to find their jobs within the queue.

Conversely, if too few jobs were submitted at a time, they might all finish before the next time the author checked the queue, reflecting lost chances to complete more analyses.

Rather than checking the status of the queue every few hours, this script was written to automate the process. When run, the script would query the number of jobs waiting in the queue that belonged to the author. That number was compared against a minimum limit defined in the script (typically 20). If the author had fewer than 20 jobs in the queue waiting to start, the script would query the number of *running* jobs that belonged to the author. If this number were less than a cutoff (e.g. 50), the script would submit a new PBS file to the queue.

If a new PBS file was to be submitted, the script would move to a folder which contained PBS scripts to be submitted and select one to submit. Because many jobs could be running simultaneously, each of which might call **refreshQueue.pl** at around the same time, the script could not simply select the first file in that folder. Such a strategy resulted in the same job being submitted multiple times before it could be moved to a different folder. Instead, one of the first 50 jobs in that folder was selected at random for submission. The **refreshQueue.pl** script would then pause for a random duration between 45 and 115 seconds before re-checking the status of the queue.

This script drastically reduced the user effort required to ensure that an appropriate number of jobs were in the queue on Diamond at all times. The user needed only to log in sporadically to collect finished results and to add more PBS job files to the appropriate folder.

D.1.3.9 batch_adjoint_multiple.pl

This script performed all the tasks necessary to set up a set of cases for analysis. It would read the master job description file and call other scripts to generate the various files required by Cart3D. A directory for the case at hand would then be created in the scratch space, and all necessary files moved to that directory. This script would also run the script which created the PBS job file to run the analysis.

D.1.3.10 pullCase.pl

This script was used to collect finished cases. The user defined a batch name, a range of case ID values, and a flight condition number. All cases which matched the description would then be moved to a separate folder in the scratch space corresponding to that particular batch and flight condition. Once this move was completed, the **collator_adj.pl** script was copied to the new folder and run in order to parse the output of the analyses.

D.1.3.11 collator_adj.pl

This script parsed the Cart3D output files for useful data and recorded it in a convenient set of files. In each case directory, it used multiple files to accumulate the desired data sets.

First, the script would read the "input.cntl" file and record the Mach number, angle of attack, sideslip angle and reference area. Secondly, it would read the "clic_lat.cntl" and "clic_lon.cntl" and record the lateral and longitudinal reference lengths, respectively.

Next, it would open the "entire.dat" file, which contains the iteration history of the forces and moments on the vehicle, and parse the final 30 lines to evaluate convergence. The average and standard deviation of each response was calculated. Because the forces were in the body-aligned frame, those forces were transformed using the angle of attack and sideslip angle to determine the lift, drag and side forces.

The next step was to read the various CLiC output files. Those files detailed the center of mass that was used for the moment calculations, as well as the final force and moment values. Additionally, those files included the calculated hinge moment acting on each control surface about its hinge line. Using the difference between the whole-body moments about the perturbed and unperturbed center of mass, the change in moment with respect to changing center of mass could be calculated. This quantity could be determined analytically as well; these calculations were used primarily as a sanity check.

The results were written to a set of output files. One file contained the forces and moments in the form of lift, drag, side force, roll, pitch, etc, as well as the convergence data and the effects of C.O.M. changes on the aerodynamic moments. Another file contained the same results but in axial-normal-lateral format. A third file contained the hinge moments

for each case. The fourth file contained only the convergence results. These files were then renamed according to the current batch name and copied to the user's home directory for easy collection.

D.1.4 Post-Processing

Once the results had been downloaded from the High Performance Computing systems, they were processed to link each set of results to the corresponding input values and filter out any cases which did not run correctly. Because the goal was to model multiple flight conditions, if a case did not run correctly for every flight condition, it was discarded from all data sets.

The parsing script would read in the various output files that had been created by **collator_adj.pl** and the input deck that had been used to generate the cases. A line was read from each data file and parsed for case ID; if this ID was not the same for every data file, the script would throw an error and halt operation.

If all ID numbers matched, the corresponding case was read from the input deck, and the current data line in each file would be fully parsed for details such as flight condition, reference scale values, and force & moment results. The convergence data – including average response value, standard deviation of the response, and the ratio of the standard deviation to the average – was also parsed.

An unexpected quirk of Cart3D was the fact that the iteration history for the aerodynamic moments was calculated as if the C.O.M. were at $(0, 0, 0)$ rather than the specified value. The post-processing script corrected the iteration data to match the specified value, along with the reference length, the average force values, and the angle of attack and sideslip angle.

A number of "goodness" checks were then performed to ensure that no nonsensical results were included in the final data set:

- If any drag coefficient was less than or equal to zero, the case would be rejected;
- if the absolute value of the lift coefficient was greater than 20, this was taken as an indication that Cart3D had converged to a nonsensical answer, and the case would be rejected;

- if the absolute value of the pitching moment coefficient was greater than 50, the case was considered to be nonsensical and rejected; and
- if the standard deviation of the pitching moment coefficient C_M *and* the ratio of the standard deviation of C_M to the average value of C_M were *both* larger than 0.05, the case was considered to be insufficiently converged, and was rejected.

This final requirement deserves more attention. Either criterion alone would not be sufficient: a case could have a larger standard deviation and still be converged if the response value was large with respect to the standard deviation. Conversely, a converged case might have a larger ratio of standard deviation to average and still be converged if the average value were close to zero. By combining the two criteria, the only rejected cases would be those which exhibited both significant absolute variability (i.e. large standard deviation) and significant relative variability (i.e. large ratio of standard deviation to average).

Cases which were not rejected were included in the combined output data set. This data set combined the input settings for each case with the aerodynamic responses at each flight condition for ease of reference.

APPENDIX E

EXAMPLE SCRIPTS FOR CONTOUR-BASED SAMPLING & GHOREYSHI COKRIGING

This appendix will document the actual code used to choose new cases for analysis. The author freely admits that programming efforts focused more on functionality than grace, and requests the indulgence of the reader whenever the implementation is inefficient or ungainly.

E.1 Contour-Based Sampling with Ghoreyshi Cokriging

This is the utility that identifies the most promising candidate for analysis. This Matlab implementation of the utility uses the bounded-range weighting function described by Picheny et al.[149] It calls several functions from the DACE toolbox,[107] which can be downloaded for free from the Technical University of Denmark at `http://www2.imm.dtu.dk/~hbni/dace/`.

E.1.1 Input Parameters

When calling this utility, the user must provide a variety of input data:

- **S**: the matrix of input values for the existing samples, with each row corresponding to a different sample;
- **Y**: the vector or matrix of response values for the existing samples, with each row corresponding to a different sample and each column being a different response;
- **modeltype**: the function handle for the underlying trend type for the Kriging model, such as "@regpoly1";
- **correlationtype**: the function handle for the correlation type that will be used for the Kriging model, such as "@corrgauss";

- **numPoints**: the number of samples that will be selected in this round (recommended value is 1);
- **threshold**: a scalar or vector, containing the target value for each response that will be included in the adaptive-sampling efforts;
- **ErrorBounds**: a scalar or vector, indicating the breadth of the "region of interest" around the target value(s) (for example, to target $Y = 0 \pm 0.1$, *threshold* would be 0 and *ErrorBounds* would be 0.1);
- **candidates**: a matrix of input values for the candidate points;
- **testpoints**: a matrix of input values for the test points that will be used to evaluate the candidates;
- **min_chance**: a scalar that indicates the minimum probability-of-interest (POI) requirement that will be used to filter out poor candidates – see Section 4.9 for more details;[1]
- **surrogates**: a cell vector of function handles for low-fidelity data;
- **suffix**: a string that will be used to identify the output data files generated by this script (a different suffix should be used for each batch of samples being selected);
- **round**: an integer that will be used to identify the particular execution of the script within a batch of samples; and,
- **plotPOI**: a flag that tells the code whether or not to plot the observed probability-of-interest values for each response. These plots may be useful when selecting a reasonable *min_ chance* value. A flag value of 1 will tell the code to make the plots and save them to the working directory; any other value for this flag will omit these plots.

[1] If none of the candidates has a probability-of-interest (POI) value greater than or equal to this requirement, the sample-selection utility will select the candidate(s) with the largest POI value(s).

E.1.2 Output Parameters

Two output parameters are returned to the script or function that calls this utility:

- **dmodel**: a structure (or cell array of structures) containing the Kriging surrogate model(s) created by the DACE Kriging toolbox for Matlab; and
- **np**: a vector (or matrix, if *numPoints*>1) with each row corresponding to a sample that was identified as promising.

In addition, the utility will write data files to the local directory. These data files may be used to review the performance of the utility and document its calculations.

In the first data file, named "sampling_size_vs_num_candidates_< $suffix$ >.csv" (where < $suffix$ > is the string given as an input parameter), each row corresponds to one execution of the utility. The row contains:

- The number of data points already in the training data set;
- the *min_chance* value that was specified by the user;
- the number of candidates which had POI values greater than *min_chance*;
- the total number of candidates available;
- the amount of time elapsed during that execution of the utility (expressed in minutes);
- the POI value for the candidate that was selected as the best sample; and,
- the number of candidates that were *not* chosen as the best sample which had POI values higher than that of the selected candidate.

Some of these values are primarily useful for documentation, such as the number of existing data points and the required POI value for that round. The number of candidates with POI values greater than *min_chance* can help the user decide whether the current *min_chance* value should be changed: if no candidates are meeting the requirement, the sample that is selected may be very conservative (i.e. close to an existing sample), which may not be desirable.

The total number of available candidates, the number that exceeded the *min_chance* requirement, and the elapsed time will all help the user decide whether to increase or decrease the number of candidates being submitted. Using more candidates will lead to increased analysis time, but may result in better options for the sample-selection algorithm. A lower *min_chance* value will filter out fewer candidates, leading to increased analysis time but allowing the algorithm to choose candidates that are more exploratory. If very few candidates are exceeding the *min_chance* requirement, the user may wish to increase the total number of candidates or decrease the *min_chance* requirement to avoid the very conservative sample-selection behavior mentioned in the previous paragraph.

The second data file, "variance_records_$<suffix>$.csv", is typically much larger but more complete. This data file includes:

- The normalized weighted integrated mean squared error (wIMSE) score, averaged across all responses, for the sample that was selected for analysis;
- the normalized wIMSE score for each response;
- the maximum likelihood that the selected point will *not* fall within the range of interest for any one response;
- the input settings for the selected point;
- the selected point's *un*-normalized wIMSE scores for each response; and,
- the average and standard deviation of all wIMSE scores for each response.

This data allows the user to investigate how the selected sample compared to the other samples that were considered. Comparing the wIMSE score of the selected point against the average and standard deviation of all candidate samples for each response will indicate *why* the sample was chosen, and may shed some light on the behavior of the algorithm: does the selected sample offer large wIMSE gains in a single response, or is it expected to produce moderate improvements across multiple responses?

E.1.3 Matlab Code

```
function [dmodel,np]=...
    krigfit(S, Y, modeltype, correlationtype, numPoints, threshold, ...
    ErrorBounds, candidates, testpoints, min_chance, suffix, ...
    surrogates, round, plotPOI)
tic
Q = length(Y(1,:)); % Number of responses to be modeled.
acceptable_distance = 0.00001;  % Can be changed - see about allowable
                                % proximity of candidate points below.
number_of_dimensions = length(S(1,:));
num_candidatepoints = length(candidates(:,1));
num_testpoints = length(testpoints(:,1));
rejected = 0;

% % Set constants for later use in Cumulative Distribution Function
% % calculations.
b0 = 0.2316419;
b1 = 0.31938153;
b2 = -0.35656378;
b3 = 1.78147794;
b4 = -1.82125598;
b5 = 1.33027443;

% Initialize some variables
numvars = length(S(1,:))+1; % The extra variable will be the
                            % cheap response estimate used by
                            % Ghoreyshi cokriging.
theta = 10*ones(1,numvars);
lob = 1e-1*ones(1,numvars);
```

```
upb = 20*ones(1,numvars);
predVal = zeros(num_candidatepoints,Q);
candidate_mse = zeros(num_candidatepoints,Q);
predVal_test = zeros(num_testpoints,Q);
W = zeros(num_testpoints,Q);
already_included = zeros(num_candidatepoints,1);

% Generate low-fidelity estimates for the existing training data points.
ypred = zeros(length(S(:,1)),Q);
for q = 1:Q
  ypred(:,q) = feval(surrogates{q},S);
end

% Create initial Ghoreyshi cokriging models for all responses based on
% the available training data.
dmodel = cell(Q,1); perf = dmodel;
for q = 1:Q
  [dmodel{q}, perf{q}] = dacefit([S ypred(:,q)], Y(:,q), modeltype,...
  correlationtype, theta, lob, upb);
end

%  Make predictions for the response value & uncertainty at each
%  test point.
W_nocand=zeros(num_testpoints,1);
saveprob1 = W_nocand;
saveprob2 = W_nocand;
Res_MSEs_nocand=zeros(num_testpoints,Q);

% Estimate the low-fidelity response values for each test and
```

```
% candidate point.
ypredtest = zeros(num_testpoints,Q);
ypredcand = zeros(num_candidatepoints,Q);
for q = 1:Q
  ypredtest(:,q) = feval(surrogates{q},testpoints);
  ypredcand(:,q) = feval(surrogates{q},candidates);
end

% Using Ghoreyshi cokriging and the estimated low-fidelity response
% values, predict the high-fidelity response values for the test points.
for q = 1:Q
  [predVal_test(:,q), Res_MSEs_nocand(:,q)] = predictor([testpoints ...
   ypredtest(:,q)],dmodel{q});
end

% Initialize variables that will be used to predict response values
% for the candidate points, as well as their likelihood of having
% response values within the range of interest.
predVal_cand = zeros(num_candidatepoints,Q);
predMSE_cand = zeros(num_candidatepoints,Q);
prob_interest = zeros(num_candidatepoints,Q);
saveprob1c = zeros(num_candidatepoints,Q);
saveprob2c = zeros(num_candidatepoints,Q);
for q = 1:Q
   [predVal_cand(:,q), predMSE_cand(:,q)] = predictor([candidates ...
      ypredcand(:,q)],dmodel{q});
   for cindex = 1:num_candidatepoints
      % Calculate standard normal cumulative distribution functions.
      % These will be used to estimate the probability that the response
```

```
% at the point in question falls within the stated bounds of
% interest.
% First, calculate the likelihood that the response value is larger
% than the upper bound of interest.
predVal1c = predVal_cand(cindex,q);
mse_herec = predMSE_cand(cindex,q);
reverse1 = 0;
% This probability method only works for testValue < threshold;
% if this is not the case, reverse the calculation.
if predVal1c > (threshold(q)+ErrorBounds(q))
   predVal1c = predVal1c - ...
      2*(predVal1c-(threshold(q)+ErrorBounds(q)));
   reverse1 = 1;
end
% Next, calculate the likelihood that the response value is larger
% than the lower bound of interest.
predVal2c = predVal_cand(cindex,q);
reverse2 = 0;
if predVal2c > (threshold(q)-ErrorBounds(q))
   predVal2c = predVal2c - ...
      2*(predVal2c-(threshold(q)-ErrorBounds(q)));
   reverse2 = 1;
end
% Boundary 1: threshold + tolerance
Z_norm1c = (threshold(q) + ErrorBounds(q) - ...
   predVal1c)/sqrt(mse_herec+0.0000000000001);
t1c = 1/(1+b0*Z_norm1c);
standard_normal1c = (1/sqrt(2*pi*1^2))*exp( -((Z_norm1c ...
   -0)^2)/(2*1^2));
```

```
% Boundary 2; threshold - tolerance
Z_norm2c = (threshold(q) - ErrorBounds(q) - ...
   predVal2c)/sqrt(mse_herec+0.0000000000001);
t2c = 1/(1+b0*Z_norm2c);
standard_normal2c = (1/sqrt(2*pi*1^2))*exp( -((Z_norm2c ...
   -0)^2)/(2*1^2));
% Using the Abramowitz & Stegun (1964) approximation for CDF(x)
% of a normal distribution
% - wikipedia says abs. error <7.5e-8?
% This method calculates the cumulative probability that the
% predicted response value is less than the threshold value.
% This method is ONLY valid when predicting the likelihood that
% a response is less than the threshold!
prob1c = 1-standard_normal1c*(b1*t1c + b2*t1c^2 + b3*t1c^3 + ...
   b4*t1c^4 + b5*t1c^5);
prob2c = 1-standard_normal2c*(b1*t2c + b2*t2c^2 + b3*t2c^3 + ...
   b4*t2c^4 + b5*t2c^5);
if reverse1 == 1
   % If the predicted value was actually greater than the lower
   % bound, the correction means we calculated the chance that
   % the predicted response is GREATER than the threshold. In
   % that case, we need to reverse it back to P(y<t).
   prob1c = 1-prob1c;
end
if reverse2 == 1
   % If the predicted value was actually greater than the upper
   % bound, we need to reverse this one too.
   prob2c = 1-prob2c;
end
```

```
      saveprob1c(cindex,q) = prob1c;
      saveprob2c(cindex,q) = prob2c;
      % The chance that the candidate point has good performance is
      % equal to prob1c (the probability that the case has a response
      % value greater than the lower bound) minus prob2c (the
      % probability that the case has a response value greater than
      % the upper bound)
      prob_interest(cindex,q) = prob1c-prob2c;
   end   % closing ''for cindex = 1:num_candidatepoints''
end      % closing ''for q = 1:Q''

% Reproduce some of the DACE normalization functions so that we can
% also normalize our test points.
% Normalizing data:
[m n] = size(S);
mS = mean(S);   sS = std(S);
j = find(sS == 0);
if  ~isempty(j),  sS(j) = 1; end
Snorm = (S - repmat(mS,m,1)) ./ repmat(sS,m,1);
% End of DACE normalization functions.

IMSE = zeros(num_candidatepoints,Q);
% Set up the portions that don't depend on the candidate point.
% This includes the matrix inversion, which'll be one of the more
% intensive operations.
Cinv = cell(1,Q);
F = cell(1,Q);
for q = 1:Q
   % Convert DACE outputs to the format given in Picheny
```

```
   % ('Adaptive Designs of Experiments for Accurate Approximation
   % of a Target Region', Journal of Mechanical Design, 2004)
   % In DACE, C is the Cholesky factorization of the correlation
   % matrix R.
   % In Picheny, C is the correlation matrix itself.
   daceCholesky = full(dmodel{q}.C);
   daceCinv = inv(daceCholesky);
   % Create inverse of correlation matrix from Cholesky factorization
   % provided by DACE toolbox
   Cinv{q} = daceCinv' * daceCinv;
 % Cholesky factorization is triangular and thus easy to invert
 % than the original matrix
   F{q} = full(dmodel{q}.C) * dmodel{q}.Ft;
     % from DACE manual, equation 3.10
end

% If the plotPOI flag is set to one, create plots of the calculated
% POI values to help the user select a reasonable min_chance value.
if plotPOI == 1
   if round ==1
      rez = 900;
      for q = 1:Q
         close all
         f1=figure(1);
         plot(prob_interest(:,1),'o')
         xlabel('Candidate Number')
         ylabel('Calculated POI values')
         title(['Probability of Interest Results for Response #' ...
            sprintf('%d',q)])
```

```
            print(f1,[suffix '_Response' sprintf('%d',q) '.png'], ...
               '-dpng', ['-r',num2str(rez)], '-opengl')
         end
      end
   end

factor1 = zeros(num_testpoints); factor2 = factor1;
for j = 1:num_candidatepoints
   % Only evaluate candidate points that exceed the required POI
   % value set in min_chance
   if min(prob_interest(j,:)) > min_chance
      % Start a clean variable to hold MSE estimates for each
      % test point.
      Res_MSEs = zeros(num_testpoints,Q);
      % If a candidate point is too close to a training data point,
      % the resulting Kriging correlation matrix becomes close to
      % singular and extreme or imaginary numbers are produced.  These
      % calculations & conditional statement attempt to avoid that
      % possibility by skipping candidate points that are very close
      % to training points.
      standins = repmat(candidates(j,:),m,1);
      R1dist = 0;
      for d = 1:number_of_dimensions
         R1dist = R1dist + (S(:,d)-standins(:,d)).^2;
      end
      R1dist = min(R1dist);
      % Only evaluate candidate points that are sufficiently far
      % from the training data points to avoid singularity problems.
      if (R1dist >=acceptable_distance)
```

```
% Cycle through all surrogate models to estimate MSEs
for q = 1:Q
   mS = mean([S ypred(:,q)]);   sS = std([S ypred(:,q)]);
   Snorm = ([S ypred(:,q)] - repmat(mS,m,1)) ./ repmat(sS,m,1);
   % Normalize the candidate point according to the DACE
   % rules defined above.
   mSj = mS; sSj = sS;
   newS = ([S ypred(:,q)]- repmat(mSj,m,1)) ./ repmat(sSj,m,1);
   candidate_norm = ([candidates(j,:) ypredcand(j,q)] - mS)...
     ./ sS;
   % Calculate the normalized distances for correlation
   % calculations.
   % Distance calculations modified from DACE:
   % Calculate distances D between points
   D_temp = repmat(candidate_norm, m, 1) - newS(1:m, :);
      % differences between points
   % End of distance calculations from DACE.
   % Compute c_new to be used in easier matrix inversion
   c_new = feval(correlationtype,dmodel{q}.theta,D_temp);
   C_kplusone_inv = [1 zeros(1,m); -Cinv{q}*c_new eye(m)] * ...
      [ 1/(1-c_new'*Cinv{q}*c_new) zeros(1,m); ...
      zeros(m,1)  Cinv{q}] * ...
      [ 1 -c_new'*Cinv{q}; zeros(m,1) eye(m)];
   Snorm_can = [candidate_norm; Snorm];
   % Compute f(x_candidate) and use it to calculate F_k+1
   [f_can ~] = feval(dmodel{q}.regr, candidate_norm);
   F_kplusone = [f_can;F{q}];
   to_be_inverted_term = ...
      (F_kplusone'*C_kplusone_inv*F_kplusone);
```

```
% The default approach would be to training a whole new
% Kriging model to include each candidate point, build
% the new correlation matrix C from the Choleski
% factorizations, and then invert that new C matrix when
% calculating MSE. The method described in Picheny reduces
% the effort involved in that process by re-using most of
% the old C matrix, since it won't change. We still had to
% invert a matrix, but it's PxP instead of MxM or so.

% For every test point, evaluate uncertainty &
% weighting values.
for k = 1:num_testpoints
   %Normalize according to the DACE rules above.
   test_norm = ([testpoints(k,:) ypredtest(k,q)] - mSj)...
      ./ sSj;
   D_test = repmat(test_norm, m+1, 1) - Snorm_can(1:m+1,:);
      % Distances between points
   % Compute c(xt), f(xt) for each test point xt
   c_test = feval(correlationtype,dmodel{q}.theta,D_test);
   [f_test ~] = feval(dmodel{q}.regr, test_norm);
   % Compute MSE at each test point
   % for each surrogate model.
   factor1(k,q) = c_test'*C_kplusone_inv*c_test;
   factor2(k,q) = (f_test ...
      - c_test'*C_kplusone_inv*F_kplusone)* ...
      (to_be_inverted_term \ ...
      (f_test-c_test'*C_kplusone_inv*F_kplusone)');
   Res_MSEs(k,q) = dmodel{1}.sigma2*(1 - ...
      c_test'*C_kplusone_inv*c_test + ...
```

```
            (f_test -c_test'*C_kplusone_inv*F_kplusone)* ...
            (to_be_inverted_term \ ...
            (f_test-c_test'*C_kplusone_inv*F_kplusone)')  );
      end % for k = 1:num_testpoints
      % This ends the loop that estimated what the MSE for
      % each test point WOULD be, if the current candidate
      % point were added to the model (and therefore reduced
      % MSE for any test points close to the candidate point).
      % This loop was for a single response.
   end % for q = 1:Q

   for q = 1:Q
      IMSE(j,q) = 0;
      % Assign reasonable values to IMSEs that haven't been
      % calculated yet - they'll be updated later, but right
      % now we need a rough average IMSE value.
      for splots = j+1:num_candidatepoints
         IMSE(splots,q) = mean(IMSE(1:(j-1),q));
      end
      for k = 1:num_testpoints
         standins = repmat(testpoints(k,:),(m),1);
         R2dist = 0;
         for d = 1:number_of_dimensions
            R2dist = R2dist + (S(:,d)-standins(:,d)).^2;
         end
         R2dist = min(R2dist);
         if R2dist<acceptable_distance
            W(k,q) = 0;
         else
```

```
% Calculate cumulative distribution functions for normal
% distributions. These will be used to estimate the
% probability that the response at the test point in
% question is within the stated bounds of interest.
% That probability will then be used to calculate the
% weighting function value for that test point.
predVal1 = predVal_test(k,q);
rev1 = 0;
if predVal1 > (threshold(q)+ErrorBounds(q))
   predVal1 = predVal1 - ...
      2*(predVal1-(threshold(q)+ErrorBounds(q)));
   rev1 = 1;
end
predVal2 = predVal_test(k,q);
rev2 = 0;
if predVal2 > (threshold(q)-ErrorBounds(q))
   predVal2 = predVal2 - ...
      2*(predVal2-(threshold(q)-ErrorBounds(q)));
   rev2 = 1;
end
% Boundary 1: threshold + tolerance
Z_norm1 = (threshold(q) + ErrorBounds(q) - ...
   predVal1)/sqrt(Res_MSEs(k,q)+0.0000000000001);
t1 = 1/(1+b0*Z_norm1);
standard_normal1 = (1/sqrt(2*pi*1^2)) * ...
   exp( -((Z_norm1 -0)^2)/(2*1^2));
% Boundary 2; threshold - tolerance
Z_norm2 = (threshold(q) - ErrorBounds(q) - ...
   predVal2)/sqrt(Res_MSEs(k,q)+0.0000000000001);
```

```
            t2 = 1/(1+b0*Z_norm2);
            standard_normal2 = (1/sqrt(2*pi*1^2)) * ...
               exp( -((Z_norm2 -0)^2)/(2*1^2));

            % Using the Abramowitz & Stegun (1964) approximation
            % for CDF(x) of a normal distribution again
            % This method calculates the cumulative probability
            % that the predicted response value is less than the
            % threshold value.
            % This method is ONLY valid when predicting the
            % likelihood that a response is less than the
            % threshold!
            prob1 = 1-standard_normal1*(b1*t1 + b2*t1^2 + ...
               b3*t1^3 + b4*t1^4 + b5*t1^5);
            % If the response is greater than the threshold,
            % reverse the calculations so that the approximation
            % is still valid.
            if rev1 ==1
               prob1 = 1-prob1;
            end
            prob2 = 1-standard_normal2*(b1*t2 + b2*t2^2 + ...
               b3*t2^3 + b4*t2^4 + b5*t2^5);
            if rev2 ==1
               prob2 = 1-prob2;
            end
            % Compute the weighting function for the k^th test
            % point with respect to the q^th response:
            W(k,q) = (prob1-prob2);
         end % elseif R2dist > acceptable_distance
```

```
            % Add contribution of this test point to the running
            % total wIMSE value for this candidate & response.
            IMSE(j,q) = IMSE(j,q) + W(k,q)*Res_MSEs(k,q);
         end % for k = 1:num_testpoints
      end % for q = 1:Q

   else % if R1dist < acceptable_distance
      % This section of code is activated if the candidate point is
      % too close to a training point.
      already_included(j) = 1;
   end % if R1dist < acceptable_distance

else % min(prob_interest(j,:))
   % Code in this section is applied when a candidate point has
   % too low of a POI score, i.e. is too unlikely to fall within
   % the specified ranges of interest for all response values.
   rejected = rejected + 1;
   already_included(j) = 2;
   if min(prob_interest(j,:)) < 0
      prob_interest(j,:)
      error('badPOIcalcs',['At least one Probability of Interest' ...
         'value is nonsensical (<0).'])
   end
end % min(prob_interest)

% Determine an IMSE value that's reasonable but unattractive. This
% value will be assigned to all candidates that either were too
% close to training data points, or had unacceptably low POI scores.
for minij = 1:j
```

```
      if already_included(minij) > 0
         for q = 1:Q
            IMSE(minij,q) = max(IMSE(:,q));
         end
      end
   end
end  % j=1:num_candidatepoints

% Prepare to normalize each column of the IMSE matrix.
avgIMSE = ones(Q,1);
stdIMSE = ones(Q,1);
for q = 1:Q
   avgIMSE(q) = mean(IMSE(:,q));
   stdIMSE(q) = std(IMSE(:,q));
   if avgIMSE(q) < 1e-3
      IMSE(:,q) = zeros(length(IMSE(:,q)),1);
      avgIMSE(q) = 0;
      stdIMSE(q) = 1;
   end
end

% Determine normalized IMSE values
norm_IMSE = zeros(size(IMSE));
for q = 1:Q
   norm_IMSE(:,q) = (IMSE(:,q)-avgIMSE(q)) / stdIMSE(q);
end

% Calculate a net IMSE score for each candidate by averaging the
% IMSE scores of all responses.
```

```
net_IMSE = zeros(num_candidatepoints,1);
for j = 1:num_candidatepoints
   net_IMSE(j) = mean(norm_IMSE(j,:));
end

% Sort the results to find the best-performing candidate.
results = sortrows([net_IMSE norm_IMSE (1-min(prob_interest,[],2)) ...
   candidates IMSE])
% This sort operation will put the candidate with the smallest (i.e.,
% best) net_IMSE value on top of the results matrix. If ALL candidates
% had POI values that were too low, the net_IMSE and norm_IMSE columns
% will all be identical, and the sort operation will then sort the
% matrix by the (1-min(prob_interest,[],2)) column.
%   This column is the complement of the POI score for each candidate,
% and is equal to the probability that the candidate DOES fall within
% the range of interest for all responses. In effect, if none of the
% candidates meet the POI requirement specified by min_chance, this
% operation causes the algorithm to select the candidate with the
% highest POI score as the next sample.

% Identify the input settings for the selected point(s).
np = results(1:numPoints,1+Q+2:(number_of_dimensions+Q+2));

% Write the first output file.
sampletime = toc/60;
outputrecordsfile = ['sampling_size_vs_num_candidates_' suffix '.csv'];
list = [];
if exist(outputrecordsfile,'file')
    list = csvread(outputrecordsfile);
```

```
end
list = [list; length(S(:,1)) min_chance (num_candidatepoints-rejected) ...
   num_candidatepoints sampletime results(1,1+Q+1) ...
   sum(results(:,1+Q+1)>results(1,1+Q+1))];
csvwrite(outputrecordsfile,list);

% Prepare to write the second output file.
VARIANCESAVER = ['variance_records_' suffix '.csv'];
list = [];
if exist(VARIANCESAVER,'file')
    list = csvread(VARIANCESAVER,1,0);
end
LTT = fopen(VARIANCESAVER,'w');
titlestring = ['Average Normalized wIMSE Score,'];

% Create column labels for each response wIMSE value.
titlestring_wIMSE = '';
for q = 1:Q
   titlestring_wIMSE = [titlestring_wIMSE ...
      'Normalized wIMSE Score for Response ' sprintf('%d',q) ','];
end
% Append the next column label.
titlestring = [titlestring titlestring_wIMSE ...
   'Probability of Interest Score,'];

% Create column labels for each input dimension.
inDim = '';
for x = 1:number_of_dimensions
   inDim = [inDim 'Input Parameter ' sprintf('%d',x) ','];
```

```
end
titlestring = [titlestring inDim];

% Create column labels for the unnormalized wIMSE score
% for each response
unnorm = '';
for q = 1:Q
   unnorm = [unnorm 'Unnormalized wIMSE Score for Response ' ...
      sprintf('%d',q) ','];
end
titlestring = [titlestring unnorm];

% Create column labels for the average values and standard deviations
% of each set of wIMSE scores.
avgs = '';
for q = 1:Q
   avgs = [avgs 'Average wIMSE Score for Response ' ...
      sprintf('%d',q) ','];
end
standarddevs = '';
for q = 1:Q
   standarddevs = [standarddevs 'Average wIMSE Score for Response ' ...
      sprintf('%d',q) ','];
end
titlestring = [titlestring avgs standarddevs];

% Write the second output file.
fprintf(LTT,'%s\r\n',titlestring);
fclose(LTT);
```

```
list = [list; results(1,:) avgIMSE' stdIMSE' ];
% csvwrite(VARIANCESAVER,list);
dlmwrite(VARIANCESAVER,list,'-append','delimiter',',','newline','pc');
end
```

E.2 Wrapper for Contour-Based Sampling/Ghoreyshi Cokriging Function

This wrapper, also written for Matlab, serves as an example of how the sample-selection function may be called.

E.2.1 Input Parameters

The wrapper is written as a script, so there are no "input parameters" *per se*, but there are a number of user-controlled parameters that will affect the behavior of the script and the sample-selection utility.

- **inputFile**: This tells the script where to find the existing training data, i.e. the "warm start." The training data should be organized with each row denoting a different case and each column capturing a different input or output variable. Later in the script, the user must define which of the columns are inputs (S) and which are responses (Y).
- **suffix**: A unique string used to identify individual runs.
- **surrogates**: A cell array of strings, corresponding to functions that will be called as low-fidelity data sources. The order of these data sources must match the order of the responses in *inputFile*.
- **restarting**: A flag that indicates whether any other points have already been selected, but not analyzed or added to the training data set in *inputFile*. If this flag is a 1, the script will read the already-selected points from the output file and incorporate them into the sample-selection process.
- **num_candidatepoints**: The number of candidate points that will be used.
- **num_testpoints**: The number of test points that will be used.

- **threshold**: A vector containing the central value for the range of interest for each response. Again, the order must match that in *inputFile*.

- **threshold_range**: A vector containing the half-width of the range of interest for each response. For example, if for the first response *threshold* is 0 and *threshold_range* is 0.1, the algorithm will target regions of the design space where the first response is likely to be between -0.1 and 0.1.

- **samples**: The total number of samples that will be selected by the wrapper.

- **samples_per_round**: The number of samples that will be selected in each round of adaptive sampling. If this number is greater than one, the algorithm will select the best *samples_per_round* out of the available candidates in each round.

- **ranges**: The ranges of the input parameters; used to make sure the candidates and test points fill the design space.

- **correlation**: A string that indicates which of the available correlation functions (defined in *allcorrelations*) should be used when making the Kriging model.

- **KrigingType**: A string that indicates which of the options for underlying trend functions should be used when making the Kriging model.

- **min_chance**: A value between 0 and 1 that indicates what Probability of Interest (POI) value the candidates will have to achieve to be considered.

The script is currently written so that multiple *min_chance* values can be defined; the value that is passed to the sample-selection utility will depend on how many samples have already been selected. If desired, the user may also wish to modify the script so that the number of candidates and/or test points changes throughout the sampling process.

E.2.2 Output Parameters

The wrapper will write all selected cases to the text file indicated by the variable *outputFile*. Note that that file will be over-written by the script; if that file contains information that

must be preserved, the user may wish to set *restarting* to 1 so that the information in *outputFile* is read into the script and retained.

The wrapper will leave all variables in system memory, including the multi-fidelity Kriging models for each response that are contained in the cell array *dmodel*.

E.2.3 Matlab Code

```
close all
clear all;

% Read in the training data.
inputFile = 'inputs_and_outputs.csv';
dataPile = csvread(inputFile,1);
S = dataPile(:,1:9);
Y = dataPile(:,10:12);

% Define a text string to identify this run.
suffix = 'appendixTest02';
% Identify the low-fidelity data source for each response.
surrogates = { 'job01_CM_9D_oneVar', 'job02_CM_9D_oneVar', ...
   'job03_CM_9D_oneVar'};

% Define the output file name where the selected samples will be written.
outputFile = ['exampleCasesToBuild_' suffix '.csv'];
% Initialize the parameters that will be used to set up and execute
% the adaptive sample selection.
restarting = 0;
num_candidatepoints = 500;
num_testpoints = 600;
threshold = [0 0 0];
```

```
threshold_range = [0.1 0.1 0.1];
samples = 10;
samples_per_round = 1;
ranges = [0.05 0.25; ... % fuserad
          0 1; ... % top1
          0 1; ... % top2
          0 1; ... % droop
          1 3; ... % fineness
          0.3 0.7; ... % root chord
          0.1 1.5; ... % span
          0 6; ... % camber
          0.1 0.5; ... % vt-wing area ratio
          ];
number_of_dimensions = length(ranges(:,1));
Q = length(Y(1,:));

% --list of available correlation models--
allcorrelations = {@correxp @corrgauss @corrlin @corrspherical ...
   @corrspline};
correlation = allcorrelations{2};
% --list of available Kriging model types--
allKrigingTypes = {@regpoly0 @regpoly1 @regpoly2};
KrigingType = allKrigingTypes{3};
% Initialize other parameters for DACE toolbox.
numvars = length(ranges(:,1));
theta = 10*ones(1,numvars+1);
lob = 1e-1*ones(1,numvars+1);
upb = 20*ones(1,numvars+1);
```

```
if restarting==1
    % Read in any points that had already been selected.
    newPile = csvread(outputFile);
    dmodeltemp = cell(Q,1);
    newS = newPile;
    % Estimate the low-fidelity response values for training points
    % and selected points.
    S_pred = zeros(length(S(:,1)),Q);
    newS_pred = zeros(length(newS(:,1)),Q);
    for q = 1:Q
        S_pred(:,q) = feval(surrogates{q},S);
        newS_pred(:,q) = feval(surrogates{q},newS);
    end
    % Based on the training data, create a multi-fidelity surrogate
    % model to estimate the high-fidelity response values for the
    % selected points.
    newY = zeros(length(newPile(:,1)),Q);
    roundstart = 1+length(newPile(:,1))
    for q = 1:Q
        [dmodeltemp{q}] = dacefit([S S_pred], Y(:,q), KrigingType, ...
            correlation, theta, lob, upb);
        newY(:,q) = predictor(newS,dmodeltemp{q});
    end
    % Add the selected points to the training data.
    S = [S; newS];
    Y = [Y; newY];
    roundstart = length(newS(:,1))+1;
else
    roundstart = 1;
```

```
end

newCases = [];
times = zeros(samples,1);
rejected = zeros(samples,2);
rejected_fraction = zeros(samples,1);
for round = roundstart:samples
    % Define the POI requirement for each round.
    if round <= 5
        min_chance = 0.07;
    elseif round <= 7
        min_chance = 0.06;
    elseif round <= 9
        min_chance = 0.05;
    else
        min_chance = 0.00;
    end

    % Create the sets of candidates and test points for this run.
    candidates = lhsdesign(num_candidatepoints,number_of_dimensions);
    for i = 1:number_of_dimensions
        candidates(:,i) = candidates(:,i)*(ranges(i,2)-ranges(i,1)) + ...
           ranges(i,1);
    end
    testpoints = lhsdesign(num_testpoints,number_of_dimensions);
    for i = 1:number_of_dimensions
        testpoints(:,i) = testpoints(:,i)*(ranges(i,2)-ranges(i,1)) + ...
           ranges(i,1);
    end
```

```
tic
np = zeros(length(S(1,:)),1);
plotPOI = 0;
% Run the adaptive sampling algorithm.
[dmodel,np]= krigfit(S,Y,KrigingType, ...
   correlation, samples_per_round, threshold, threshold_range, ...
   candidates, testpoints, min_chance, suffix, surrogates, round, ...
   plotPOI);
times(round) = toc;
elapsedthistime = times(round)
% Estimate the low-fidelity response values for the new sample(s).
np_pred = zeros(length(np(:,1)),Q);
for q = 1:Q
    np_pred(:,q) = feval(surrogates{q},np);
end
% Using the estimated low-fidelity responses and the most recent
% surrogate model, estimate the high-fidelity response values
% for the new sample(s).
Ynew = zeros(1,Q);
for q = 1:Q
    Ynew(q) = predictor([np np_pred(:,q)],dmodel{q});
end
% Add the new sample to the training data.
S = [S; np];
Y = [Y; Ynew];

% Prepare to write the new samples to the output file.
% If necessary, include any previous samples.
```

```
    if ((restarting ==1) && (round ==roundstart))
        np = [newS; np];
    end
    newCases = [newCases; np];
    % Write the samples to the output file.
    csvwrite(outputFile,newCases);
end
```

REFERENCES

[1] Abramowitz, M. and Stegun, I., *Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables*. New York: National Bureau of Standards, 1964. pg. 932.

[2] ADMesh, *version 0.95*. http://sites.google.com/a/varlog.com/www/admesh-htm accessed April 15, 2013: Anthony D. Martin, 1996.

[3] Aeschliman, D. and Oberkampf, W., "Experimental methodology for computational fluid dynamics code validation," tech. rep., Sandia National Labs., 1997.

[4] Aftosmis, M., "flowcart robust mode." http://people.nas.nasa.gov/~aftosmis/cart3d/flowCart_run.html#robustmode, accessed May 6, 2013.

[5] Aftosmis, M., "FAQ post titled 'questions, concepts and tips on meshing and using %cubes'." Cart3D User's Group Post, June 2006. https://groups.google.com/forum/#!topic/cart3d/hdrBd_SVh34, accessed April 30, 2013.

[6] Aftosmis, M., Berger, M., and Adomavicius, G., "A parallel multilevel method for adaptively refined Cartesian grids with embedded boundaries," tech. rep., AIAA 2000-0808, 2000.

[7] Aftosmis, M., Berger, M., and Alonso, J., "Applications of a Cartesian mesh boundary-layer approach for complex configurations," in *44th AIAA Aerospace Sciences Meeting*, (Reno, Nevada), January 2006. AIAA 2006–0652.

[8] AIAA, "Guide for the verification and validation of computational fluid dynamics simulations," tech. rep., American Institute of Aeronautics and Astronautics, 1998. AIAA-G-077-1998.

[9] Alexandrov, N. M., Lewis, R. M., Gumbert, C. R., Green, L. L., and Newman, P. A., "Approximation and model management in aerodynamic optimization with variable-fidelity modeling," *Journal of Aircraft*, vol. 38, pp. 1093–1101, November-December 2001.

[10] Allen, D. M., "The relationship between variable selection and data augmentation and a method for prediction," *Technometrics*, vol. 16, pp. 125–127, February 1974.

[11] Barche, J. e. a., "Experimental data base for computer program assessment," tech. rep., Advisory Group for Aerospace Research & Development (AGARD) Report 138, 1979.

[12] Bastedo, W.G., J. and Mueller, T., "Performance of finite wings at low Reynolds numbers," in *Proceedings of the Conference on Low Reynolds Number Airfoil Aerodynamics* (Mueller, T., ed.), June 1985. UNDAS-CP-77B123.

[13] Baume, O., Skøien, J., Carré, F., Heuvelink, G., and Pebesma, E., "Data harmonization of environmental variables: From simple to general solutions," in *European Conference of the Czech Presidency of the Council of the European Union Towards Environment*, pp. 162–169, 2009.

[14] Baume, O., Skøien, J., Heuvelink, G., Pebesma, E., and Melles, S., "A geostatistical approach to data harmonization – Application to radioactivity exposure data," *International Journal of Applied Earth Observation and Geoinformation*, vol. 13, 2011.

[15] Baume, O. P., Skien, J., Heuvelink, G. B., and Pebesma, E. J., "Data harmonization with geostatistical tools: a Bayesian extension," 2008.

[16] Billah, K. and Scanlan, R., "Resonance, Tacoma Narrows Bridge failure, and undergraduate physics textbooks," *American Journal of Physics*, vol. 59, no. 2, pp. 118–24, 1991.

[17] Blackerby, W. and Cahill, J., "High Reynolds number tests of a C-141A aircraft semispan model to investigate shock-induced separation," tech. rep., Lockheed-Georgia Company, Marietta, Georgia, 1975. NASA CR-2604.

[18] Box, G. and Draper, N., "A basis for the selection of a response surface design," *Journal of the American Statistical Association*, vol. 54, pp. 622–654, September 1959.

[19] Bradford, J. and Hellman, B., "Return to launch site trajectory options for a reusable booster without a secondary propulsion system," in *AIAA Space 2009 Conference and Exposition*, (Pasadena, CA), 2009. AIAA 2009–6439.

[20] BRAINN, *Basic Regression Analysis for Integrated Neural Networks version 2.3*. Carl Johnson and Jeff Schutte, 2009.

[21] Bratley, P. and Fox, B., "Algorithm 659: Implementing Sobol's quasirandom sequence generator," *ACM Transactions on Mathematical Software*, vol. 14, pp. 88–100, March 1988.

[22] Breton, A., "Manifesto of surrealism," *P. Waldberg (1997), Surrealism (World Of Art), New York: Thames and Hudson*, 1997.

[23] Cahill, J. and Connor, P., "Correlation of data related to shock-induced trailing-edge separation and extrapolation to flight Reynolds number," tech. rep., Lockheed-Georgia Company, Marietta, Georgia, 1979. NASA CR-3179.

[24] Carpenter, J., "Fast design and operability status," in *Commercial and Government Responsive Access to Space Technology Exchange*, (Atlanta, GA), October 2011.

[25] Chaderjian, N. M., Rogers, S. E., Aftosmis, M. J., Pandya, S. A., Ahmad, J. U., and Tejnil, E., "Automated Euler and Navier-Stokes database generation for a glide-back booster," in *Proceedings of the 3rd International Conference on Computational Fluid Dynamics*, pp. 251–256, July 2004. USEFUL Owner: dcrowley Added to JabRef: 2010.08.16.

[26] Chaderjian, N., Rogers, S., Aftosmis, M., Pandya, S., Ahmad, J., and Tejnil, E., "Automated CFD database generation for a 2nd generation glide-back

booster," in *21st AIAA Applied Aerodynamics Conference*, (Orlando, Florida), June 2003. AIAA 2003–3788.

[27] Chen, V. C., Tsui, K.-L., Barton, R. R., and Meckeshimer, M., "A review on design, modeling, and applications of computer experiments," *IIE Transactions*, vol. 38, no. 4, pp. 273–291, 2006.

[28] Choi, S., Alonso, J., Kroo, I., and Wintzer, M., "Multifidelity design optimization of low-boom supersonic jets," *Journal of Aircraft*, vol. 45, no. 1, 2008.

[29] Chou, S.-J., *A Conceptual Methodology for Assessing Acquisition Requirements Robustness Against Technology Uncertainties*. PhD thesis, Georgia Institute of Technology, May 2011. pgs. 56–66.

[30] Cioppa, T. and Lucas, T., "Efficient nearly orthogonal and space-filling Latin hypercubes," *Technometrics*, vol. 49, no. 1, 2007.

[31] CLiC, *version 1.2*. `http://people.nas.nasa.gov/aftosmis/cart3d/clic/clic.html` accessed April 17, 2013: Michel Delanaye, 1999.

[32] Cobleigh, B., "Development of the X-33 aerodynamic uncertainty model," tech. rep., Dryden Flight Research Center, 1998. NASA TP-1998-206544.

[33] Cochran, W. and Cox, G., *Experimental Designs*. New York: John Wiley & Sons, Inc., 2nd ed., 1957.

[34] Cohn, D., "Neural network exploration using optimal experiment design," *Neural Networks*, vol. 9, no. 6, 1996.

[35] Collier, C., Yarrington, P., Pickenheim, M., and Bednarcyk, B., "An approach to preliminary design and analysis," in *48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, AIAA, 2007. AIAA 2007-2176.

[36] Collins, K., *A Multi-Fidelity Framework for Physics Based Rotor Blade Simulation and Optimization*. PhD thesis, Georgia Institute of Technology, 2008.

[37] Courant, R., Friedrichs, K., and Lewy, H., "On the partial difference equations of mathematical physics," *IBM Journal*, pp. 215–234, March 1967.

[38] Cox, D. and John, S., "SDO: A statistical method for global optimization," *Multidisciplinary Design Optimization: State of the Art*, 1997.

[39] Cressie, N., *Statistics for Spatial Data*. Wiley Series in Probability and Mathematical Statistics, New York, NY: John Wiley & Sons, Inc., 1993. pgs. 127–129.

[40] Crowley, D., Robertson, B., Douglas, R., Mavris, D., and Hellman, B., "Aerodynamic surrogate modeling of variable geometry," in *50th AIAA Aerospace Sciences Meeting*, (Nashville, TN), January 2012. AIAA 2012–0268.

[41] Cruz, J., Cianciolo, A., Powell, R., Simonsen, L., and Tolson, R., "Entry, descent, and landing technology concept trade study for increasing payload mass to the surface of mars," in *4th International Symposium on Atmospheric Reentry Vehicles and Systems*, 2005.

[42] Davis, D., *SMC Systems Enginering Primer & Handbook, 3rd ed.*, 29 April 2005.

[43] DeLaurentis, D., *A Probabilistic Approach to Aircraft Design Emphasizing Stability and Control Uncertainties*. PhD thesis, Georgia Institute of Technology, November 1998.

[44] DeLaurentis, D. and Mavris, D., "Uncertainty modeling and management in multidisciplinary analysis and synthesis," in *38th Aerospace Sciences Meeting*, (Reno, NV), January 2000. AIAA 2000–0422.

[45] Divan, P., "Aerodynamic Preliminary Analysis System II part II - user's manual," tech. rep., North American Aircraft Division, Rockwell International, 1981. NASA CR-165628.

[46] Dorsey, J., Myers, D., and Martin, C., "Reusable launch vehicle tank/intertank sizing trade study," in *38th Aerospace Sciences Meeting*, January 2000. AIAA 2000-1043.

[47] Doyle, J., Rosema, C., Underwood, M., and Auman, L., "Recent improvements for the 8/08 release of Missile Datcom," in *47th AIAA Aerospace Sciences Meeting*, 2009. AIAA 2009-0907.

[48] Doyle, S., Alston, K., and Winter, T., "Developing the aerodynamics module for the integrated multidisciplinary optimization object system," in *49th AIAA Aerospace Sciences Meeting*, January 2011. AIAA 2011-0008.

[49] Edwards, S. personal communication, October 2012.

[50] Eggers, T., "Longitudinal stability and trim of an Ariane 5 fly-back booster," in *12th AIAA International Space Planes and Hypersonic Systems and Technologies*, (Norfolk, Virginia), 2003. AIAA 2003–7055.

[51] Eggers, T., "Longitudinal stability and trim of an Ariane 5 fly-back booster," *AIAA Paper*, vol. 7055, no. AIAA-2003-7055, 2003.

[52] Eggers, T., "Aerodynamic behaviour of a liquid fly-back booster in transonic cruise flight," in *AIAA 21st Applied Aerodynamics Conference*, 2003. AIAA 2003–3422.

[53] FAA, "National airspace system system engineering manual, ed. 3.1," 2006.

[54] Falkiewicz, N., Cesnik, C., Crowell, A., and McNamara, J., "Reduced-order aerothermoelastic framework for hypersonic vehicle control simulation," *AIAA Journal*, vol. 49, pp. 1625–1646, August 2011.

[55] Farhang-Mehr, A. and Azarm, S., "Bayesian meta-modelling of engineering design simulations: A sequential approach with adaptation to irregularities in the response behavior," *International Journal for Numerical Methods in Engineering*, vol. 62, 2005.

[56] Forrester, A., Sóbester, A., and Keane, A., "Multi-fidelity optimization via surrogate modelling," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science*, vol. 463, no. 2088, pp. 3251–3269, 2007.

[57] Forrester, A., Sóbester, A., and Keane, A., *Engineering Design via Surrogate Modeling: A Practical Guide*. John Wiley & Sons, 2008.

[58] Forrester, A. I. and Keane, A. J., "Recent advances in surrogate-based optimization," *Progress in Aerospace Sciences*, vol. 45, pp. 50–79, 2009.

[59] Gamble, J., Cooke, D., Underwood, J., Stone Jr., H., and Schlosser, D., "The development and application of aerodynamic uncertainties and flight test verification of the space shuttle orbiter," in *Space Shuttle Technical Conference* (Chaffee, N., ed.), pp. 264–294, 1985. NASA CP-2342, Part I.

[60] Gano, S., Renaud, J., and Sanders, B., "Hybrid variable fidelity optimization by using a Kriging-based scaling function," *AIAA Journal*, vol. 43, November 2005.

[61] Garmendia, D., Douglas, R., Mavris, D., and Hellman, B., "Development of an automated parametric geometry environment for a reusable booster," in *AIAA Space 2011 Conference*, (Long Beach, CA), September 2011. AIAA 2011–7164.

[62] Ghoreyshi, M., Badcock, K., and Woodgate, M., "Integration of multi-fidelity methods for generating an aerodynamic model for flight simulation," in *46th AIAA Aerospace Sciences Meeting and Exhibit*, (Reno, NV), January 2008. AIAA 2008–0197.

[63] Ghoreyshi, M., Badcock, K., and Woodgate, M., "Accelerating the numerical generation of aerodynamic models for flight simulation," *Journal of Aircraft*, vol. 46, pp. 972–980, May–June 2009.

[64] Gibson, W., *All Tomorrow's Parties*. New York: the Berkley Publishing Group, 2003.

[65] Gonzalez, H., Erickson, G., McLachlan, B., and Bell, J., "Effects of various fillet shapes on a 76/40 double delta wing from mach 0.18 to 0.7," in *Symposium on Advanced Flow Management: Part A - Vortex Flows and High Angle of Attack for Military Vehicles*, NATO Research & Technology Organisation (RTO) Applied Vehicle Technology (AVT), 2001. RTO-MP-069(I).

[66] Gramacy, R. and Lee, H., "Bayesian treed Gaussian process models with an application to computer modeling," *Journal of the American Statistical Association*, vol. 103, pp. 1119–1130, September 2008.

[67] Gramacy, R. and Lee, H., "Adaptive design and analysis of supercomputer experiments," *Technometrics*, vol. 51, no. 2, pp. 130–145, 2009.

[68] Gramacy, R. and Lee, H., "Cases for the nugget in modeling computer experiments," *Statistics and Computing*, pp. 1–10, 2010.

[69] Grefenstette, J., "Optimization of control parameters for genetic algorithms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC–16, pp. 122–128, January/February 1986.

[70] Griffiths, D. and Higham, D., *Numerical Methods for Ordinary Differential Equations*. Springer Undergraduate Mathematics Series, London, UK: Springer-Verlag, 2010.

[71] Hampsten, K. and Hickman, R., "Next generation air force spacelift," in *AIAA Space 2010 Conference & Exposition*, (Anaheim, CA), 2010. AIAA 2010–8723.

[72] Han, Z., "Improving adjoint-based aerodynamic optimization via gradient-enhanced Kriging," in *50th AIAA Aerospace Sciences Meeting*, 2012. AIAA 2012-0670.

[73] Han, Z.-H., Görtz, S., and Hain, R., "A variable-fidelity modeling method for aero-loads prediction," *New Results in Numerical and Experimental Fluid Mechanics VII*, 2010.

[74] Hassoun, M. H., *Fundamentals of Artificial Neural Networks*. MIT Press, 1995.

[75] Hastie, T., Tibshirani, R., and Friedman, J., *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics, New York: Springer, 2008. pgs. 241-249.

[76] Hayter, A., *Probability and Statistics for Engineers and Scientists*. Pacific Grove, CA: Duxbury Thomson Learning, second edition ed., 2002. pages 65–69, 657.

[77] Hedar, A.-R., "Global optimization test problems: Perm function." `http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO_files/Page2545.htm`, accessed May 14, 2013.

[78] Hedar, A.-R., "Global optimization test problems: Sphere function." `http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO_files/Page1113.htm`, accessed May 15, 2013.

[79] Hellman, B., "Comparison of return to launch site options for a reusable booster stage," in *1st Space Systems Engineering Conference*, Georgia Institute of Technology, 2005.

[80] Hemsch, M. and Morrison, J., "Statistical analysis of CFD solutions from 2nd drag prediction workshop," in *42nd AIAA Aerospace Sciences Meeting*.

[81] Holland, J., *Adaptation in Natural and Artificial Systems: an Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT press, 1992. Chapter 6.

[82] Hornik, K., "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, pp. 251–257, 1991.

[83] Huang, D., Allen, T., Notz, W., and Miller, R., "Sequential Kriging optimization using multiple-fidelity evaluations," *Structural and Multidisciplinary Optimization*, vol. 32, 2006.

[84] Hutchins, C., Missoum, S., and Takahashi, T., "Fully parameterized wing model for preliminary design," in *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, 2010.

[85] Hyndman, R. and Koehler, A., "Another look at measures of forecast accuracy," *International Journal of Forecasting*, no. 22, pp. 679–678, 2006.

[86] Iliff, K. and Shafer, M., "Space shuttle hypersonic aerodynamic and aerothermodynamic flight research and the comparison to ground test results," tech. rep., Dryden Flight Research Facility, 1993. NASA TM-4499.

[87] Iooss, B., "Numerical study of the metamodel validation proce," in *2009 First International Conference on Advances in System Simulation*, pp. 100–105, IEEE, 2009.

[88] Iooss, B., Boussouf, L., Feuillard, V., and Marrel, A., "Numerical studies of the metamodel fitting and validation processes," *International Journal on Advances in Systems and Measurements*, vol. 3, no. 1 & 2.

[89] Jameson, A., "Re-engineering the design process through computation," in *AIAA 35th Aerospace Sciences Meeting & Exhibit*, January 1997.

[90] Jenkins, D., Landis, T., and Miller, J., "American X-vehicles: An inventory, X-1 to X-50," tech. rep., NASA, 2003. Monographs in Aerospace History No. 31, SP–2003–4531.

[91] JMP, *Version 8.0*. SAS Institute, 2008.

[92] Johnson, M., Moore, L., and Ylvisaker, D., "Minimax and maximin distance designs," *Journal of Statistical Planning and Inference*, 1990.

[93] Journel, A. and Huijbregts, C., *Mining Geostatistics*. New York: Academic Press, 1978.

[94] Kennedy, M. and O'Hagan, A., "Predicting the output from a complex computer code when fast approximations are available," *Biometrika*, vol. 87, March 2000.

[95] Kennedy, M. and O'Hagan, A., "Bayesian calibration of computer models," *Journal of the Royal Statistical Society: Series B*, vol. 63, no. 3, pp. 425–464, 2001.

[96] Kleijnen, J. P. and Sargent, R. G., "A methodology for fitting and validating metamodels in simulation," *European Journal of Operational Research*, no. 120, pp. 14–29, 2000.

[97] Kleijnen, J. and van Beers, W., "Application-driven sequential designs for simulation experiments: Kriging metamodelling," *The Journal of the Operational Research Society*, vol. 55, August 2004.

[98] Klevanski, J. and Sippel, M., "Special aspects of flight dynamics of a reusable cryogenic booster stage," in *Fifth European Symposium on Aerothermodynamics for Space Vehicles*, vol. 563, 2005.

[99] Koch, P., Simpson, T., Allen, J., and Mistree, F., "Statistical approximations for multidisciplinary design optimization: The problem of size," *Journal of Aircraft*, vol. 36, pp. 275–286, January-February 1999.

[100] Kohavi, R., "A study of cross-validation and boostrap for accuracy estimation and model selection," in *International Joint Conference on Artificial Intelligence*, vol. 14, pp. 1137–1145, 1995.

[101] Kox, A., Klein, M., and Schulmann, R., eds., *The Collected Papers of Albert Einstein*, ch. 30: The Foundation of the General Theory of Relativity. Princeton University Press, 1997.

[102] Laflin, K., Klausmeyer, S., Zickuhr, T., Vassberg, J., Wahls, R., Morrison, J., Brodersen, O., Rakowitz, M., Tinoco, E., and Godard, J.-L., "Data summary from second aiaa computational fluid dynamics drag prediction workshop," *Journal of Aircraft*, 2005.

[103] Lee, C., *Bayesian Collaborative Sampling: Adaptive Learning for Multidisciplinary Design*. PhD thesis, Georgia Institute of Technology, 2011.

[104] Li, W., Krist, S., and Campbell, R., "Transonic airfoil shape optimization in preliminary design environment," in *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, AIAA, September 2004.

[105] Lockwood, B. and Anitescu, M., "Gradient-enhanced universal Kriging for uncertainty propagation," *Nuclear Science and Engineering*, vol. 170, pp. 168–195, February 2012.

[106] Loeppky, J. L., Sacks, J., and Welch, W. J., "Choosing the sample size of a computer experiment: A practical guide," *Technometrics*, vol. 51, pp. 366–376, November 2009.

[107] Lophaven, S., Nielsen, H., and Søndergaard, J., *DACE - A Matlab Kriging Toolbox, Version 2.0*. Technical University of Denmark, Lyngby, Denmark, 2002. IMM-REP-2002-12.

[108] Lophaven, S., Nielsen, H., and Søndergaard, J., "Aspects of the Matlab toolbox DACE," tech. rep., Technical University of Denmark, 2002. IMM–REP–2002–13.

[109] Luckring, J., Hemsch, M., and Morrison, J., "Uncertainty in computational aerodynamics," in *41st AIAA Aerospace Sciences Meeting*, (Reno, NV), January 2003. AIAA 2003–0409.

[110] MacKay, D., "Information-based objective functions for active data selection," *Neural computation*, vol. 4, no. 4, 1992.

[111] Mackman, T. and Allen, C., "Adaptive sampling for CFD data interpolation using radial basis functions," in *27th AIAA Applied Aerodynamics Conference*, 2009. AIAA 2009-3515.

[112] Mackman, T. and Allen, C., "Aerodynamic data modeling using multi-criteria adaptive sampling," in *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, (Fort Worth, TX), September 2010. AIAA 2010–9194.

[113] Mackman, T. and Allen, C., "Multidimensional adaptive sampling for global metamodeling," in *48th AIAA Aerospace Sciences Meeting*, (Orlando, FL), January 2010. AIAA 2010–1418.

[114] Mackman, T., Allen, C., M, G., and K.J., B., "Comparison of adaptive sampling methods for generation of surrogate aerodynamic models," in *49th AIAA Aerospace Sciences Meeting*, 2011. AIAA 2011-1171.

[115] Mahalanobis, P., "On the generalized distance in statistics," *Proceedings of the National Institute of Sciences of India*, vol. 2, pp. 49–55, April 1936.

[116] Manin, Y. and Panchishkin, A., *Introduction to Modern Number Theory: Fundamental Problems, Ideas and Theories*. Encyclopaedia of Mathematical Sciences, Vol. 49, Springer, 2nd ed., 2005. p. 342.

[117] Martin, J. and Simpson, T., "Use of kriging models to approximate deterministic computer models," *AIAA Journal*, vol. 43, pp. 853–862, April 2005.

[118] Masse, D. and Wilhite, A., "Aerodynamic outer mold line optimization of a reusable unmanned orbiter vehicle," in *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference*, September 2010. AIAA 2010-9354.

[119] Matheron, G., "Principles of geostatistics," *Economic Geology*, vol. 58, no. 8, pp. 1246–1266, 1963.

[120] Matlab, "Manual for lhsdesign function." MathWorks Online Documentation Center. `http://www.mathworks.com/help/stats/lhsdesign.html` accessed April 29, 2013.

[121] Matlab, "Manual for sobolset function." MathWorks Online Documentation Center. `http://www.mathworks.com/help/stats/sobolset.html` accessed May 13, 2013.

[122] Matlab, "Vectorization." MathWorks Online Documentation Center. `http://www.mathworks.com/help/matlab/matlab_prog/vectorization.html` accessed April 19, 2013.

[123] MATLAB, *version 7.11.0 (R2010b)*. Natick, Massachusetts: The MathWorks Inc., 2010.

[124] McDonald, R., "Reply to 'autoinputs and model size effects'." Cart3D User's Group Post, May 2012. `https://groups.google.com/forum/#!topic/cart3d/CR-JeFOPH2s`, accessed April 30, 2013.

[125] McGrew, T., Alspector-Kelly, M., and Allhoff, F., *The Philosophy of Science: An Historical Anthology*. Malden, MA: Blackwell Publishing, 2009.

[126] McKay, M., Beckman, R., and Conover, W., "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, pp. 239–245, May 1979.

[127] McKee, K., Street, M., Remillard, C., and Hellman, B., "Advancing ORS technologies and capabilities with a space tourist suborbital vehicle," in *AIAA Space 2009 Conference*.

[128] Melkumyan, A. and Ramos, F., "A sparse covariance function for exact Gaussian process inference in large datasets," in *21st International Joint Conference on Artificial Intelligence*, Association for the Advancement of Artificial Intelligence, 2009.

[129] Mendenhall, M., Chou, H., and Love, J., "Computational aerodynamic design and analysis of launch vehicles," in *38th Aerospace Sciences Meeting & Exhibit*, AIAA, January 2000. AIAA-2000-0385.

[130] Mendenhall, M., Lesieutre, D., Caruso, S., Dillenius, M., and Kuhn, G., "Aerodynamic design of Pegasus: Concept to flight with computational fluid dynamics," *Journal of Spacecraft and Rockets*, vol. 31, pp. 1007–1015, 1994.

[131] Mendenhall, M., Lesieutre, D., Whittaker, C., Curry, R., and Moulton, B., "Aerodynamic analysis of Pegasus – Computations vs reality," in *31st AIAA Aerospace Sciences Meeting & Exhibit*, 1993. AIAA 93-0520.

[132] Morris, M. and Mitchell, T., "Exploratory designs for computational experiments," *Journal of Statistical Planning and Inference*, vol. 43, no. 3, pp. 381–402, 1995.

[133] Muylaert, J., Walpot, L., Rostand, P., Rapuc, M., Brauckmann, G., Paulson, J., Trockmorton, D., and Weimuenster, K., "Extrapolation from wind tunnel to flight: Shuttle orbiter aerodynamics," tech. rep., Advisory Group for Aerospace Research and Development, 1998. AGARD-AR-319 Volume 2.

[134] Myers, R., Montgomery, D., and Anderson-Cook, C., *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. Wiley Series in Probability and Statistics, Hoboken, NJ: John Wiley & Sons, Inc., third edition ed., 2009.

[135] Nemec, M., "Reply to 'adjoint sometimes doesn't refine mesh'." Cart3D User's Group Post, February 2011. `https://groups.google.com/forum/#!topic/cart3d/-wHlXsIonZQ`, accessed May 6, 2013.

[136] Neumaier, A., *Introduction to Numerical Analysis*. Cambridge University Press, 2001. pg. 82.

[137] Nichols, R. and Buning, P., *User's Manual for OVERFLOW 2.1, v2.1t*, 2008.

[138] Oberkampf, W. and Roy, C., *Verification and Validation in Scientific Computing*. Cambridge University Press, 2010.

[139] Oberkampf, W. and Trucano, T., "Validation methodology in computational fluid dynamics," in *AIAA Fluids*, 2000. AIAA 2000–2549, SAND2000–1656C.

[140] Oberkampf, W. and Trucano, T., "Verification & validation benchmarks," tech. rep., Sandia National Laboratory, 2007. SAND2007–0853.

[141] Oberkampf, W., Trucano, T., and Hirsch, C., "Verification, validation and predictive capability in computational engineering and physics," in *Foundations for Verification and Validation in the 21st Century Workshop*, 2002.

[142] PACE, "Pacelab suite." Published via company website, December 2011. `http://www.pace.de/fileadmin/user_upload/dokumente/Resources/PacelabSuite_Whitepaper.pdf` accessed April 19, 2013.

[143] Pamadi, B., Brauckmann, G., Ruth, M., and Fuhrmann, H., "Aerodynamic characteristics, database development, and flight simulation of the X-34 vehicle," *Journal of Spacecraft and Rockets*, vol. 38, no. 3, pp. 334–344, 2001.

[144] Pamadi, B., Covell, P., Tartabini, P., and Murphy, K., "Aerodynamic characteristics and glide-back performance of Langley Glide-Back Booster," tech. rep., NASA Langley Research Center, August 2004.

[145] Pebesma, E., "Multivariable geostatistics in S: the GSTAT package," *Computers & Geosciences*, vol. 30, no. 7, pp. 683–691, 2004.

[146] Pedersen, M. and Chipperfield, A., "Local unimodal sampling," tech. rep., Hvass Laboratories, 2008. Technical Report HL0801.

[147] Pepelyshev, A. and Oakley, J., "On the choice of correlation function and cross-validation for Gaussian processes," tech. rep., Project for Managing Uncertainty in Complex Models, July 2009.

[148] Picard, R. and Cook, R., "Cross-validation of regression models," *Journal of the American Statistical Association*, vol. 79, pp. 575–583, September 1984.

[149] Picheny, V., Ginsbourger, D., Roustant, O., Haftka, R., and Kim, N., "Adaptive designs of experiments for accurate approximation of a target region," *Journal of Mechanical Design*, vol. 132, July 2010.

[150] Post, M. L., Dobransky, M., Shackelford, W., and Hellman, B., "Experimental investigation of a suborbital reusable booster concept with canards," in *49th AIAA Aerospace Sciences Meeting*, (Orlando, FL), January 2011. AIAA 2011–1304.

[151] Qian, P. Z., Tang, B., and Wu, C. J., "Nested space-filling designs for computer experiments with two levels of accuracy," *Statistica Sinica*, vol. 19, pp. 287–300, 2009.

[152] Qian, P., "Nested Latin hypercubes," *Biometrika*, vol. 96, no. 4, 2009.

[153] Qian, P., "Sliced Latin hypercube designs," *Journal of the American Statistical Association*, vol. 107, no. 497, pp. 393–399, 2012.

[154] Qian, P. and Wu, C., "Bayesian hierarchical modeling for integrating low-accuracy and high-accuracy experiments," *Technometrics*, vol. 50, no. 2, pp. 192–204, 2008.

[155] Qian, Z., Seepersad, C., Joseph, V., Allen, J., and Wu, C., "Building surrogate models based on detailed and approximate simulations," *Journal of Mechanical Design*, vol. 128, 2006.

[156] Queipo, V., Haftka, R., Shyy, W., Goel, T., Vaidyanathan, R., and Tucker, P., "Surrogate-based analysis and optimization," *Progress in Aerospace Sciences*, vol. 41, pp. 1–28, 2005.

[157] Quiñonero-Candela, J. and Rasmussen, C., "A unifying view of sparse approximate Gaussian process regression," *Journal of Machine Learning Research*, pp. 1939–1959, December 2005.

[158] Raczynski, C., *A Methodology for Comprehensive Strategic Planning and Program Prioritization*. PhD thesis, Georgia Institute of Technology, August 2008. pgs. 75-77.

[159] Ranjan, P., Bingham, D., and Michaildis, G., "Sequential experimental design for contour estimation from complex computer codes," *Technometrics*, vol. 50, November 2008.

[160] Rasmussen, C. and Williams, C., *Gaussian Processes for Machine Learning*. the MIT Press, 2006.

[161] Raymer, D., *Aircraft Design: a Conceptual Approach*. American Institute of Aeronautics and Astronautics, 1999.

[162] Raymer, D., *Enhancing Aircraft Conceptual Design using Multidisciplinary Optimization*. PhD thesis, Royal Institute of Technology, Stockholm, Sweden, 2002.

[163] Redeker, G., "A selection of experimental test cases for the validation of CFD codes," tech. rep., North Atlantic Treaty Organization Advisory Group for Aerospace Research & Development, 1994. AGARD–AR–303 Vol. 1.

[164] Remy, N., Boucher, A., and Wu, J., *Applied Geostatistics with SGeMS: A User's Guide*. New York: Cambridge University Press, 2009. pg. 12.

[165] Richtmyer, R. and Morton, K., *Difference Methods for Initial-Value Problems*. Malabar, FL: Krieger Publishing Company, 2nd ed., 1994. pg. 101.

[166] Risse, K., Anton, E., and Henke, R., "Methodology for flying qualities prediction and assessment in preliminary aircraft design," in *10th AIAA Aviation Technology, Integration, and Opeartions (ATIO) Conference*, September 2010. AIAA 2010-9261.

[167] Robinson, T., Eldred, M., Wilcox, K., and Haimes, R., "Surrogate-based optimization using multifidelity models with variable parameterization and corrected space mapping," *AIAA Journal*, vol. 46, November 2008.

[168] Rodriguez, D., "Reply to 'flowcart running problem'." Cart3D User's Group Post, February 2012. `https://groups.google.com/forum/#!topic/cart3d/A4V-WCNMUNg`, accessed April 30, 2013.

[169] Sacher, P., Perrier, P., Raymer, D., Haines, B., Hoeijmakers, H., Lamar, J., and Boppe, C., "Special course on engineering methods in aerodynamic analysis and design of aircraft," tech. rep., Advisory Group for Aerospace Research & Development (AGARD) Report 783, 1991.

[170] Sacks, J., Welch, W., Mitchell, T., and Wynn, H., "Design and analysis of computer experiments," *Statistical Science*, vol. 4, no. 4, pp. 409–435, 1989.

[171] Scharl, J. and Mavris, D., "Building parametric and probabilistic dynamic vehicle models using neural networks," in *AIAA Modeling and Simulation Conference*.

[172] Scharl, J., Mavris, D., and Burdun, I., "Use of flight simulation in early design: Formulation and application of the virtual testing and evaluation methodology," in *2000 SAE/AIAA World Aviation Conference*, 2000. AIAA 2000–5590.

[173] Schonlau, M., Welch, W., and Jones, D., "Global versus local search in constrained optimization of computer models," *Lecture Notes-Monograph Series*, vol. 34, 1998. New Developments and Appliations in Experimental Design.

[174] Shan, S. and Wang, G., "Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions," *Journal of Structural and Multidisciplinary Optimization*, vol. 41, no. 2, pp. 219–241.

[175] Sharma, J., Vidal, L., Douglas, R., and Mavris, D., "Development of an environment for determining vehicle mass properties of a reusable booster design," in *AIAA Space 2011 Conference & Exposition*, (Long Beach, California), September 2011. AIAA 2011–7165.

[176] Sippel, M. and Klevanski, J., "Preliminary definition of an aerodynamic configuration for a reusable booster stage within tight geometric constraints," in *Proceedings of the Fifth European Symposium on Aerothermodynamics for Space Vehicles*, 2005. ESA SP-563.

[177] Smith, J., "Could Air Force Space Command benefit from commercial space companies like SpaceX,XCOR,Virgin Galactic, and Bigelow Aerospace?," in *AIAA Space 2011 Conference*, (Long Beach, CA), September 2011. AIAA 2011–7171.

[178] Smith, S., "Reply to 'autoinputs and model size effects'." Cart3D User's Group Post, May 2012. `https://groups.google.com/forum/#!topic/cart3d/CR-JeFOPH2s`, accessed April 30, 2013.

[179] Snelson, E. and Ghahramani, Z., "Variable noise and dimensionality reduction for sparse Gaussian processes," in *Proceedings of the 22nd Conference in Uncertainty in Artificial Intelligence*, (Cambridge, MA), 2006.

[180] Sobol, I., "On the distribution of points in a cube and the approximate evaluation of integrals," *USSR Computational Mathematics and Mathematical Physics*, 1967.

[181] Sova, G., Divan, P., and Spacht, L., "Aerodynamic Preliminary Analysis System II part II - user's manual," tech. rep., North American Aircraft Operations, Rockwell International, 1991. NASA CR-182077.

[182] Stein, A. and Corsten, L., "Universal Kriging and cokriging as regression procedures," *Biometrics*, vol. 47, no. 2, pp. 575–587, 1991.

[183] Talay, T., *Introduction to the Aerodynamics of Flight*. Scientific and Technical Information Office, National Aeronautics and Space Administration, 1975. `http://history.nasa.gov/SP-367/cover367.htm` accessed April 3, 2012.

[184] Toal, D. and Keane, A., "Efficient multipoint aerodynamic design optimization via cokriging," *Journal of Aircraft*, vol. 48, no. 5, 2011.

[185] Torczon, V. and Trosset, M., "Using approximations to accelerate engineering design optimization," in *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, vol. 2, pp. 738–748. AIAA 98–4800.

[186] Turner, C. J. and Campbell, M. I., "Generic sequential sampling for metamodel approximations," tech. rep., Los Alamos National Laboratory, 2003.

[187] Viana, F. and Haftka, R., "Cross validation can estimate how well prediction variance correlates with error," *AIAA Journal*, vol. 47, pp. 2266–2270, September 2009.

[188] Waggoner, E. e. a., "A selection of experimental test cases for the validation of CFD codes volume I," tech. rep., Advisory Group for Aerospace Research & Development (AGARD) Report 303, 1994.

[189] Wang, G. G. and Shan, S., "Review of metamodeling techniques in support of engineering design optimization," *Journal of Mechanical Design*, vol. 129, pp. 370–380, May 2006.

[190] Weil, J. and Powers, B., "Correlation of predicted and flight derived stability and control derivatives – with particular application to tailless delta wing configurations," Tech. Rep. NASA TM-81361, Dryden Flight Research Center, NASA, 1981.

[191] Wells, J., "When effective theories predict: The inevitability of Mercury's anomalous perihelion precession," June 2011. Philosophy of Physics Lecture delivered at the University of Michigan.

[192] Werner-Westphal, C., Heinze, W., and Horst, P., "Multidisciplinary integrated preliminary design applied to unconventional aircraft configurations," *Journal of Aircraft*, vol. 45, March-April 2008.

[193] Wu, C. and Hamada, M., *Experiments: Planning, Analysis, and Parameter Design Optimization*. John Wiley & Sons, Inc., 2000.

[194] Xiong, Y., Chen, W., and Tsui, K., "A new variable-fidelity optimization framework based on model fusion and objective-oriented sequential sampling," *Journal of Mechanical Design*, vol. 130, 2008.

[195] Yamazaki, W. and Mavriplis, D., "Derivative-enhanced variable fidelity surrogate modeling for aerodynamic functions," in *49th AIAA Aerospace Sciences Meeeting*, 2011. AIAA 2011-1172.

[196] Yang, X.-S., *Engineering Optimizations: An Introduction with Metaheuristic Applications*. Hoboken, NJ: John Wiley & Sons, 2010. pg. 263.

[197] Yin, J., Ng, S., and Ng, K., "Kriging model with modified nugget effect for random simulation with heterogeneous variances," in *Industrial Engineering and Engineering Management, 2008. IEEM 2008. IEEE International Conference on*, pp. 1714–1718, IEEE, 2008.

[198] Young, J. and Underwood, J., "Development of aerodynamic uncertainties for the space shuttle orbiter," *Journal of Spacecraft*, vol. 20, no. 6, pp. 513–517, 1983.

[199] Young, J., Underwood, J., Hillje, E., Whitnah, A., Romere, P., Gamble, J., Roberts, B., Ware, G., Scallion, W., Spencer Jr., B., Arrington, J., and Olsen, D., "The aerodynamic challenge of the design and development of the space shuttle orbiter," in *Space Shuttle Technical Conference – Part I* (Chaffee, N., ed.), pp. 209–263, 1985. NASA CP-2342.

[200] "Yubico Frequently Asked Questions." `http://www.yubico.com/support/faq/` accessed April 3, 2013.

[201] Zhang, F., ed., *The Schur Complement and its Applications*, vol. 4 of *Numerical Methods and Algorithms*. Springer, 2005. pg. 19.

[202] Zhang, S., Zhu, P., Chen, W., and Arendt, P., "Concurrent treatment of parametric uncertainty and metamodeling uncertainty in robust design," *Structural and Multidisciplinary Optimization*, vol. 47, no. 1, pp. 63–76, 2013.

[203] Zhao, L., Choi, K., and Lee, I., "Metamodeling method using dynamic Kriging for design optimization," *AIAA Journal*, vol. 49, pp. 2034–2046, September 2011.