UNCERTAINTY QUANTIFICATION AND OPTIMIZATION UNDER UNCERTAINTY USING SURROGATE MODELS

Thesis

Submitted to

The School of Engineering of the

UNIVERSITY OF DAYTON

In Partial Fulfillment of the Requirements for

The Degree of

Master of Science in Aerospace Engineering

By

Komahan Boopathy

UNIVERSITY OF DAYTON

Dayton, Ohio

May, 2014

UNCERTAINTY QUANTIFICATION AND OPTIMIZATION UNDER UNCERTAINTY USING SURROGATE MODELS

Name: Boopathy, Komahan

APPROVED BY:

Markus P. Rumpfkeil, Ph.D.
Advisor Committee Chairman
Assistant Professor, Department of Mechanical and Aerospace Engineering

Raymond M. Kolonay, Ph.D.
Committee Member
Adjunct Professor, Department of Mechanical and Aerospace Engineering

Aaron Altman, Ph.D.
Committee Member
Associate Professor, Department of Mechanical and Aerospace Engineering

John G. Weber, Ph.D.
Associate Dean for Graduate Studies
School of Engineering

Tony E. Saliba, Ph.D.
Dean, School of Engineering
& Wilke Distinguished Professor

ABSTRACT

UNCERTAINTY QUANTIFICATION AND OPTIMIZATION UNDER UNCERTAINTY USING SURROGATE MODELS

Name: Boopathy, Komahan
University of Dayton

Advisor: Dr. Markus P. Rumpfkeil

Surrogate models are widely used as approximations to exact functions that are computationally expensive to evaluate. The choice of model training information and the estimation of the accuracy of surrogate models are major research avenues. In this work, a unified dynamic framework for surrogate model training point selection and error estimation is proposed. Building auxiliary local surrogate models over sub-domains of the global surrogate model forms the basis of the framework. A *discrepancy function*, defined as the absolute difference between response predictions from *global* and *local* surrogate models for randomly chosen test candidates, drives the framework.

The framework preferably evaluates the expensive exact function at locations, where the value of the discrepancy function is high and when a distance-constraint to previously existing training points are satisfied. As a result, the surrogate model is continually refined in regions of higher uncertainty in prediction, and a better spread of training points is also achieved. Unlike most training point selection approaches, the framework addresses surrogate training from two disparate contexts, as training in the *presence* and *absence*

of derivative information. The local surrogate models use the derivative information when available and affect the framework via the discrepancy function, and helps determine the locations that require derivative information. The benefits of the dynamic training approach are demonstrated with analytical test functions and the construction of a two-dimensional aerodynamic database. The results show that the proposed method improves the convergence monotonicity and produces more accurate surrogate models, when compared to random and quasi-random training point selection strategies.

The newly introduced *discrepancy function* is proposed as an approximation to the actual error in the prediction of the surrogate model leading to the quantities: *root mean square discrepancy* (RMSD) and *maximum absolute discrepancy* (MAD). The results demonstrate a close agreement of RMSD and MAE with the actual root mean square error (RMSE) and maximum absolute error (MAE), respectively. Therefore, RMSD and MAD are proposed as measures for the accuracy of the surrogate models in applications of practical interest. The benefit of surrogate validation comes without warranting any additional exact function evaluations, which makes the framework computationally viable.

Multivariate interpolation and regression model is employed to build local surrogates, whereas the kriging and polynomial chaos expansions serve as global surrogate models. This demonstrates the applicability of the proposed framework to any surrogate model with an open choice of training data selection.

Finally, the dynamically trained surrogate models are applied to uncertainty quantifications and optimizations under mixed epistemic and aleatory uncertainties (OUU), for structural and aerodynamic test cases. In the OUUs epistemic uncertainties are propagated via box-constrained optimizations, whereas the aleatory uncertainties are propagated via

inexpensive sampling of the surrogate models. The structural test cases include designing a three-bar truss and a cantilever beam, whereas the aerodynamic test case involves the robust optimization (lift-constrained drag minimization) of an airfoil under steady flow conditions.

Dedicated to my family and friends

PUBLICATIONS

Referred Journal Articles

K. Boopathy and M.P. Rumpfkeil, "A Unified Framework for Training Point Selection and Error Estimation for Surrogate Models", AIAA Journal. Accepted.

Conference Proceedings

K. Boopathy and M.P. Rumpfkeil, "Robust Optimizations of Structural and Aerodynamic Designs", 15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Atlanta, June 2014. Accepted.

K. Boopathy and M.P. Rumpfkeil, "A Multivariate Interpolation and Regression Enhanced Kriging Surrogate Model", 21st AIAA Computational Fluid Dynamics Conference, San Diego, June 2013. AIAA Paper 2013-2964.

K. Boopathy and M.P. Rumpfkeil,"Building Aerodynamic Databases Using Enhanced Kriging Surrogate Models", AIAA Region III Student Conference, Chicago, April 2013. Won second price for best student paper.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor **Dr. Markus P. Rumpfkeil** who has been a source of knowledge, inspiration and motivation throughout my life as a scholar at the University of Dayton. I greatly appreciate all the time that he has spent on personal guidance, research discussions, proof reading of publications, and so on, over the past two years.

I also wish to thank my thesis committee members: **Dr. Raymond M. Kolonay** and **Dr. Aaron Altman**. Dr. Kolonay's course on "Structural Optimization" provided me with plenty of inspiration and helped me to understand the realms of optimization as well as to identify test problems for the current and future research.

I am always grateful and indebted to my mother **Jegathambal**, my love **Sivadarsini** and my family members for their unwavering support. I would like to extend a warm appreciation to my friends and lab mates: **Sathyaram Balasubramani**, **Sidaard Gunasekaran**, **Santosh Kumar** and **Kevin Wabick** for their continued patronage.

I also thank **Wataru Yamazaki** for his Kriging surrogate model, **Qiqi Wang** for his MIR model and **Karthik Mani** for his Eulerian flow and adjoint solver.

I also acknowledge the partial financial support from the Office for Graduate Academic Affairs through the Graduate Student Summer Fellowship Program.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

α	angle of attack
$\boldsymbol{\beta}$	epistemic realizations
$\boldsymbol{\alpha}$	aleatory realizations
$\boldsymbol{\nabla}\widehat{f}$	surrogate approximated gradients
$\boldsymbol{d}$	vector of design variables
$\delta(\boldsymbol{\xi})$	discrepancy function at a test location
$\epsilon(\boldsymbol{\xi})$	actual error at a test location
μ	mean
$\boldsymbol{\eta}$	epistemic random variables
ψ	orthogonal basis function
$\boldsymbol{R}$	correlation matrix
$\boldsymbol{r}$	correlation vector
σ	standard deviation
σ^2	variance

$\boldsymbol{u}$	polynomial chaos coefficients or weights
$\widetilde{N}$	number of Monte Carlo samples
$\boldsymbol{x}$	training point location for the surrogate model
$\boldsymbol{x}^{*(j)}$	closest training point to j-th test candidate
$\boldsymbol{\xi}$	aleatory random variables/test candidate location
c	control parameter for distance-constraint
C_D	drag coefficient
C_L	lift coefficient
f	exact function value or simulation output
f^*	extremum of the function within the specified bounds
F_s	factor of safety
g	inequality constraint
g_i^*	extremum of the constraints within the specified bounds
M	number of variables or dimensions
M_∞	Mach number
N	number of training points for the global surrogate
n	number of simulation output evaluations for box-constrained optimization
N_{cyc}	number of training points selected per cycle
N_{init}	number of training points for the initial surrogate

N_{local} number of training points for the local surrogate

N_{test} number of test candidates

$p(\boldsymbol{\xi})$ probability distribution of the random variables

P_k probability of constraint satisfaction

R spatial correlation function

T number of terms in polynomial chaos expansion

$\boldsymbol{\nabla}^2 \widehat{f}$ surrogate approximated Hessian

$\mathcal{J}$ objective function

$\rho^{(j)}$ distance between j-th test candidate and its closest training point

$\widehat{f}$ surrogate approximated function value

$\widehat{f}_{global}$ response value from the global surrogate

$\widehat{f}_{local}$ response value from the local surrogate

CHAPTER I

INTRODUCTION AND MOTIVATION

Numerical simulations are extensively used in science and engineering research to solve real world problems whose theoretical solutions are undetermined and in cases where it is impractical to conduct experiments. For example, analytical solution for the general Navier-Stokes equations is unknown and it is impractical to conduct experiments on vehicles operating in hostile and unpredictable environmental conditions (e.g. reentry vehicles). Under such circumstances computational methods are very powerful in assisting the design process and they play a pivotal role in design improvements and reductions in development cost and time.

In recent years, owing to an increasing availability of computational resources and sophisticated algorithms, numerical methods have been extensively used in engineering research and development. However, in spite of several advancements in computer hardware (e.g. CPU speed) and the increasing use of parallel computing (also known as high performance computing), a striking imbalance still exists between the requirements and availability of computational power. For example, the number of required mesh points for direct numerical simulations (DNS) scales with the Reynolds number as $Re^{9/4}$, but the current state-of-the-art computing can only support on the order of 10^9 - 10^{10} mesh points and is thus limited

to Reynolds numbers of about $10,000$. Similarly, other high-fidelity physics-based simulations such as large eddy simulations (LES) and unsteady Reynolds-averaged Navier-Stokes (RANS) computations, also require significant amounts of computational resources and time.

A simple gradient-based airfoil shape optimization requires many optimizer iterations and hence flow solutions. The entire flow field needs to be solved for at each design iteration and typically the gradient also needs to be computed (through finite-difference, adjoint [1–3] or other techniques), which in total can require hundreds of flow solutions, potentially demanding enormous computational time and storage. Therefore, the designer has to trade-off accuracy versus computational time or limit the design spaces in scope, which can lead to inefficient designs. Such a computational burden explains the need for alternatives to expensive high-fidelity simulations, at least when function evaluations are repeatedly required.

1.1 Surrogate Models

In order to reduce the computational burden from multiple simulations, surrogate models (also called response surfaces or metamodels) have been employed by the research community over the past few decades. Surrogate models provide an "approximate and inexpensive to evaluate" representation of an output quantity of interest as a function of input variables. A surrogate model, being an approximate representation of the original function space, is bound to have an approximation error. A lot of today's surrogate model research is directed on improving the accuracy of existing models as well as developing versatile and robust surrogates [4–8] considering its wide range of applications. Some popular applications are discussed in the following paragraphs.

1.1.1 Applications of Surrogate Models

In general, a surrogate model is used to approximate or predict a quantity of interest based on available computational, experimental or statistical data, and hence finds applications in a wide array of disciplines. Some popular engineering applications are reviewed as follows.

1.1.1.1 Database of Expensive Functions

In early stages of a design process it is desirable to rapidly evaluate several design configurations. These analyses, that can either be physical experiments or numerical simulations, are expensive and time-consuming. As an example, in the design of aerospace systems one of the main requirements is the accurate and efficient prediction of force and moment coefficients of the vehicle. This information is required for different trial designs, and an optimization procedure requires many design iterations (trials), which can be very expensive and prolong the time span of the whole design process. Moreover, design of complex systems such as airplanes are highly coupled with several disciplines (e.g. propulsion, aerodynamics, structures, controls, etc.) demanding many such optimization processes. Due to these time-intensive operations, developing a successful design and its associated production can take a long development time. Hence, design teams are shifting their focus on building databases that could aid in the design process. Some examples are:

- NASA's *Constellation program* has been working on a Heavy Lift Launch Vehicle (HLLV) named Ares V, where a simulation protocol was developed to generate databases of the aerodynamic force and moment coefficients for HLLV ascent [9].

- The NASA Langley Research Center was involved in the development of a preflight aerodynamic database of the X-34 Reusable Launch Vehicle (RLV) [10], covering the

entire range of Mach numbers, angles of attack, side-slip and control surface deflections anticipated in the complete flight envelope.

- The Center for Computer Applications in Aerospace Science and Engineering (a part of DLR) is working towards a "Digital Flight" (full flight simulation), for which they develop databases covering the entire flight envelope for many different aircraft configurations.

For such applications, surrogate models can be employed to predict the exact function values, thereby providing a computational advantage through an effective reduction of the overall number of required exact simulations.

1.1.1.2 Global Optimization

Gradient-based optimization algorithms typically converge to a local optimum and the final optimum is highly dependent on the starting point in the design space. There is no guarantee that the gradient-based optimizers lead to the global optimum unless the convexity of the objective function is guaranteed. An illustration of a multi-modal design space with several local optimum is provided in Figure 1.1. On the other hand, global optimization strategies such as the genetic algorithms (GA), which are based on an extensive search of the design space, are computationally intensive particularly when applied to problems involving high-fidelity physics-based simulations. A surrogate model can be employed to alleviate some of the computational burden in this process. For example, when a surrogate model is constructed with the given training data, the most promising locations in the model can be explored for relatively lower computational cost. To aid in this, an initial global surrogate model is constructed using the available training data. The surrogate model is then sampled extensively to find the potential global optimum and the model is locally

updated or refined until reaching the optimum. Even for such optimization applications,

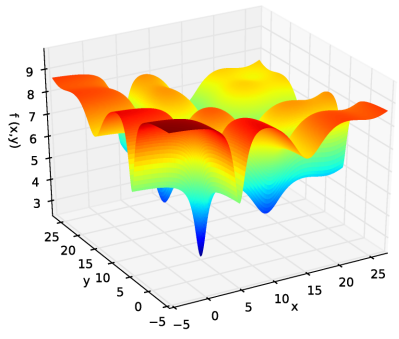


Figure 1.1: A two variable multi-modal objective function.

the construction of an accurate initial global surrogate model provokes a higher likelihood of locating the global optimum during subsequent local refinements of the surrogate model, when compared to a poor initial surrogate that fails to capture the real behavior leading to a premature convergence.

1.1.1.3 Uncertainty Quantification and Optimization Under Uncertainty

A deterministic optimization approach assumes no variations in the design variables and other parameters. This can easily lead to sub-optimal performance or failure of many deterministically optimized designs. For instance, when an aircraft designed to cruise at specific optimal settings (e.g. Mach number, angle of attack, shape) deviates from these settings (e.g. due to continuous wind gusts, ice accumulation, faulty calibration of instruments, wear and tear) the flight performance can be adversely affected leading to an increased fuel burn or other undesirable characteristics. Therefore, given the uncertainties in input variables, parameters and operating environments, it is expected to have some measure of confidence

placed on the output quantities of interest (e.g. weight, thrust, lift etc.), giving rise to the field of uncertainty quantification (UQ) and optimization under uncertainty (OUU). Optimization under uncertainty (also known as stochastic optimization) is an important avenue in computational research owing to the rising need for robust and reliable designs, where the key objective is to quantify uncertainties and account for them in the regular optimization process. OUU can be subdivided into two main fields namely: *robust design optimization* (RDO) and *reliability based design optimization* (RBDO) [11–13]. RDO techniques can be used to produce a design that is more robust (less sensitive) to design variable and/or operating environment changes, whereas RBDO minimizes the probability of failure of the system.

In recent years, design teams and regulatory agencies are increasingly being asked to specifically characterize and quantify different types of uncertainties and separate their individual effects [14–18]. The most popular and easiest approach for the propagation of uncertainties is the Monte Carlo simulation (MCS), where the simulation output f is sampled many times to obtain output statistics and to determine worst case scenarios. However, multiple realizations of the output function f are not always computationally tractable (e.g. for high-fidelity physics-based simulations such as computational fluid dynamics (CFD) or finite element analyses (FEA)). To overcome this problem, a surrogate model can be constructed to model the uncertainties, which can then be sampled inexpensively and exhaustively to propagate the uncertainties and determine output statistics. This approach is referred to as the inexpensive Monte Carlo simulation (IMCS) [19]. More details on these topics are provided in chapter VIII.

1.1.2 Research Avenues

Major research areas which are extensively pursued related to surrogate modeling are discussed in the following paragraphs.

1.1.2.1 Choice of Training Points

The accuracy of a surrogate model is influenced primarily by the non-linearity of the function to be modeled and by the choice of training point locations. Training point selection is typically done by using design of experiments (DoE) techniques such as uniform design (UD) [20]. Many other methods, which have been originally developed to approximate multi-dimensional integrals, are also being used for training data selection for surrogate models: these include Monte Carlo (MC) [21], latin hypercube (LHS) [22], quadrature nodes [23, 24], and low-discrepancy sequences [25]. These strategies typically tend to suffer from deficiencies, such as exponential growth in the number of required points with dimensionality (e.g. quadrature nodes), missing important regions by chance (e.g. LHS, MC), poor and correlated distribution of training points in higher dimensions (e.g. Sobol [26] and Halton [25] sequences), etc. Apart from having been developed for a different purpose, all these strategies are domain-based, i.e. they do not take into account function values or their non-linearities and sensitivities. Thus, plenty of research has been conducted to address the aforementioned problems and to develop adaptive strategies which consider response values and similar criteria such as *expected improvement* and *mean squared error* [27–29]. The appropriate choice of training points is an important open research question as recently pointed out by Roderick *et al.* [30] and Cheng *et al.* [31].

1.1.2.2 Surrogate Model Approximation Error

Theoretically, the approximation error associated with surrogate models can be quantified by comparison with the exact function values, by means of computable quantities such as root mean square error (L_2-norm) and maximum absolute error (L_∞-norm). However, in real-life applications, it is computationally impractical to calculate any of these quantities, since they require too many expensive exact function evaluations. Without having a measure for the accuracy of surrogate models, the validity of application results involving surrogate models become highly questionable. In this work a kriging model which minimizes the expected mean squared error (MSE) is employed. MSE can simply be used to estimate the approximation error, however, in practice MSE is really much more a measure of how well the training points fill the design space uniformly than an actual model approximation error. Other validation methods such as split sampling, cross-validation, bootstrapping and Akaike's information criterion (AIC) [32] either provide limited information regarding the accuracy of surrogates, or require additional exact function evaluations [33]. Recently, Mehmani *et al.* [28] proposed a framework for the regional error estimation of surrogates (REES), but the authors do not show how it compares to actual approximation errors. This portrays a continuous evolution of methods for surrogate validation.

1.1.2.3 Curse of Dimensionality

The exponential rise in the required amount of training data for the surrogate model as the number of input variables, M, increase is referred to as the "curse of dimensionality". To address this problem, the introduction of higher-order derivative information (gradients and Hessian) within surrogate models, as additional training data, has attracted a lot of attention. For example, many gradient-enhanced surrogate models have been developed

and have shown very beneficial results [8, 34, 35]. This is mainly due to the availability of computationally efficient and accurate gradient evaluation methods such as the adjoint formulation [3, 36]. Perhaps the easiest to implement and most popular method for obtaining derivatives is the finite-difference method which is highly sensitive to the choice of step-size as well as computationally expensive with an increasing number of inputs. The adjoint method [1–3], on the other hand, provides a more efficient means of calculating the derivative and is more accurate as well, particularly when targeting a single output objective: the effort needed to compute the full gradient is comparable to the effort needed to compute the function itself. It is appealing to use function values and their derivative information for the construction of surrogate models, since there are $M + 1$ pieces of information available for the constant cost of roughly two function evaluations using adjoint techniques. However, a direct differentiation would be computationally more efficient when the number of constraints is larger than the number of design variables (as in most structural optimization problems).

The ability to compute second-order sensitivity derivatives is also highly desirable for many science and engineering simulation problems [19, 37–39]. For example, the availability of Hessian information allows the use of much stronger Newton optimization strategies, which holds the potential for greatly reducing the expense of solving difficult optimization problems. An efficient Hessian evaluation method has been developed by Rumpfkeil and Mavriplis [40, 41] and using the same logic as above, it is also very appealing to utilize Hessian information within surrogate models in addition to the gradient information. The Hessian provides $M \cdot (M + 1)/2$ pieces of information for roughly the cost of M function evaluations, since, in general, the most efficient full Hessian constructions require the solution of M forward linear problems (one corresponding to each input parameter) [19, 37, 41].

In summary, when using gradient and Hessian information along with function values to train the surrogate model, the expensive exact function is expected to be computed far fewer times, or alternatively it supplies additional "free" training information to the surrogate, which either ways help to reduce the "curse of dimensionality".

Another promising avenue is the use of variable-fidelity surrogate modeling, where the computational burden is reduced by employing a larger amount of low-fidelity data (e.g. Euler evaluations) in conjunction with a smaller amount of high-fidelity data (e.g. Navier-Stokes evaluations) [8]. This indeed offers a great potential for savings since, for example, Euler evaluations are 50–100 times cheaper to obtain compared to equivalent RANS evaluations [42].

1.2 Research Objectives

The research objectives of this work are outlined as follows:

1. To develop a training point selection framework for surrogate models that will provide the user with a better choice of training points than conventional methods. This will be studied as: (*i*) selection in the absence of derivative information (function values only) and (*ii*) selection in the presence of derivative information (function, gradient and Hessian values),

2. To propose quantities for surrogate model error estimate that can be used in problems of practical interest to assess the accuracy of surrogate models,

3. To apply and demonstrate the training point selection and error estimation framework on kriging and polynomial chaos surrogate models which are widely used within the research community,

4. To advance gradient-enhanced polynomial chaos to Hessian-enhanced polynomial chaos methods,

5. To compare the performances of kriging and polynomial chaos surrogate models on analytical and aerodynamic test problems in terms of model accuracy,

6. To apply the improved surrogate models to uncertainty quantification and optimization under uncertainty (mixed epistemic/aleatory) on structural and aerodynamic test problems.

1.3 Thesis Organization

The organization of this thesis is as follows. Chapter II reviews the three surrogate modeling approaches that are used in this work: kriging, polynomial chaos and multivariate interpolation and regression. Reviews of different training point selection methods are given in chapter III. Chapter IV provides brief accounts on existing methods for surrogate model validation. Chapter V details the proposed dynamic training point selection and error estimation algorithm. Chapter VI presents the results implementing the proposed training point selection and error estimation framework with kriging and polynomial chaos on analytical test functions. Chapter VII details the construction of an aerodynamic database of drag and lift coefficients with kriging and polynomial chaos, and discusses the performance of the proposed framework for this aerodynamic test problem. Chapter VIII details the methods for uncertainty quantification and optimization under mixed epistemic and aleatory uncertainties. This is followed by robust optimization results for structural and aerodynamic designs in chapter IX. Chapter X summarizes the major conclusions and outlines future research directions.

CHAPTER II

SURROGATE MODELS

The computational cost associated with the use of high-fidelity physics-based simulation models pose a serious impediment to successful application of optimization algorithms [43]. In many engineering problems, thousands of function evaluations may be required to locate an optimal solution. Therefore, when expensive high-fidelity simulations are employed, the naive application of optimization algorithms can demand exorbitant number of exact simulations. The computational requirement is even higher when uncertainties need to be quantified and accounted in the optimization process.

In the 1960s numerical optimization was limited to perhaps fifty variables and was computationally very expensive [44]. In the 1970s, Schmit *et al.* [45] introduced the concept of a sequential approximate optimization to improve the efficiency of prevalent structural optimization practices. The basic idea was to analyze an initial design to generate data that could be used to construct approximations (local) to the objective function and constraints and supply them to the optimizer. Appropriate move limits were applied to ensure the validity of these approximations. After solving this approximate optimization problem, the models are updated and the process is continued until suitable convergence criteria are met. Subsequently, more general approximation techniques (local and global) have been developed and it is now increasingly commonplace to employ computationally cheap

approximation models in lieu of exact calculations. For a detailed understanding of the evolution of approximation schemes, the reader is referred to the work of Schmit [46].

Several methods exist in the scientific literature for the purpose of modeling the input – output relationship of an exact simulation [43]. This chapter aims to describe the three surrogate modeling approaches employed in this work: (1) kriging [47], (2) polynomial chaos expansions [23] and (3) multivariate interpolation and regression [48, 49].

2.1 Kriging Surrogate Model

The kriging surrogate model was originally developed in the field of geostatistics by a South African mining engineer Danie G. Krige [6]. Kriging was introduced in engineering design following the work of Sacks *et al.* [50] and has found a lot of aerospace applications [34, 35, 51, 52]. Kriging predicts the function value by using stochastic processes and has the flexibility to represent multi-modal and non-smooth functions. A detailed mathematical background for the kriging surrogate models is provided in the upcoming paragraphs. This review is extracted from works in the literature such as Han *et al.* [53], Rosenbaum *et al.* [27, 54], Yamazaki *et al.* [55, 56] and Jones [57].

2.1.1 Original Kriging Model

2.1.1.1 Model Construction

Universal Kriging

The formulation of "universal kriging model" is given by,

$$\widehat{f}(x) = \sum_{k=1}^{m} \mu_k f_k(x) + Z(x), \tag{2.1}$$

where $\widehat{f}$ is the approximated function value. The first term is a lower-order polynomial of order m modeling the mean behavior using regression and $Z(x)$ represents a local variation from the mean behavior modeled as a stochastic process.

Ordinary Kriging

The simplest model called "ordinary kriging" has the following formulation [11],

$$\widehat{f}(x) = \mu + Z(x), \tag{2.2}$$

where μ is an unknown constant term modeling the mean behavior. In both formulations $Z(x)$ is assumed to be a stationary Gaussian random process satisfying:

$$\mathrm{E}[Z(x)] = 0 \tag{2.3}$$

and

$$\mathrm{Cov}[Z(x^{(i)}), Z(x^{(j)})] = \sigma^2 R(x^{(i)}, x^{(j)}), \tag{2.4}$$

where R is the user defined spatial correlation function.

The following mathematical discussions are based on *ordinary kriging* (constant mean term) for simplicity and an extension to *universal kriging* is straight-forward.

2.1.1.2 Model Training

Let $\boldsymbol{X} = \{x^{(1)}, x^{(2)}, \ldots, x^{(N)}\}$ be a vector of N training point locations, where the exact function $f(x)$ is evaluated leading to a vector of training data denoted as $\boldsymbol{F} = \{f^{(1)}, f^{(2)}, \ldots, f^{(N)}\}$.

Correlation Parameters

The training of the kriging model occurs through correlation parameters as follows.

1. **Correlation matrix:** Kriging uses a spatial correlation function to relate stochastic variables between the observed training points, that is, the available data $\boldsymbol{F}$ is correlated through a correlation matrix $\boldsymbol{R}$. Hence the size of the resulting correlation

matrix is $N \times N$. The correlation matrix takes the form,

$$\boldsymbol{R} = \begin{bmatrix} R(x^{(1)}, x^{(1)}) & \cdots & R(x^{(1)}, x^{(N)}) \\ \vdots & \ddots & \vdots \\ R(x^{(N)}, x^{(1)}) & \cdots & R(x^{(N)}, x^{(N)}) \end{bmatrix}. \tag{2.5}$$

2. **Correlation vector:** A correlation vector, $\boldsymbol{r}$, represents the correlation between an untried point x and observed training points $x^{(i)}$ and can be written as,

$$\boldsymbol{r} = \begin{bmatrix} R(x^{(1)}, x) \\ \vdots \\ R(x^{(N)}, x) \end{bmatrix}. \tag{2.6}$$

3. **Correlation function:** The construction of correlation matrix and vector requires a user-specified correlation function.This correlation function, R, determines the nature of the data fitting by the kriging model. Any function which renders the correlation matrix positive definite can be used as the correlation function, and this is one of the greatest flexibility of kriging. Some popular correlation functions are listed in Table 2.1. The correlation function is dependent on the spatial distance between the

Table 2.1: Popular spatial correlation functions for the kriging surrogate model.

Basis Function	$R(x^{(i)}, x^{(j)})$	
Exponential	$\exp(-\theta\|x^{(i)} - x^{(j)}\|^p)$	
Gaussian	$\exp(-\theta\|x^{(i)} - x^{(j)}\|^2)$	
Cubic Spline	$1 - 6\xi^2 + 6\xi^3$	for $0 \le \xi < 0.5$
	$2(1-\xi)^3$	for $0.5 \le \xi < 1$
	0	for $\xi \ge 1$
	where $\xi = \theta\|x^{(i)} - x^{(j)}\|$	
Wendland C^2	$\frac{(1-\xi)^4(4\xi+1)}{3}$	for $0 \le \xi \le 1$
	0	for $\xi > 1$
	where $\xi = \theta\|x^{(i)} - x^{(j)}\|$	
Wendland C^4	$\frac{(1-\xi)^6(35\xi^2+18\xi+3)}{3}$	for $0 \le \xi \le 1$
	0	for $\xi > 1$
	where $\xi = \theta\|x^{(i)} - x^{(j)}\|$	

given points $x^{(i)}$ and $x^{(j)}$ and a few hyper-parameters (denoted as Θ) that may be introduced by the chosen correlation function.

Maximum Likelihood Estimation

The available training data $\boldsymbol{F}$ is assumed to be realizations of a normally distributed random variable $f(x)$ with mean μ and variance σ^2, that is, $f \sim \mathcal{N}(\mu, \sigma^2)$. In the process of fitting the training data a *maximum likelihood estimation (MLE)* is carried out, that tunes the unknown parameters of the random process (the mean and variance) as well as the hyper-parameters, Θ, introduced by the correlation function (e.g. p and θ by an exponential), to maximize the *probability density function* (also termed as likelihood function) of the assumed Gaussian random process:

$$L(\mu, \sigma^2, \Theta) = \frac{1}{(2\pi\sigma^2)^{N/2}|\boldsymbol{R}|^{1/2}} \exp\left[\frac{-(\boldsymbol{F} - \boldsymbol{I}\mu)^T \boldsymbol{R}^{-1}(\boldsymbol{F} - \boldsymbol{I}\mu)}{2\sigma^2}\right]. \tag{2.7}$$

Here, $\boldsymbol{I}$ is a column vector of ones since a constant μ is used. Taking natural logarithm on both sides yields:

$$\ln\{L(\mu, \sigma^2, \Theta)\} = -\frac{N}{2}\ln(2\pi\sigma^2) - \frac{1}{2}\ln(|\boldsymbol{R}|) - \frac{(\boldsymbol{F} - \boldsymbol{I}\mu)^T \boldsymbol{R}^{-1}(\boldsymbol{F} - \boldsymbol{I}\mu)}{2\sigma^2}. \tag{2.8}$$

This form of the likelihood function is known as the ln-likelihood function and is preferred over the original representation for stability reasons [58]. Now, by using the necessary condition for optimality [11], the optimal estimations of mean,

$$\mu^*(\Theta) = \frac{\boldsymbol{I}^T \boldsymbol{R}^{-1} \boldsymbol{F}}{\boldsymbol{I}^T \boldsymbol{R}^{-1} \boldsymbol{I}}, \tag{2.9}$$

and variance of the random process,

$$\sigma^{2^*}(\Theta) = \frac{(\boldsymbol{F} - \boldsymbol{I}\mu)^T \boldsymbol{R}^{-1}(\boldsymbol{F} - \boldsymbol{I}\mu)}{N}, \tag{2.10}$$

can be obtained analytically, but they still depend on the hyper-parameters Θ influencing the spatial correlation function and thereby the correlation matrix $\boldsymbol{R}$. However, the problem

is now reduced to finding the optimum hyper-parameters Θ. By substituting Eqns. (2.9) and (2.10) into Eq. (2.8) and by ignoring constant values (since the optimum is not affected) a simplified ln-likelihood function (objective function) can be obtained:

$$\ln\{L(\Theta)\} \approx -\frac{N\ln(\sigma^{2^*}) - \ln(|\boldsymbol{R}|)}{2}, \tag{2.11}$$

which in turn has to be maximized to find the optimum hyper-parameters Θ. The values of μ^* and σ^{2^*} corresponding to the optimum hyper-parameters can then be found using Eqns. (2.9) and (2.10). Once these parameters are determined for a particular set of training data, the kriging model can be used to predict the function value at any given point x which is described in more detail next.

2.1.1.3 Model Prediction

Once the correlation parameters are determined, kriging predicts the function value at any given location x by maximizing the likelihood of "observed data and prediction" by means of an augmented system similar to the one in Eq. (2.8) which takes the simplified form (again ignoring constants),

$$\ln(L) \approx -\frac{\begin{pmatrix} \boldsymbol{F} - \boldsymbol{I}\mu^* \\ \widehat{f} - \mu^* \end{pmatrix}^T \begin{bmatrix} \boldsymbol{R} & \boldsymbol{r} \\ \boldsymbol{r}^T & 1 \end{bmatrix}^{-1} \begin{pmatrix} \boldsymbol{F} - \boldsymbol{I}\mu^* \\ \widehat{f} - \mu^* \end{pmatrix}}{2\sigma^{2^*}}. \tag{2.12}$$

By differentiating,

$$\frac{\partial \ln(L)}{\partial \widehat{f}} = -\frac{(\widehat{f} - \mu^*)}{\sigma^{2^*}(1 - \boldsymbol{r}^T\boldsymbol{R}^{-1}\boldsymbol{r})} + \frac{\boldsymbol{r}^T\boldsymbol{R}^{-1}(\boldsymbol{F} - \boldsymbol{I}\mu^*)}{\sigma^{2^*}(1 - \boldsymbol{r}^T\boldsymbol{R}^{-1}\boldsymbol{r})} = 0. \tag{2.13}$$

The only unknown quantity in Eq. (2.13) is the kriging predicted function value $\widehat{f}$. Thus by rearranging, the kriging prediction at a given point x can be mathematically written as:

$$\widehat{f}(x) = \mu^* + \boldsymbol{r}^T\boldsymbol{R}^{-1}(\boldsymbol{F} - \boldsymbol{I}\mu^*), \tag{2.14}$$

where μ^* is the generalized least-squares estimator of the term μ in Eq. 2.2 modeling the mean behavior. The factor $\boldsymbol{R}^{-1}(\boldsymbol{F} - \boldsymbol{I}\mu^*)$ can be considered as a *weighting vector* $\boldsymbol{w}$ and hence the predicted response is a weighted sum of the correlation vector $\boldsymbol{r}$, such that,

$$\widehat{f}(x) = \mu^* + \boldsymbol{r}^T \boldsymbol{w}. \tag{2.15}$$

2.1.1.4 Mean Squared Error

The kriging model provides an error estimate (uncertainty bound) for its predictions in terms of a mean squared error (MSE) which can be written as:

$$\begin{aligned} \mathrm{MSE}[\widehat{f}(x)] &= \sigma^{2^*} \left[1 - \begin{pmatrix} \boldsymbol{r} \\ 1 \end{pmatrix}^T \begin{bmatrix} \boldsymbol{R} & \boldsymbol{I} \\ \boldsymbol{I}^T & 0 \end{bmatrix}^{-1} \begin{pmatrix} \boldsymbol{r} \\ 1 \end{pmatrix} \right] \\ &= \sigma^{2^*} \left[1 - \boldsymbol{r}^T \boldsymbol{R}^{-1} \boldsymbol{r} + \frac{(1 - \boldsymbol{r}^T \boldsymbol{R}^{-1} \boldsymbol{r})^2}{\boldsymbol{I}^T \boldsymbol{R}^{-1} \boldsymbol{I}} \right]. \end{aligned} \tag{2.16}$$

MSE is zero at training point locations and increases with the spatial distance from training points. Thus, MSE is more of a measure of space filling than the actual error in the approximation and it seldom matches the actual error in the model.

2.1.1.5 Expected Improvement

An expected improvement function (EI) quantifies the likely improvement in the objective function (when using kriging for optimization) in considering to evaluate the exact function at a trial location x. It takes the form:

$$\mathrm{EI}(x) = (f_{\min} - \widehat{f}(x)) \phi \left(\frac{f_{\min} - \widehat{f}(x)}{\sigma^*(x)} \right) + \sigma^*(x) \varphi \left(\frac{f_{\min} - \widehat{f}(x)}{\sigma^*(x)} \right) \tag{2.17}$$

Here, f_{min} refers to the minimum among available training data $\boldsymbol{F}$, ϕ is the Gaussian density function and φ is the Gaussian distribution function. The first term favors "exploitation" of high confidence regions, whereas the second term favors "exploration" of regions that have high uncertainty. Thus, EI can be seen as a figure of merit that balances local and global search for the optimum.

2.1.1.6 Summary

A short summary of the steps involved in constructing a kriging surrogate is provided as follows [54]:

1. Choose a design of experiments strategy for $\boldsymbol{X} = \{x^{(1)}, x^{(2)}, \ldots, x^{(N)}\}$ and evaluate the exact function values to obtain $\boldsymbol{F(X)} = \{f^{(1)}, f^{(2)}, \ldots, f^{(N)}\}$.

2. Pick a spatial correlation function R.

3. Find the unknown model and hyper-parameters using a maximum likelihood estimation approach.

4. Solve the kriging linear system to find the weighting vector $\boldsymbol{w}$.

5. Evaluate the model at a given location x as $\widehat{f}(x) = \mu^* + \boldsymbol{r}^T\boldsymbol{w}$. Additional information such as *mean squared error* and *expected improvement* can also be obtained.

2.1.2 Cokriging Models

Cokriging models are kriging models which also include derivative values of the exact function (e.g. gradients, Hessian) in their formulations. Both, direct and indirect cokriging models have been developed and have shown to provide beneficial results [8, 47, 56].

2.1.2.1 Indirect Cokriging

In the indirect cokriging approach additional training points are constructed around an actual training point by using Taylor series extrapolations. The kriging is then constructed using all available data points (actual and extrapolated). This approach is highly dependent on extrapolation step sizes specified by the user which can either cause ill-conditioning of

the correlation matrix if chosen too small or give poor Taylor series approximations if chosen too large.

2.1.2.2 Direct Cokriging

In the direct cokriging approach, the covariances between gradient and/or Hessian information are directly included in the correlation matrix and are modeled by differentiating the correlation function. The direct approach is preferable due to its lack of tunable parameters and better condition numbers of the correlation matrix. The correlation matrix for a gradient-enhanced kriging model takes the form,

$$\boldsymbol{R} = \begin{bmatrix} R(x^{(1)},x^{(1)}) & \cdots & R(x^{(1)},x^{(N)}) & \frac{\partial R(x^{(1)},x^{(1)})}{\partial x^{(1)}} & \cdots & \frac{\partial R(x^{(1)},x^{(N)})}{\partial x^{(N)}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ R(x^{(N)},x^{(1)}) & \cdots & R(x^{(N)},x^{(N)}) & \frac{\partial R(x^{(N)},x^{(1)})}{\partial x^{(1)}} & \cdots & \frac{\partial R(x^{(N)},x^{(N)})}{\partial x^{(N)}} \\ \frac{\partial R(x^{(1)},x^{(1)})}{\partial x^{(1)}} & \cdots & \frac{\partial R(x^{(1)},x^{(N)})}{\partial x^{(N)}} & \frac{\partial^2 R(x^{(1)},x^{(1)})}{\partial^2 x^{(1)}} & \cdots & \frac{\partial^2 R(x^{(1)},x^{(N)})}{\partial x^{(1)}\partial x^{(N)}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial R(x^{(N)},x^{(1)})}{\partial x^{(N)}} & \cdots & \frac{\partial R(x^{(N)},x^{(N)})}{\partial x^{(N)}} & \frac{\partial^2 R(x^{(N)},x^{(1)})}{\partial x^{(N)}\partial x^{(1)}} & \cdots & \frac{\partial^2 R(x^{(N)},x^{(N)})}{\partial^2 x^{(N)}} \end{bmatrix}. \quad (2.18)$$

The correlation vector takes the form,

$$\boldsymbol{r} = \begin{bmatrix} R(x^{(1)},x) \\ \vdots \\ R(x^{(N)},x) \\ \frac{\partial R(x^{(1)},x)}{\partial x^{(1)}} \\ \vdots \\ \frac{\partial R(x^{(N)},x)}{\partial x^{(N)}} \end{bmatrix}. \quad (2.19)$$

The correlation matrix for Hessian-enhanced direct cokriging models are not shown here due to the complexity of its representation. The size of the correlation matrix increases to $N \cdot (1 + M)$ when gradient-enhanced and $N \cdot (1 + M + \frac{M(M+1)}{2}^2)$ when Hessian-enhanced.

2.1.3 Variable-Fidelity Kriging Model

It is known that the computational expenses with physics-based simulations can be very high. Even the reduced simulation requirements by surrogate models can become

hard to obtain, especially when the number of input variables (dimensions) increases. A promising avenue is the use of variable-fidelity (also known as multi-fidelity) surrogate modeling. The general idea is to combine trends from low-fidelity data (e.g., coarser meshes, less sophisticated models) with interpolations of high-fidelity data (e.g., finer meshes, better models, experimental data). For example, low-fidelity data from Euler evaluations can be combined with a smaller amount of high-fidelity data from Navier-Stokes evaluations. This indeed offers a great potential for savings since, for example, Euler evaluations are 50–100 times cheaper to obtain compared to equivalent RANS evaluations [42]. The low-fidelity trends are connected with high-fidelity data by using a bridge function (also known as connection or scaling function) that can be multiplicative, additive or a hybrid (both multiplicative and additive) [53].

Variable-fidelity kriging surrogate models have been introduced and have shown to provide a better computational efficiency compared to regular kriging surrogate models. The reader is referred to Han *et al.* [53] and Yamazaki *et al.* [8, 55] for more mathematical background on this subject.

2.2 Polynomial Chaos Expansions

Polynomial chaos refers to polynomial representation of uncertainty (chaos), where the stochastic quantities of the random process (e.g. mean and variance) are represented as spectral expansions of orthogonal polynomials. It can be classified into *stochastic Galerkin* (intrusive) and *stochastic collocation* (non-intrusive) methods. A spectral expansion of the solution f dependent on multidimensional random variable $\boldsymbol{x}$ takes the form,

$$f(\boldsymbol{x}) = \sum_{k=0}^{\infty} u_k \psi_k(\boldsymbol{x}), \tag{2.20}$$

where $\psi(\boldsymbol{x})$ denotes the selected basis function.

2.2.1 Original Polynomial Chaos

In the seminal work of Wiener [4], a spectral expansion of Hermite polynomials for Gaussian random variables were used to represent certain stochastic processes, where a second-order random process admits a chaos representation of the following form [24]:

$$\begin{aligned} f(\boldsymbol{x}) = u_0 H_0 + \sum_{i_1=1}^{M} u_{i_1} H_1(x_{i_1}) + \sum_{i_1=1}^{M} \sum_{i_2=1}^{i_1} u_{i_1 i_2} H_2(x_{i_1}, x_{i_2}) \\ + \sum_{i_1=1}^{M} \sum_{i_2=1}^{i_1} \sum_{i_3=1}^{i_2} u_{i_1 i_2 i_3} H_3(x_{i_1}, x_{i_2}, x_{i_3}) + \dots . \end{aligned} \tag{2.21}$$

The basis is formed by Hermite polynomials H_p of order p and $x_{ij},\ j = 1, 2, \dots, M$ is the set of multidimensional Gaussian random variables. The Hermite chaos expansion has been an effective tool for solving stochastic differential equations with Gaussian normal inputs [5, 23, 59]. However, Cameron and Martin [60] showed that the original chaos expansion is applicable to any functional and will converge in a mean-square sense (or L_2 sense). The original chaos expansion has also been used to solve inputs with a log-normal distribution [61].

2.2.2 Generalized Polynomial Chaos

The Wiener-Askey scheme of polynomials [61,62] include various orthogonal polynomials whose weighting functions are identical to the probability density function (PDF) of the distributions of the random variable $\boldsymbol{x}$. Each type of polynomial in the Wiener-Askey scheme corresponds to a particular probability distribution as shown in Table 2.2 [63]. Hermite polynomials used in the original polynomial chaos is a subset of the Wiener-Askey scheme of polynomials. The correct choice of random distribution polynomial as the basis function ψ is shown to exhibit optimal convergence rates [59].

Table 2.2: Wiener-Askey scheme polynomials with their corresponding weighing functions.

Distribution	PDF	Polynomial	Weight	Range
Normal	$\frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}}$	Hermite $H_p(x)$	$e^{\frac{-x^2}{2}}$	$[-\infty, \infty]$
Uniform	$\frac{1}{2}$	Legendre $P_p(x)$	1	$[-1, 1]$
Exponential	e^{-x}	Laguerre $L_p(x)$	e^{-x}	$[0, \infty]$
Beta	$\frac{(1-x)^\alpha (1+x)^\beta}{2^{\alpha+\beta+1}\ B(\alpha+1, \beta+1)}$	Jacobi $J_p^{\alpha,\ \beta}(x)$	$(1-x)^\alpha (1+x)^\beta$	$[-1, 1]$
Gamma	$\frac{x^\alpha e^{-x}}{\Gamma(\alpha+1)}$	Generalized Laguerre $L_p^\alpha(x)$	$x^\alpha e^{-x}$	$[0, \infty]$

Xiu and Karniadakis [59] advanced the original (also called Hermite) polynomial chaos into a generalized polynomial chaos or Wiener-Askey polynomial chaos, where the random variable $\boldsymbol{x}$ can be associated with different measures. The generalized polynomial chaos expansion (PCE) is given as follows,

$$\begin{aligned} f(\boldsymbol{x}) = u_0\psi_0 + \sum_{i_1=1}^{M} u_{i_1}\psi_1(x_{i_1}) + \sum_{i_1=1}^{M}\sum_{i_2=1}^{i_1} u_{i_1 i_2}\psi_2(x_{i_1}, x_{i_2}) \\ + \sum_{i_1=1}^{M}\sum_{i_2=1}^{i_1}\sum_{i_3=1}^{i_2} u_{i_1 i_2 i_3}\psi_3(x_{i_1}, x_{i_2}, x_{i_3}) + \ldots, \end{aligned} \tag{2.22}$$

where ψ_p is the polynomial basis of order p taken from the Wiener-Askey scheme depending on the probability distribution of multidimensional random variables $x_{i_j},\ j = 1, 2, \ldots, M$.

Multidimensional Basis Functions

Multidimensional orthogonal polynomial bases can be easily obtained by tensorization of the corresponding one-dimensional polynomial bases using a multi-index notation [23,24,64],

$$\psi_k(\boldsymbol{x}) = \ \boldsymbol{\psi}_k(x_1, x_2, \ldots, x_M) = \prod_{i=1}^{M} \psi_{\alpha_i^k}(x_i), \tag{2.23}$$

where the multi-index α_i^k denotes the order of the one-dimensional polynomial. As an example, a two-variable chaos expansion is shown below:

$$\begin{aligned} f(x_1, x_2) &= u_0\psi_0(x_1)\psi_0(x_2) \\ &+ u_1\psi_1(x_1)\psi_0(x_2) + u_2\psi_0(x_1)\psi_1(x_2) \\ &+ u_{11}\psi_2(x_1)\psi_0(x_2) + u_{21}\psi_1(x_1)\psi_1(x_2) + u_{22}\psi_0(x_1)\psi_2(x_2) \\ &+ u_{111}\psi_3(x_1)\psi_0(x_2) + u_{211}\psi_2(x_1)\psi_1(x_2) + u_{221}\psi_1(x_1)\psi_2(x_2) + u_{222}\psi_0(x_1)\psi_3(x_2) \\ &+ u_{1111}\psi_4(x_1)\psi_0(x_2) + \ldots, \end{aligned} \tag{2.24}$$

where $\boldsymbol{u} = \{u_0, u_1, u_2, u_{11}, u_{22}, \ldots\}$ is the vector of coefficients.

A Compact and Truncated Form

For simplicity of notation a compact form of the expansion in Eq. (2.22) given by:

$$f(\boldsymbol{x}) \;=\; \sum_{k=0}^{\infty} u_k\psi_k(\boldsymbol{x}), \tag{2.25}$$

will be used throughout, where $\psi_k(\boldsymbol{x})$ is defined by Eq. 2.23. For computational purposes, the infinite expansion in Eq. (2.25) can be truncated and represented as,

$$\widehat{f}(\boldsymbol{x}) \;=\; \sum_{k=0}^{P} u_k\psi_k(\boldsymbol{x}), \tag{2.26}$$

where the total number of terms T is given by [59],

$$T = P + 1 = \frac{(M+p)!}{M!p!}. \tag{2.27}$$

2.2.3 Intrusive and Non-intrusive forms of PCE

The computation of polynomial chaos coefficients (weights) $\boldsymbol{u}$ falls into two categories as intrusive and non-intrusive methods [65]. This work is focused on non-intrusive methods and the reader is referred to the literature for a more detailed review on intrusive methods than given here [23, 24, 64].

Intrusive Methods

Intrusive methods, also referred to as the stochastic Galerkin (SG) methods require the formulation and solution of the stochastic version of the original model. The deterministic code has to be altered through the substitution of polynomial chaos expansions. A Galerkin projection in random space is applied to derive the equations in weak form, which generates a system of coupled state equations. The expansion coefficients $\boldsymbol{u}$ can then be obtained by solving the linear system of state equations [23, 64]. The requirement to modify existing source code quite extensively can make the implementation of intrusive PCE difficult or even impossible.

Non-intrusive Methods

Non-intrusive methods, also referred to as the stochastic collocation (SC), use interpolation methods and projections of a set of deterministic simulations onto a polynomial basis [66]. In doing so, the residual of the governing equations are required to be zero at discrete nodes in the domain which are called collocation points [23]. The weights $\boldsymbol{u}$ of the multivariate spectral expansions in Eq. (2.25) are computed by practices that do not mandate the modification of existing deterministic solvers and they can be used as a black-box. Similar to SG methods, SC methods achieve fast convergence when the solutions possess sufficient smoothness in the random space. Stochastic collocation can be further classified as *pseudo-spectral integration* or *response surface method* (also known as point collocation, linear regression methods) [23,67]. In the pseudo-spectral approach, the inputs $\boldsymbol{x}$ are chosen deterministically at quadrature nodes [23, 24, 64] or sparse grids. In the response surface approach, the inputs can be evaluated at random locations $\boldsymbol{x}$ and the coefficients $\boldsymbol{u}$ are obtained by solving the linear system formed by Eq. (2.25). The non-intrusive approach

(polynomial regression) developed by Roderick *et al.* [30] is adopted in this work and is discussed in more detail in the following paragraphs. The random collocation points are referred to as training points in this work.

2.2.4 Constructing Polynomial Chaos Response Surface

A regression procedure for obtaining the best polynomial approximation in the presence of function, gradient, and Hessian information is discussed below.

1. **Setting up the linear system:** Polynomial fitting conditions are enforced at the training points $\boldsymbol{x}^{(j)},\ j = 1, 2, \ldots, N$, resulting in a linear system with N equations and T unknowns. It is required that $N \geq T$ to be able to solve the linear system. Legendre orthogonal polynomials are employed as the basis function in this work.

$$\begin{bmatrix} \psi_0(\boldsymbol{x}^{(1)}) & \psi_1(\boldsymbol{x}^{(1)}) & \cdots & \psi_P(\boldsymbol{x}^{(1)}) \\ \psi_0(\boldsymbol{x}^{(2)}) & \psi_1(\boldsymbol{x}^{(2)}) & \cdots & \psi_P(\boldsymbol{x}^{(2)}) \\ \vdots & \vdots & \ddots & \vdots \\ \psi_0(\boldsymbol{x}^{(N)}) & \psi_1(\boldsymbol{x}^{(N)}) & \cdots & \psi_P(\boldsymbol{x}^{(N)}) \end{bmatrix} \begin{Bmatrix} u_0 \\ u_1 \\ \vdots \\ u_P \end{Bmatrix} = \begin{Bmatrix} f(\boldsymbol{x}^{(1)}) \\ f(\boldsymbol{x}^{(2)}) \\ \vdots \\ f(\boldsymbol{x}^{(N)}) \end{Bmatrix} \tag{2.28}$$

2. **Solving the linear system:** The linear system given in Eq. (2.28) enforces polynomial interpolation at the N training points, when the amount of available data is equal to the number of coefficients to solve for (*i.e.,* $N = T$). To improve the conditioning of the basis matrix, an oversampling factor of two (smallest integer greater than one) is recommended in the literature [63] and is adopted in this work as well. When oversampling (*i.e.,* $N > T$) is enforced, the linear system can only be solved in a least-squares sense and the resulting response surface is a regression model.

3. **Evaluating the response surface:** Once the coefficients $\boldsymbol{u}$ are solved for, the PCE response surface defined by Eq. (2.26) is successfully obtained. It can now be used to get the approximated function value at any given location $\boldsymbol{x}$, and the surrogate model

is ready for its potential applications such as uncertainty quantification and optimization. In addition, gradient $\boldsymbol{\nabla}\widehat{f}$ and Hessian approximations $\boldsymbol{\nabla}^2\widehat{f}$ from the PCE surrogate model at any location $\boldsymbol{x}$ are readily obtained by differentiating Eq. (2.26) which can facilitate, for example, Newton-based optimization strategies. Mathematically,

$$\boldsymbol{\nabla}\widehat{f} = \sum_{k=0}^{P} u_k \frac{\partial \psi_k(\boldsymbol{x})}{\partial \boldsymbol{x}} \tag{2.29}$$

and

$$\boldsymbol{\nabla}^2\widehat{f} = \sum_{k=0}^{P} u_k \frac{\partial^2 \psi_k(\boldsymbol{x})}{\partial^2 \boldsymbol{x}}, \tag{2.30}$$

where $\frac{\partial \psi_k(\boldsymbol{x})}{\partial \boldsymbol{x}}$ and $\frac{\partial^2 \psi_k(\boldsymbol{x})}{\partial^2 \boldsymbol{x}}$ involve again the multi-index notation given by Eq. 2.23.

Table 2.3: Selected orthogonal polynomials with their first and second derivatives and recursive relation.

Order	Hermite $H_p(x)$	Legendre $P_p(x)$	Hermite $\frac{\partial H_p(x)}{\partial x}$	Legendre $\frac{\partial P_p(x)}{\partial x}$	Hermite $\frac{\partial^2 H_p(x)}{\partial^2 x}$	Legendre $\frac{\partial^2 P_p(x)}{\partial^2 x}$
0	1	1	0	0	0	0
1	x	x	1	1	0	0
2	x^2-1	$\frac{3x^2-1}{2}$	$2x$	$3x$	2	3
3	x^3-3x	$\frac{5x^3-3x}{2}$	$3x^2-3$	$\frac{15x^2-3}{2}$	$6x$	$15x$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
p	$xH_{p-1}(x)$ $-(p-1)H_{p-2}(x)$	$\frac{(2p-1)xP_{p-1}(x)}{p}$ $-\frac{(p-1)P_{p-2}(x)}{p}$	$pH_{p-1}(x)$	$-\frac{(p+1)xP_p(x)}{x^2-1}$ $+\frac{(p+1)P_{p+1}(x)}{x^2-1}$	$p(p-1)H_{p-2}$	$\frac{(p+1)[(p+2)x^2+1]P_p(x)}{(x^2-1)^2}$ $-\frac{(p+1)(2p+5)xP_{p+1}(x)}{(x^2-1)^2}$ $+\frac{(p+1)(p+2)P_{p+2}(x)}{(x^2-1)^2}$

2.2.5 Enhancing PCE with Derivative Information

The polynomial chaos has been enhanced to incorporate gradient information which has shown promising results to reduce the curse of dimensionality (see Roderick *et al.* [30, 68]). The authors have named their approach *polynomial regression with derivative information*

(PRD). As a straight-forward extension here, it will be enhanced with Hessian information as well. The required first and second derivatives of the orthogonal polynomials ψ are shown in Table 2.3.

2.2.5.1 Gradient-Enhanced Polynomial Chaos

When the linear system in Eq. (2.28) is augmented with gradient information, each row of the basis matrix gives rise to M additional rows, where M is the number of components in $\boldsymbol{x}$. Thus, if there are N training points the size of the basis matrix becomes $N' \times T$, where $N' = N \cdot (1 + M)$.

$$\begin{bmatrix} \begin{pmatrix} \psi_0(\boldsymbol{x}^{(1)}) & \psi_1(\boldsymbol{x}^{(1)}) & \cdots & \psi_P(\boldsymbol{x}^{(1)}) \\ \frac{\partial \psi_0(\boldsymbol{x}^{(1)})}{\partial x_1} & \frac{\partial \psi_1(\boldsymbol{x}^{(1)})}{\partial x_1} & \cdots & \frac{\partial \psi_P(\boldsymbol{x}^{(1)})}{\partial x_1} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \psi_0(\boldsymbol{x}^{(1)})}{\partial x_M} & \frac{\partial \psi_1(\boldsymbol{x}^{(1)})}{\partial x_M} & \cdots & \frac{\partial \psi_P(\boldsymbol{x}^{(1)})}{\partial x_M} \end{pmatrix} \\ \vdots \end{bmatrix} \begin{Bmatrix} u_0 \\ u_1 \\ \vdots \\ u_P \end{Bmatrix} = \begin{Bmatrix} \begin{pmatrix} f(\boldsymbol{x}^{(1)}) \\ \frac{\partial f(\boldsymbol{x}^{(1)})}{\partial x_1} \\ \vdots \\ \frac{\partial f(\boldsymbol{x}^{(1)})}{\partial x_M} \end{pmatrix} \\ \vdots \end{Bmatrix}$$

2.2.5.2 Hessian-Enhanced Polynomial Chaos

When the linear system is augmented with Hessian information too, the size of the basis matrix becomes $N' \times T$, where $N' = N \cdot (1 + M + \frac{M(M+1)}{2})$

$$\begin{bmatrix} \begin{pmatrix} \psi_0(\boldsymbol{x}^{(1)}) & \psi_1(\boldsymbol{x}^{(1)}) & \cdots & \psi_P(\boldsymbol{x}^{(1)}) \\ \frac{\partial \psi_0(\boldsymbol{x}^{(1)})}{\partial x_1} & \frac{\partial \psi_1(\boldsymbol{x}^{(1)})}{\partial x_1} & \cdots & \frac{\partial \psi_P(\boldsymbol{x}^{(1)})}{\partial x_1} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \psi_0(\boldsymbol{x}^{(1)})}{\partial x_M} & \frac{\partial \psi_1(\boldsymbol{x}^{(1)})}{\partial x_M} & \cdots & \frac{\partial \psi_P(\boldsymbol{x}^{(1)})}{\partial x_M} \\ \frac{\partial^2 \psi_0(\boldsymbol{x}^{(1)})}{\partial^2 x_1} & \frac{\partial^2 \psi_1(\boldsymbol{x}^{(1)})}{\partial^2 x_1} & \cdots & \frac{\partial^2 \psi_P(\boldsymbol{x}^{(1)})}{\partial^2 x_1} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 \psi_0(\boldsymbol{x}^{(1)})}{\partial x_1 \partial x_M} & \frac{\partial^2 \psi_1(\boldsymbol{x}^{(1)})}{\partial x_1 \partial x_M} & \cdots & \frac{\partial^2 \psi_P(\boldsymbol{x}^{(1)})}{\partial x_1 \partial x_M} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 \psi_0(\boldsymbol{x}^{(1)})}{\partial^2 x_M} & \frac{\partial^2 \psi_1(\boldsymbol{x}^{(1)})}{\partial^2 x_M} & \cdots & \frac{\partial^2 \psi_P(\boldsymbol{x}^{(1)})}{\partial^2 x_M} \end{pmatrix} \\ \vdots \end{bmatrix} \begin{Bmatrix} u_0 \\ u_1 \\ \vdots \\ u_P \end{Bmatrix} = \begin{Bmatrix} \begin{pmatrix} f(\boldsymbol{x}^{(1)}) \\ \frac{\partial f(\boldsymbol{x}^{(1)})}{\partial x_1} \\ \vdots \\ \frac{\partial f(\boldsymbol{x}^{(1)})}{\partial x_M} \\ \frac{\partial^2 f(\boldsymbol{x}^{(1)})}{\partial^2 x_1} \\ \vdots \\ \frac{\partial^2 f(\boldsymbol{x}^{(1)})}{\partial x_1 \partial x_M} \\ \vdots \\ \frac{\partial^2 f(\boldsymbol{x}^{(1)})}{\partial^2 x_M} \end{pmatrix} \\ \vdots \end{Bmatrix}$$

The approach (polynomial regression/interpolation) is non-intrusive as only the right hand side of the equation needs function, gradient and Hessian evaluations, and a black box

approach can be used to obtain them. When gradients and Hessian are included, the linear system is generally over-determined (contains more data than needed) and it can be solved only in a least-squares sense resulting in a regression model.

2.3 Multivariate Interpolation and Regression

2.3.1 Mathematical Formulation

Wang *et al.* [48, 49] proposed a multivariate interpolation and regression (MIR) scheme where each data point is represented as a Taylor series expansion, and the higher-order derivatives in the Taylor series are treated as random variables. Mathematically, the exact function, f, in an M-dimensional design space is approximated as [48, 49],

$$\widehat{f}(\boldsymbol{x}) = \sum_{i=1}^{N_v} a_{vi}(\boldsymbol{x})\tilde{f}(\boldsymbol{x}_{vi}) + \sum_{i=1}^{N_g} \boldsymbol{a}_{gi}(\boldsymbol{x})\boldsymbol{\nabla}\tilde{f}(\boldsymbol{x}_{gi}), \tag{2.31}$$

where N_v is the number of function data points and N_g is the number of gradient data points (if used). The approximation coefficients a_{vi} and $\boldsymbol{a}_{gi}$ are then chosen by solving an equality constrained least-squares problem such that $\sum_{i=1}^{N_v} a_{vi} = 1$. $\tilde{f}$ and $\boldsymbol{\nabla}\tilde{f}$ are the function, f, and gradient values, $\boldsymbol{\nabla}f$, added with their corresponding measurement errors σ_{vi} and $\boldsymbol{\sigma}_{gi}$ (if any). The scheme produces an interpolatory response surface when the data points are exact, or a regression model when non-zero measurement errors are associated with the data points.

2.3.2 Parameters of MIR

MIR comes with a few parameters that influence the approximation. They are discussed in the following paragraphs.

2.3.2.1 Magnitude Parameter

The magnitude parameter β affects the solution of the equality constrained least-squares problem in determining the MIR coefficients, but only when measurement errors are present. The magnitude of variation of the target function, β, can be estimated as the standard deviation of available data values as shown below,

$$\beta = \sqrt{\frac{\sum_{i=1}^{N_v}(f(\boldsymbol{x}_{vi}) - \bar{f})^2}{N_v - 1}}, \text{ where } \bar{f} = \sum_{i=1}^{N_v} \frac{f(\boldsymbol{x}_{vi})}{N_v}. \tag{2.32}$$

In the presence of measurement errors, $f(\boldsymbol{x}_{vi})$ are unknown, so it is recommended to use the following equation instead,

$$\beta = \sqrt{\frac{\sum_{i=1}^{N_v}(\tilde{f}(\boldsymbol{x}_{vi}) - \bar{\tilde{f}})^2}{N_v - 1}}, \text{ where } \bar{\tilde{f}} = \sum_{i=1}^{N_v} \frac{\tilde{f}(\boldsymbol{x}_{vi})}{N_v}. \tag{2.33}$$

The value of β reflects the magnitude of variation of the exact function $f(\boldsymbol{x})$ and determines the sensitivity of the approximated function value to available data. When β is small the response surface is less sensitive to available data and looks like a nonlinear regression deviating from the data points by a larger threshold. When β is large the response surface essentially goes through every data point. In this work, the measurement errors are assumed to be absent leading to an interpolatory response surface.

2.3.2.2 Wave Number Parameter

The wave number γ determines the smoothness of the response surface. An optimum wave number is attained when the magnitude of estimated approximation error $|f(\boldsymbol{x}_{vi}) - \widehat{f}(\boldsymbol{x}_{vi})|$ (using a leave-one-out approach described in section 4.2.2.2) and the prediction interval (computed by the scheme) are the same. The upper and lower bounds of gamma are determined from the smallest and largest distance between the data points, that is, $\gamma \in [\gamma_{min}, \gamma_{max}]$. A logarithmic bisection procedure is employed on this interval to find the

actual γ that corresponds to the equality of approximation error and prediction interval. A higher wave number tends to produce a smoother response surface almost like a piecewise lower order polynomial. On the contrary, a lower wave number tends to produce overshoots in the response surface.

2.3.2.3 Taylor Order Parameter

The Taylor order, n, defines the order of the Taylor series expansion employed in MIR. The choice of an optimum Taylor order is difficult as it depends on the function to be modeled as well as its dimensionality. Though it is expected that a higher Taylor order would lead to an improved approximation, round-off errors originating from the solution of the least-squares problem in determining the coefficients, tend to propagate to the approximated function value via Eq. (2.31) and deteriorate its accuracy. Therefore, a higher Taylor order does not always guarantee an improved approximation.

The first two parameters discussed in sections 2.3.2.2 and 2.3.2.1 are computed automatically by the scheme whereas the Taylor order is user specified.

2.3.3 Summary

A brief overview of the MIR approximation scheme is provided as follows:

- **Input of data and parameters:** The training data $(\boldsymbol{x}_{vi}, \tilde{f}(\boldsymbol{x}_{vi}))$ and $(\boldsymbol{x}_{gi}, \boldsymbol{\nabla}\tilde{f}(\boldsymbol{x}_{gi}))$ is gathered along with their corresponding user-specified measurement errors (if any).
- **Computation of parameters:** The user specifies the Taylor order parameter. The remaining two parameters of the scheme, namely the magnitude and wave number, are then determined by the scheme using the input data.

- **Prediction:** For each location $\boldsymbol{x}$ at which the approximated function value is desired, an equality constrained least-squares problem is solved to yield the coefficients a_{vi} and $\boldsymbol{a}_{gi}$. The approximated function value $\widehat{f}(\boldsymbol{x})$ is then given by Eq. (2.31).

2.3.4 Computational Cost and Limitations

The total computational cost involved in building and evaluating the MIR response surface for m arbitrary points (locations where the function is to be predicted) is of the order of $\boldsymbol{O}((K \cdot N_V + m)(N_V^2 + M \cdot N_G)^3))$, where K is the number of required bisection iterations for finding the optimum γ, M is the number of dimensions, and the other parameters are the same as discussed in section 2.3.1. This operation count shows that MIR scheme is computationally more expensive than most existing schemes. Nevertheless, they recommend the scheme for applications requiring a higher accuracy rather than computational efficiency. Wang *et al.* [48, 49] also emphasize the importance of preventing close spacing of training points that can lead to possible linear dependence in matrix operations.

In this work, multivariate interpolation and regression is used only to construct local approximations over the sub-domains of the global surrogate models (kriging and polynomial chaos), and forms an integral part of the proposed framework for training point selection and error estimation. The term "local" signifies the reduced surrogate domain (sub-domain) and its associated training data (the use of closest existing training points). The pertaining discussions on the framework are postponed to chapter V.

CHAPTER III

DESIGN OF EXPERIMENTS

Design of experiments (DoE) are a set of strategies that choose or allocate training point locations for a surrogate model. *Training point selection* is also being termed as *sampling* by many investigators, but *sampling* should only refer to the process of probing the surrogate model to obtain approximated function values. This unfortunate terminology has been adopted by many investigators perhaps due to the use of many sampling techniques for training point selection (e.g. Monte Carlo sampling for random training point selection, latin hypercube sampling for pseudo-random training point selection and so on).

The location and number of training points used to construct the surrogate model is known to have a significant effect on its accuracy. Training point selection approaches (or DoE) can be broadly classified into domain-based and response-based approaches [11]. In domain-based approaches, training points are chosen based on the information available from the domain (e.g. distance between two training points), whereas in response-based approaches, the training points are chosen based on the information provided by the surrogate model (e.g. function values, expected improvement). Domain-based training point selection is either random or based on space-filling concepts that try to fill the domain evenly with training points. It is, in general, not possible for the user to select the number of training points a priori to ensure a given accuracy of a surrogate due to the non-linearity

of most functions of interest. Response-based approaches enhance the efficiency of the training point selection process by using information from the existing metamodel. This is because the user could monitor the progress of the model and choose to stop or extend the training point selection process. The following is a brief outline of some important domain- and response-based approaches used for training point selection found in the literature.

3.1 Domain-Based Approaches

3.1.1 Monte Carlo

Monte Carlo (MC) techniques [21, 69, 70] are perhaps the simplest of all DoE methods. Here, a random number generator is used to select training point locations in the domain. A major drawback of MC is that large areas of the domain may be left unexplored while others may be sampled densely [26, 43, 58].

3.1.2 Latin Hypercube

Latin hypercube sampling (LHS) was introduced by McKay *et al.* [22] for designing computer experiments as an alternative to MC sampling techniques. The basic idea is to divide the range of each variable into N bins of equal probability, which yields N^M bins in the domain, where M is the dimension of the problem. Subsequently, N training points are generated for each variable such that no two values lie in the same bin (as shown in Figure 3.1). The LHS algorithm generates training points in a box-shaped domain as follows [43],

$$x_j^{(i)} = \frac{\pi_j^{(i)} + U_j^{(i)}}{N}, \quad \forall \quad 1 \leq j \leq M, \quad 1 \leq i \leq N, \tag{3.1}$$

where $x_j^{(i)}$ is the $j-$th component of the $i-$th training point, $U \in [0, 1]$ is an uniform random number, and π is an independent random permutation of the sequence of integers $0, 1, \ldots, N-1$. In Figure 3.2 persistent fluctuations in the accuracy (determined by the root

mean square error (RMSE)) of a generic surrogate model built using LHS or MC as DoE method can be noticed. In spite of increasing the number of training points, the RMSE does not decrease, because all these points are picked at random. Thus, a superior strategy for training point selection is required to ensure that the RMSE will reduce when the number of training points increase, and this has been one of the major motivations for this research.

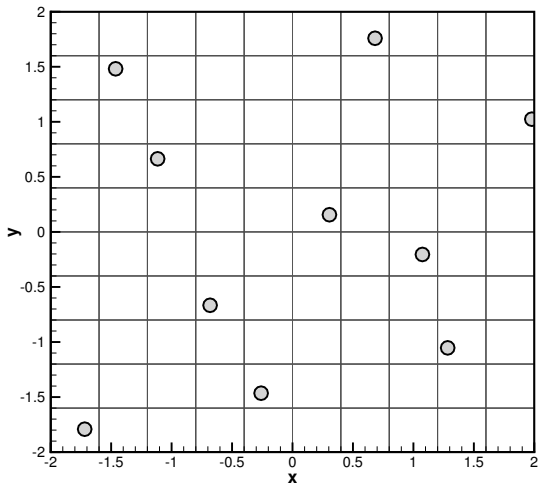


Figure 3.1: An example of LHS in a two-dimensional domain.

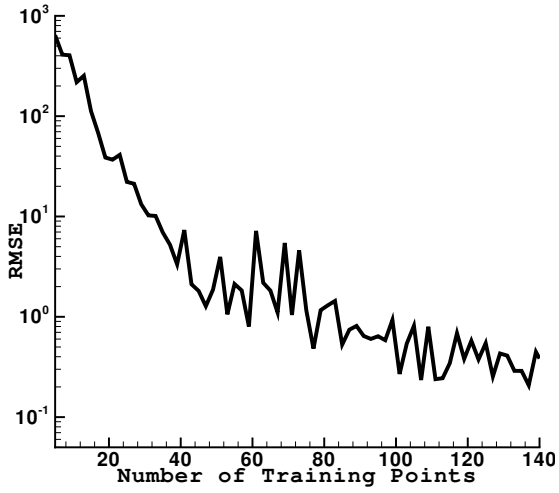


Figure 3.2: Convergence history of a generic surrogate model using LHS or MC.

3.1.3 Delaunay Triangulation

Delaunay triangulation is a geometrical method of training point selection, where the domain is divided into hyper-triangles and the training points are chosen at geometrical significant location such as the centers of the hyper-triangles and midpoints of the edges as shown in Figure 3.3. A major drawback of Delaunay triangulation is that it does not scale well to higher dimensions: the required minimum number of points become quickly large [71] .

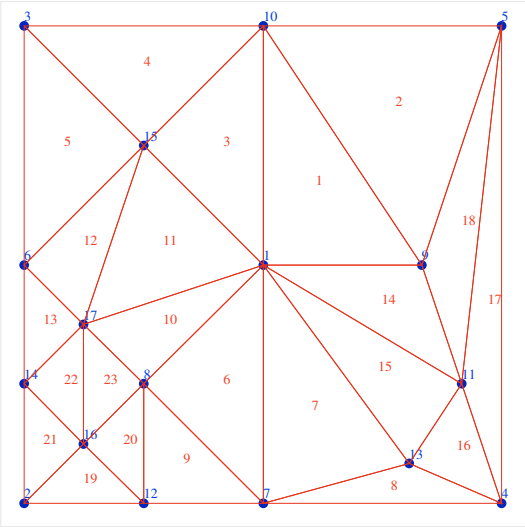


Figure 3.3: A Delaunay triangulation schematic is shown, where points numbered 1 to 5 are the initial training points and 12 points have been added subsequently by splitting the domain into triangles.

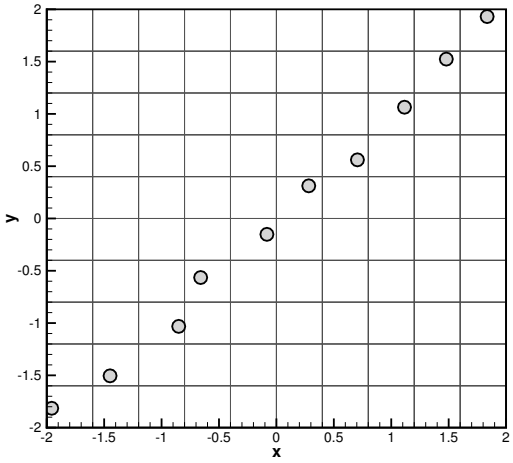
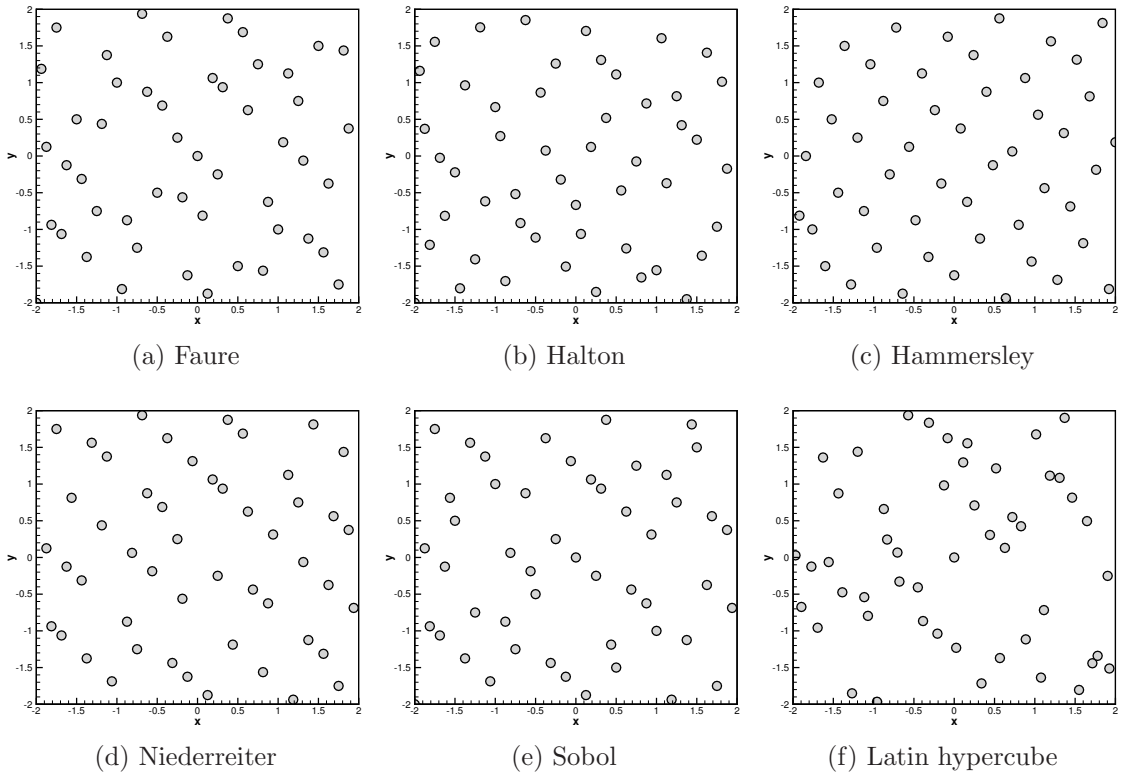


Figure 3.4: An illustration showing poor LHS distribution of training points.

3.1.4 Quasi-random Sequences

Quasi-random sequences [70] also known as *quasi-Monte Carlo* or *low discrepancy sequences* produce a sequence of deterministic points that fill the multidimensional space more uniformly than uncorrelated random points produced by pseudo-random number generators. They are primarily developed for the purpose of approximating multidimensional integrals efficiently using a better space filling than random or pseudo-random approaches such as LHS. Although pseudo-random numbers and quasi-random sequences both produce space filling designs, there are significant differences between the two. For a pseudo-random number generator such as LHS, it is possible for all N points to coincidentally be restricted to a particular region in the domain, or a distribution such as the one shown in Figure 3.4



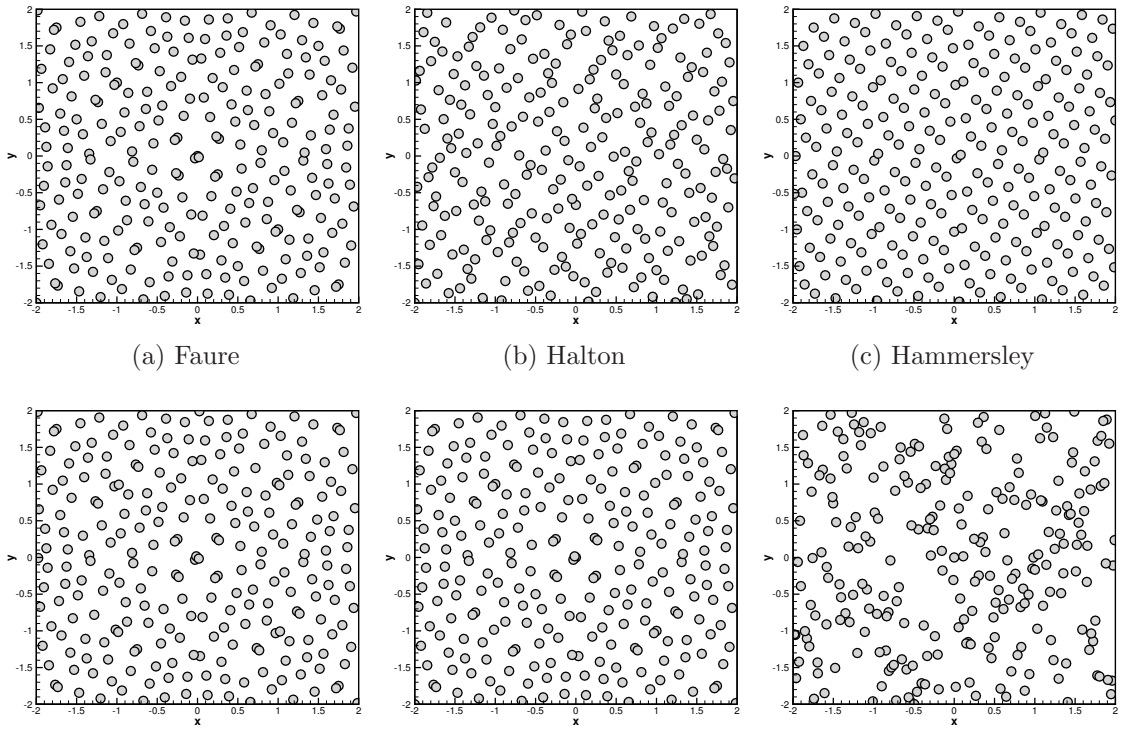
(a) Faure (b) Halton (c) Hammersley

(d) Niederreiter (e) Sobol (f) Latin hypercube

Figure 3.5: Training point distributions with quasi-random sequences using 50 points. A typical LHS distribution is also shown for comparison.

can occur. These are indeed undesirable characteristics in a training point selection strategy. It is not the case with quasi-random sequences where the points are constrained by a low-discrepancy requirement and are generated in a highly correlated manner such that the next point knows where the previous points are located.

Figures 3.5 and 3.6 show typical distributions using popular quasi-random sequences such as Faure, Halton, Hammersley, Niederreiter, and Sobol [25, 26, 70]. It can be noticed that these low discrepancy sequences fill the domain more uniformly, when compared to LHS that features a poor distribution. Though quasi-random sequences serve well to approximate multidimensional integrals by virtue of their nice space-filling properties, they can not be



(a) Faure (b) Halton (c) Hammersley

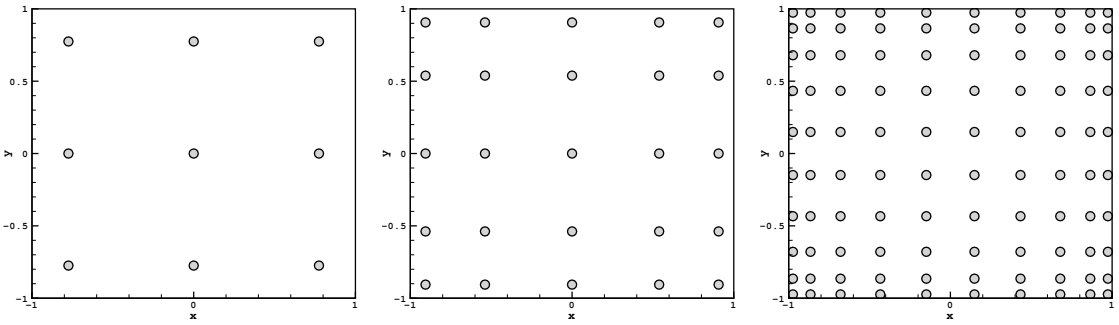
(d) Niederreiter (e) Sobol (f) Latin hypercube

Figure 3.6: Training point distributions with quasi-random sequences using 250 points. A typical LHS distribution is also shown for comparison.

expected to be a good candidate for training point selection, as they are insensitive to the function to be modeled and do not consider the response values or any form of available information from the metamodel.

3.1.5 Tensor Product and Sparse Grid Quadratures

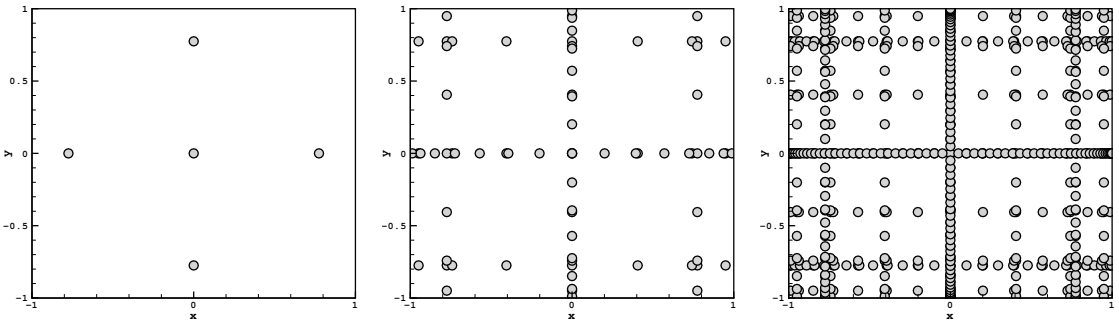
Quadratures determine the nodes (training points) based on the underlying probability distribution of the random variables *i.e.,* at the roots of corresponding orthogonal polynomials: Gauss-Hermite and Gauss-Legendre quadratures have their nodes distributed at the roots of Hermite and Legendre polynomials, respectively. They are also originally developed to approximate multidimensional integrals more effectively. Figure 3.7 shows an



(a) Order 3 (b) Order 5 (c) Order 10

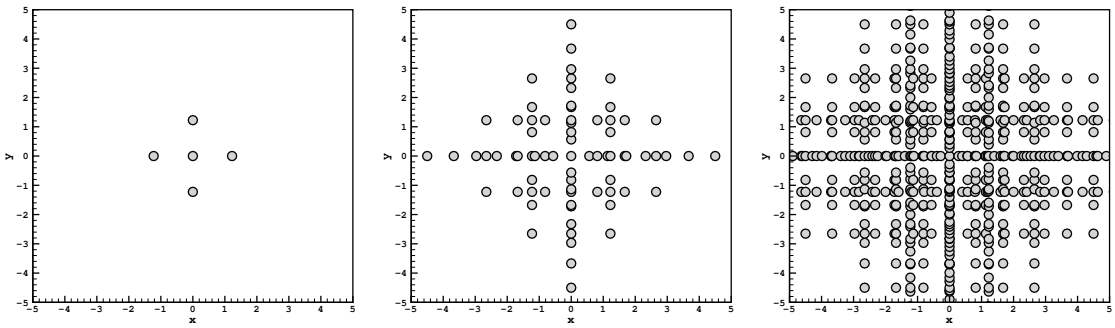
Figure 3.7: Gauss-Legendre grid distribution in two dimensions.

example of Gauss-Legendre quadrature with increasing order in a two-dimensional space. Multidimensional quadratures are easily obtained by tensor products of corresponding one-dimensional quadratures. Though they are shown to provide optimal convergence [23, 24], they become computationally intractable in higher dimensions, as they require $(N+1)^M$ function evaluations in M-dimensions.



(a) Level 1 (b) Level 3 (c) Level 5

Figure 3.8: Gauss-Legendre sparse grid distribution in two dimensions.



(a) Level 1 (b) Level 3 (c) Level 5

Figure 3.9: Gauss-Hermite sparse grid distribution in two dimensions.

Though efforts have lead to the development of sparse grid quadratures [72] (e.g. Smolyak sparse grids), they still suffer from high cost requirements. Moreover, sparse grids require smoother functions as they involve extrapolations. Figures 3.8 and 3.9 show some typical sparse grid quadrature node distributions.

3.2 Response-Based Approaches

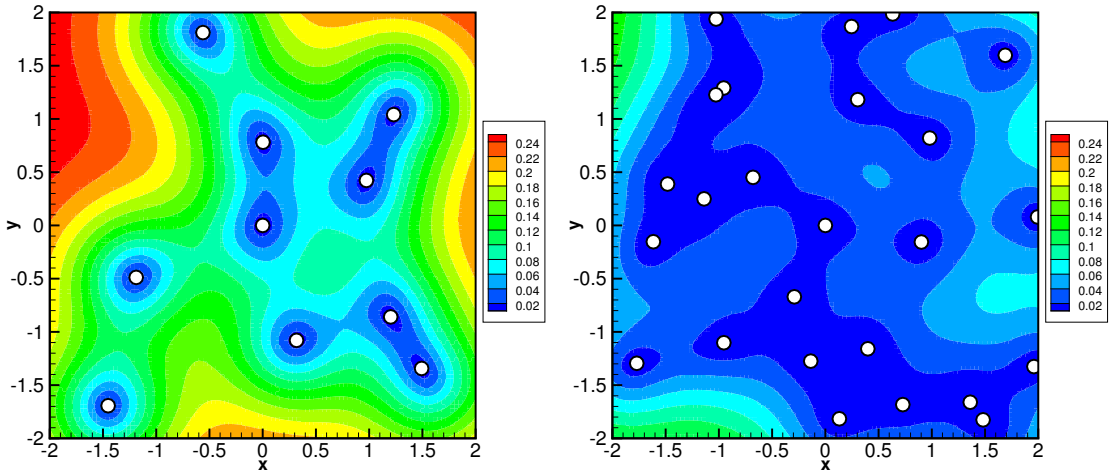
Response-based approaches for training the surrogate model allow the user to make use of the information available from the existing surrogate model. A brief note on some of these techniques are provided as follows.

3.2.1 Mean Squared Error & Expected Improvement

Mean Squared Error

The kriging surrogate model provides an uncertainty estimate in the form of the MSE given by Eq. 2.16. This parameter can be used to guide the training point selection by placing points in regions where the MSE is maximal. Figure 3.10 shows an example of the distribution of kriging MSE using the two-dimensional Runge test function. It can be seen

that the MSE is zero at training point locations and increases as the distance from training points increase. Therefore MSE-based selection processes produce a space filling design as MSE is a function of the distance between training points as discussed in chapter 2.1. Placing training points in regions where MSE is maximal is equivalent to only focusing on a global search and not considering chances for local improvements. This has led to the development of a figure of merit called expected improvement (EI) [73] that balances local and global search performances.



(a) N=10 (b) N=25

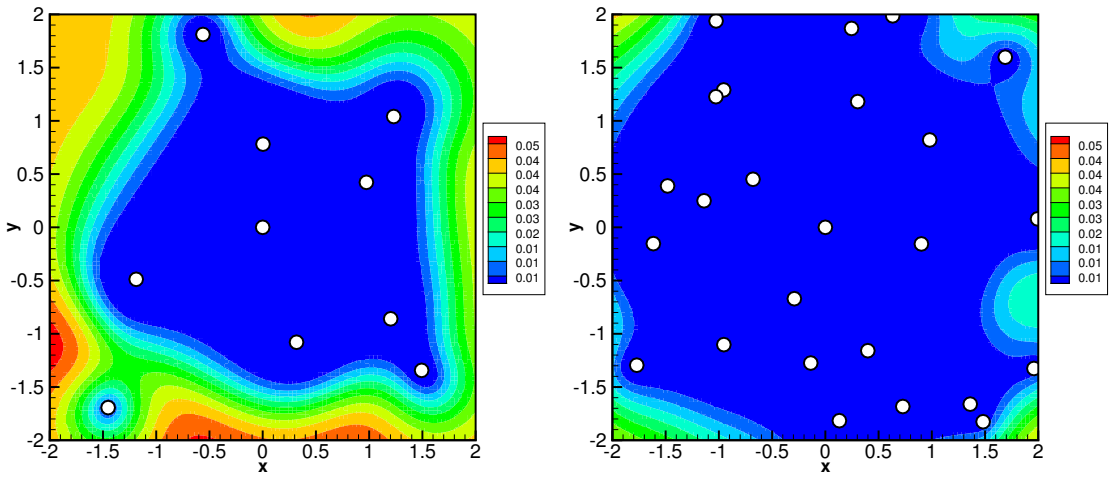
Figure 3.10: An example of MSE estimate provided by kriging using two-dimensional Runge function. The white circles refer to training point locations.

Expected Improvement

Jones *et al.* [73] suggested that training points can be selected in regions where the *expected improvement* (EI) function defined by Eq. 2.17 is maximal, *i.e.* in regions where the expectation of improvement in the objective function is maximal. During this process, the

training point set is updated, the metamodel is reconstructed, and the process of choosing additional training points is continued until the expected improvement from a potential new training point has become sufficiently small.

The EI based training point selection approach has great potential for finding the global optimum. However, this approach assumes MSE to be the actual error in the kriging prediction, when it is indeed not, and regions near the current best point have a greater EI value, causing the algorithm to cluster points near the current best point. Figure 3.11 shows the contours of the EI function over the domain of a two-dimensional Runge function.



(a) N=10 (b) N=25

Figure 3.11: Expected improvement provided by kriging using two-dimensional Runge function. The white circles refer to training point locations.

Though MSE/EI can be a better option than random training point selection, the methods have some disadvantages that are summarized below:

- MSE formulation does not incorporate any information from the actual response values $f(\boldsymbol{x})$.

- Not all surrogate modeling approaches provide MSE/EI estimates like kriging.
- Even though kriging has an MSE estimate, mathematically it is not an actual measure of error in the kriging prediction.
- EI is much more suited for optimization rather than building a globally accurate surrogate model as required for UQ.

3.2.2 Trust Region Method

The trust region method is associated with an approximation to the exact function which is only "trusted" within a small region of the current optimization iterate. The size of the trust region is modified during the search, based on how well the model agrees with actual function evaluations. If the approximation is in good agreement the trust region is expanded and it is contracted if the approximation is determined to be poor. Alexandrov *et al.* [29] proposed a *metamodel management framework* using trust region method for updating the metamodels (adding training data) according to improvements in the objective function during an optimization procedure.

CHAPTER IV

SURROGATE MODEL VALIDATION METHODS

Surrogate models are approximate mathematical representations of the exact function. Thus, model assessment strategies are required to ensure the adequacy of a surrogate for its applications. The "exact error" in a surrogate model can be quantified only by comparing it with the exact function. But this is impractical due to the computational resources required for exact function evaluations and perhaps can defeat the purpose of constructing a surrogate model in the first place. Chapter III provided a review of training point selection strategies, where some of them are shown to be driven by error-estimate-like quantities (e.g. kriging MSE estimate). Historically, several methods have been employed by the researchers for surrogate model validation, and a brief description of some of them are provided in this chapter.

A classification of methods available for calculating surrogate model error can be provided based on the following:

- Limitation to a particular surrogate modeling approach,
- Providing global or local measure of error,
- Furnishing true errors or approximations to the true error.

From a computational stand-point, they can also be categorized as:

1. Methods requiring additional exact function evaluations and

2. Methods without additional exact function evaluations.

The former methods are exact measures of surrogate model accuracy wheres the latter methods *introduce quantities that tend to match the exact error* or *provide standalone quantities* that can be used as a measure of precision. Depending upon the desired application area of surrogate models, either global (average) or local accuracy measures (pertaining to a given location) may be necessary, and this distinction is also addressed wherever possible in the following review of popular methods.

4.1 Methods with Additional Exact Function Evaluations

4.1.1 Root Mean Square Error

The root mean square error (RMSE) also known as L_2-norm can be written as,

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(f^{(i)} - \widehat{f}^{(i)})^2}, \tag{4.1}$$

where n is the number of test points that can either be nodes of a Cartesian mesh or random points in the M-dimensional domain over which the surrogate model is built. RMSE provides a global measure of surrogate accuracy.

4.1.2 R-Square Method

Jin *et al.* [74] introduced a quantity called R-Square (the coefficient of determination) that provides a measure of how well the model is likely to predict the exact function value at an arbitrary location $\boldsymbol{x}$ and is written as,

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(f^{(i)} - \widehat{f}^{(i)})^2}{\sum_{i=1}^{n}(f^{(i)} - \bar{f})^2} = 1 - \frac{\text{Mean Square Error (MSE)}}{\text{Variance}}. \tag{4.2}$$

R^2 measures discrepancies in the model (modeled by MSE) as well as irregularity in the problem (modeled by variance). Here, the mean is computed as $\bar{f} = \frac{1}{n}\sum_{i=1}^{n} f^{(i)}$. It is to be noted that a common factor of n gets canceled out in the above equation. A higher value of R-Square signifies a more accurate metamodel.

4.1.3 Relative Average Absolute Error

Jin *et al.* [74] also introduced a relative average absolute error (RAAE) defined as,

$$\text{RAAE} = \frac{\sum_{i=1}^{n} |f^{(i)} - \widehat{f}^{(i)}|}{n \times \sigma}, \tag{4.3}$$

where $\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(f^{(i)} - \bar{f})^2}$ refers to the standard deviation. A smaller value of RAAE corresponds to a more accurate surrogate model. This metric describes the overall accuracy of the model in the domain.

4.1.4 Maximum Absolute Error

The maximum absolute error (MAE) (also known as L_∞-norm) between the exact function f and approximated function values $\widehat{f}$ takes the mathematical representation,

$$\text{MAE} = \mathbf{max}\{|f^{(i)} - \widehat{f}^{(i)}|\} \qquad i = 1, \ldots, n, \tag{4.4}$$

n being the number of tested locations. MAE measures the greatest error occurring between the surrogate and exact function.

4.1.5 Relative Maximum Absolute Error

Jin *et al.* [74] introduced a relative maximum absolute error (RMAE) defined as,

$$\text{RMAE} = \frac{\mathbf{max}\{|f^{(i)} - \widehat{f}^{(i)}|\}}{\sigma} \qquad i = 1, \ldots, n, \tag{4.5}$$

which is a quantity measuring localized error in a particular region of the domain.

4.1.6 Split Sampling

Split sampling [33] also known as *holdout method* or *subset validation* divides the set of available data into two disjoint sets: training and testing data as shown in Figure 4.1. While the training data set is used for the surrogate construction, the resulting surrogate is tested by assessing the difference between the predicted value $\widehat{f}(\boldsymbol{x})$ and actual value $f(\boldsymbol{x})$, on the remaining set of testing points. The differences at each of the tested locations are summed up to give a *mean absolute test set error*, which can be used to evaluate the model's accuracy. The error estimate highly depends on the partition of available data and is thus

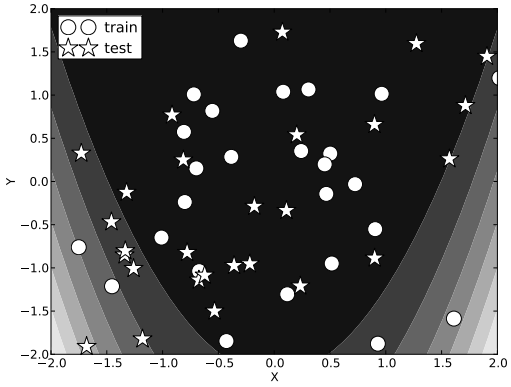


Figure 4.1: A demonstration of split sampling using a two-dimensional Rosenbrock function.

prone to produce varied or biased estimates. Also, the testing data set does not become a part of the surrogate construction process: thus a lot of exact function evaluations are not put to good use. Split sampling is considered belonging to methods requiring exact function evaluations to calculate the surrogate model error, since a separate set of precise observations are reserved for model validation and not used in surrogate building. Split sampling is the forerunner of the cross validation strategy discussed in section 4.2.2 which

addresses the inefficient use of exact function evaluations, by including all the available data for surrogate model training.

4.2 Methods without Additional Exact Function Evaluations

4.2.1 Bootstrapping

Efron [75] introduced bootstrapping to measure uncertainties in surrogate model predictions using three estimates: *simulation, statistical, and specification errors*. Bootstrapping has numerous variants namely non-parametric bootstrapping, smoothed bootstrapping, and 0.632 bootstrapping [76, 77]. A typical and simplest bootstrapping procedure is outlined below:

1. Generate a set of training data $\boldsymbol{F}(\boldsymbol{X}) = \{f^{(1)}, f^{(2)}, \ldots, f^{(N)}\}$ from exact simulations,
2. Construct a surrogate model using the available data and evaluate the model to get the desired output P (e.g. mean or variance of the exact function),
3. Generate simulated values $\widehat{f}(\boldsymbol{x})$ from the surrogate model $\widehat{\boldsymbol{F}}(\boldsymbol{X}) = \{\widehat{f}^{(1)}, \widehat{f}^{(2)}, \ldots, \widehat{f}^{(N)}\}$,
4. Use the simulated data as training data to construct the surrogate and re-estimate the desired output P.

The above steps are repeated m number of times as determined by the user and the outputs are compared to assess the model accuracy.

Bootstrapping is simple to implement and provides a measure for the stability of the results. However, the bootstrapping procedure is time-consuming and is thus not applicable for higher dimensional problems. For a detailed review on bootstrapping the reader is referred to works in the literature [76–78].

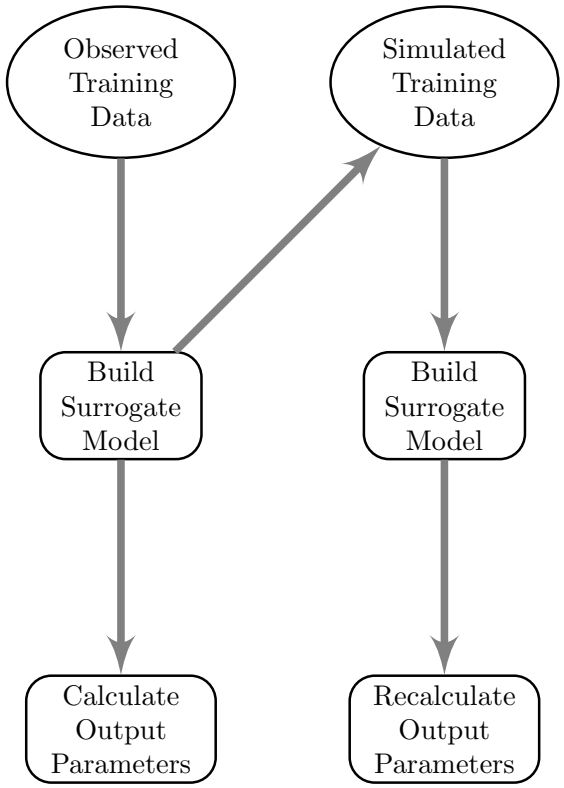


Figure 4.2: A sequence of steps involved in bootstrapping.

4.2.2 Cross Validation

Cross validation [78–80] is a popular method to estimate a surrogate model's accuracy which does not mandate any additional function evaluations.

4.2.2.1 k-fold Cross Validation

In $k-$fold cross validation the data set N is divided into k disjoint subsets with n training points in each set, and the surrogate model is constructed using the union of $k-1$ subsets of data (or equivalently $n(k-1)$ data points) and the remaining points from the left-out subset (n points) are used to find the *cross validation error* (CVE). The entire procedure is repeated k times, each time with a different subset for validation. Thus, the whole data

is used for training as well as validation, making it more attractive than split sampling.

$$\mathrm{CVE}^{(j)} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(f^{(i)} - \widehat{f}^{(i)})^2} \quad j = 1, \ldots, k, \tag{4.6}$$

where $f^{(i)}$ is already available and $\widehat{f}^{(i)}$ is provided by the surrogate model. The average error across all k trials is computed as *mean cross validation error*,

$$\mathrm{MCVE} = \sqrt{\frac{1}{k}\sum_{j=1}^{k}\mathrm{CVE}^{(j)}}. \tag{4.7}$$

A variant of this method has been proposed by Efron [80] to randomly divide the data into a test and training set m different times and repeat the above procedure to reduce the dependency of CVE on partitioning.

4.2.2.2 Leave-one-out Cross Validation

If the number of subsets, k, is equal to the total number of training points, N, then the approach is termed *leave-one-out cross validation*. In this case the surrogate model has to be reconstructed N times using $N-1$ training points each time. The cross validation errors can be calculated as,

$$\mathrm{CVE}^{(j)} = |f^{(i)} - \widehat{f}^{(i)}| \quad j = 1, \ldots, k, \tag{4.8}$$

and

$$\mathrm{MCVE} = \sqrt{\frac{1}{k}\sum_{j=1}^{k}\mathrm{CVE}^{(j)}} = \sqrt{\frac{1}{N}\sum_{j=1}^{N}\mathrm{CVE}^{(j)}}. \tag{4.9}$$

The cross-validation estimate of prediction error is nearly unbiased but can be highly variable. The disadvantage of this method is that the surrogate has to be constructed multiple times which can be computationally intensive as the training data size (N) increases. However, having multiple approximations may improve the robustness of the whole surrogate generation and validation approach, since all available data is used for both training and testing purposes. For more details on cross validation the reader is referred to Efron *et al.* [78, 80].

4.2.3 Regional Error Estimation

Mehmani *et al.* [28] developed a methodology to quantify the surrogate error in different regions of the domain, which is called the regional error estimation of surrogates (REES) method. The REES method provides a model independent error measure that does not require any additional function evaluations. After segregating the domain into sub-spaces (or regions) *variation of the error with sample density* (VESD) regression models are constructed to predict the accuracy of the surrogate in each subspace. These regression models are trained by the errors (the mean and maximum error) and evaluated for the intermediate surrogates in an iterative process. At each iteration, the intermediate surrogate is constructed using different subsets of training points and tested over the remaining points. Their results indicate that the REES measure is capable of evaluating the regional performance of a surrogate with reasonable accuracy.

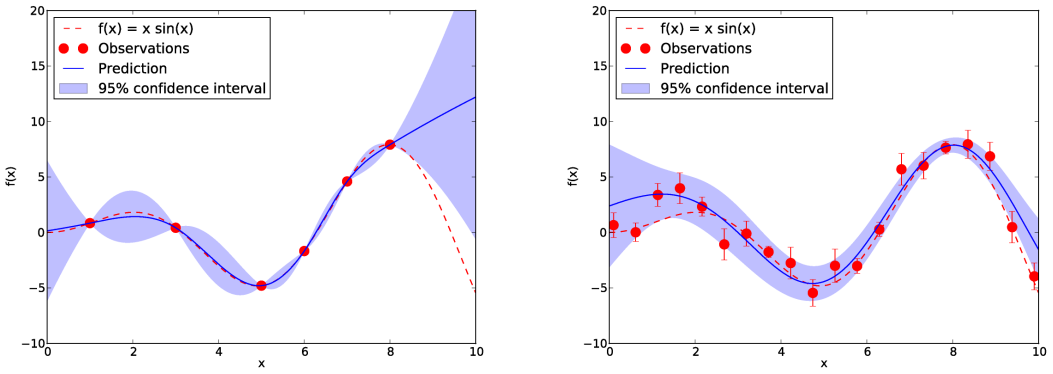
4.2.4 Surrogate Model Inbuilt Estimates

4.2.4.1 Kriging

The kriging prediction of a function value $\widehat{f}(\boldsymbol{x})$ comes along with an estimate of uncertainty in prediction, *the mean square error* which is shown in Figure 4.3a. More details on this measure were provided in sections 2.1.1.4 and 3.2.1.

4.2.4.2 Gaussian Process Regression

Gaussian process regression (GPR) [43] is similar to kriging but differs by virtue of its regression property. It also provides bounds on its prediction like kriging which is shown in Figure 4.3b. For kriging and GPR, the approximation bounds are based on statistical assumptions that errors are due to noise which follows a normal distribution with zero mean



(a) Kriging

(b) Gaussian process regression

Figure 4.3: An illustration for confidence bounds on surrogate model predictions of $f(x) = x\sin(x)$.

and variance σ^2. These measures are not an approximation to the actual error but rather an uncertainty bound which is a function of distance from existing training points.

4.2.4.3 Multivariate Interpolation and Regression

Similarly, the multivariate interpolation and regression (MIR) model provides uncertainty bounds on the approximated function value as $[\widehat{f}(\boldsymbol{x}) - c\sigma(\boldsymbol{x}), \widehat{f}(\boldsymbol{x}) + c\sigma(\boldsymbol{x})]$, where the function prediction $\widehat{f}(\boldsymbol{x})$ and its possible deviation $\sigma(\boldsymbol{x})$ are calculated in an equality-constrained least-squares problem [48, 49].

In this chapter a review of several error estimation methods, used within the surrogate modeling community to assess the accuracy of model predictions, has been provided. In spite of the presence of these methods, the main drawback among these methods are as follows:

- The error measurement methods discussed in section 4.1 require additional exact function evaluations and can therefore be prohibitively expensive. These methods

are generally only used for initial validation purposes on inexpensive analytical test functions.

- Many parameters offer guidance on the accuracy level of the surrogate model, but mathematically they are not an actual error measure (e.g. MSE, REES). Moreover, many surrogate modeling approaches do not come with built-in uncertainty bounds (e.g. polynomial chaos).

The lack of a good error estimator for surrogate models in spite of their numerous practical applications has been a major motivation for this work. This has led to the development of the proposed *unified training point selection and error estimation framework* requiring no additional exact function evaluations and which is applicable to any surrogate modeling approach. The details of the framework are elaborated in the following chapter.

A python based data mining and data analysis tool *scikit-learn* [81] was used to produce Figures 4.1 and 4.3.

CHAPTER V

PROPOSED SURROGATE TRAINING AND VALIDATION FRAMEWORK

The previous two chapters discussed the drawbacks associated with prevalent DoE and error estimates for surrogate models. In this chapter a unified framework for training point selection and error estimation for surrogate models is proposed. The fundamental driver of the framework is the information available from local surrogate models built over sub-domains of the main surrogate model. The local surrogate models refer to the ones that are built over the sub-domains of the global surrogate, using only a portion of the available data. A detailed account on the proposed dynamic training point selection and error estimation framework is provided in the following paragraphs.

5.1 Discussion of Steps Involved

Figure 5.1 shows a schematic diagram of the algorithm. The steps involved in the process are as follows.

I. Initialization

The exact function (also gradient and Hessian, if desired) is evaluated at the center of the domain and a few other points picked by LHS, totaling N_{init} training points. The choice of N_{init} is left to the user; however, it is recommended to start with a reasonably

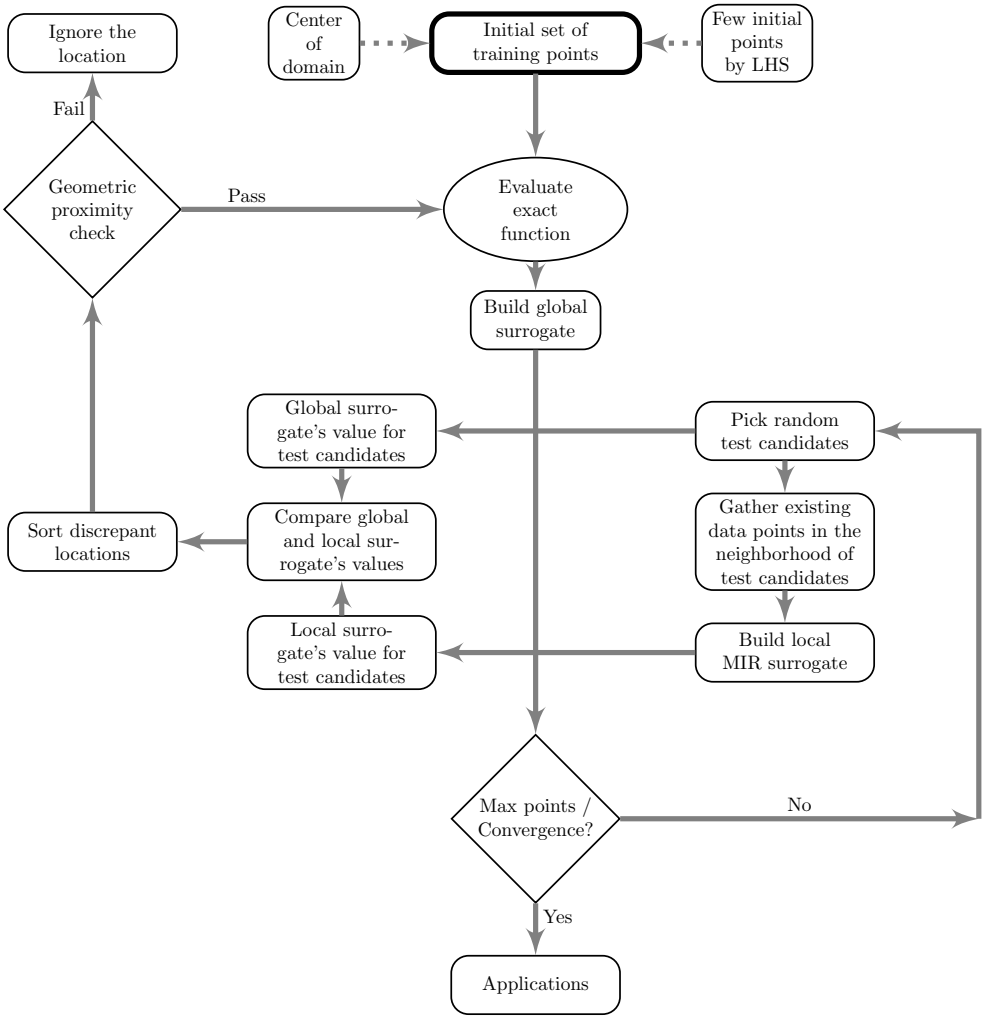


Figure 5.1: A schematic diagram of the proposed framework for training point selection using a local surrogate (MIR).

small number depending on the dimensionality and size of the domain. In this work, for two-dimensional test cases, N_{init} has been set to five for kriging and twelve for PCE, where the latter includes an oversampling factor of two and the required number of data points to build a second-order PCE (setting $p = 2$ and $M = 2$ in Eq. 2.27). For five-dimensional test cases, N_{init} is set to fifty for kriging and forty-two for PCE.

II. Evaluation of Surrogate Models

Numerous test candidates, $\boldsymbol{\xi}^{(j)},\ j = 1, \ldots, N_{test}$, are picked throughout the entire domain via LHS and the following is done with the two surrogates:

(a) **Global surrogate:** The global surrogate model, which is built using all available training data, is simply evaluated at all these test candidate locations yielding $\widehat{f}^{(j)}_{global}(\boldsymbol{\xi}),\ j = 1, \ldots, N_{test}$ and the values are stored.

(b) **Local surrogate:** During the first selection cycle, only one local surrogate is built using all available N_{init} training data points (making it a "second global surrogate"). The local surrogate model is also simply evaluated yielding $\widehat{f}^{(j)}_{local}(\boldsymbol{\xi}),\ j = 1, \ldots, N_{test}$. During subsequent selection cycles, local surrogate models are rebuilt using N_{local} closest existing training points for each test candidate $\boldsymbol{\xi}^{(j)}$ to evaluate $\widehat{f}^{(j)}_{local}(\boldsymbol{\xi})$.

In practical applications, it is intractable to evaluate the exact function to calculate the root mean square error of the global (main) surrogate model. A discrepancy function $\delta(\boldsymbol{\xi})$ is proposed as an approximation to the actual error $\epsilon(\boldsymbol{\xi})$ between the exact function and surrogate model at any given location $\boldsymbol{\xi}$, and is defined as the absolute difference in predictions from global and local surrogate models: $\delta(\boldsymbol{\xi}) = |\widehat{f}_{global}(\boldsymbol{\xi}) - \widehat{f}_{local}(\boldsymbol{\xi})|$. The underlying assumption is that the local surrogate models provide a more accurate representation of *their corresponding sub-domain* than the global surrogate model since piecewise approximations are generally considered to be more accurate for highly non-linear and non-smooth functions [82, 83]. Though the local surrogate models can be inaccurate in some cases, for example, due to scarcity of training data, extrapolations in unbounded space, improper tuning of the parameters for the local models, etc., they can still serve as reference models

for the global surrogate model. In the same context of discussion, multiple local approximations can also be constructed with yet another local surrogate model (e.g. radial basis functions [84], neural networks [85]) in addition to the MIR local surrogate for added fidelity, but this avenue is not explored in this work.

Proposed error measures: As discussed in chapter IV, actual surrogate model error estimates such as RMSE and MAE are prohibitively expensive to obtain. A brief note on RMSE and MAE has been provided in section 4.1. In order to validate the surrogate models in applications of practical interest two error estimates are proposed:

(a) A *root mean square discrepancy* (RMSD) defined as:

$$\text{RMSD} = \sqrt{\frac{1}{N_{test}} \sum_{j=1}^{N_{test}} (\widehat{f}_{global}^{(j)} - \widehat{f}_{local}^{(j)})^2} = \sqrt{\frac{1}{N_{test}} \sum_{j=1}^{N_{test}} (\delta^{(j)})^2}, \tag{5.1}$$

can be used as an approximation to the actual root mean square error (RMSE or L_2-norm) of the surrogate model.

(b) Similarly, a *maximum absolute discrepancy* (MAD) defined as:

$$\text{MAD} = \mathbf{max}\{|\widehat{f}_{global}^{(j)} - \widehat{f}_{local}^{(j)}|\} \qquad j = 1, \ldots, N_{test}, \tag{5.2}$$

measures the worst discrepancy between the local and global surrogate models and can be used to emulate the actual maximum absolute error (MAE or L_∞-norm).

Remark 1: The data used for building the local surrogate models is a subset of already available data, $f^{(i)}(\boldsymbol{x}),\ i = 1, 2 \ldots, N$. Thus, no additional exact function evaluations are needed for constructing the local surrogates.

Remark 2: Although the number of training points used to build a local surrogate, N_{local}, can be as high as the number of points used to build the global surrogate, N, it

is recommended to only use a N_{local} that is sufficient to produce a reasonably accurate local surrogate model. As a rule of thumb, studies similar to the ones shown in section 5.2 can be used to assess the accuracy of the local surrogate models. When it is not possible to determine the accuracy of the local surrogate models a priori (as in most scenarios), a certain percentage of the available training data (e.g. 25%) can be allocated for training the local surrogate models. In this work, 5–50 closest existing data points (depending on the dimension of the problem) are used to build the local surrogate models. Using more points to increase the accuracy of the local surrogate model will also increase the computational expenses in building and evaluating it.

Remark 3: The user should choose an appropriate number of test candidates, N_{test}, depending on the dimensionality of the surrogate. For example, heuristics used for Monte Carlo [26] or inexpensive Monte Carlo simulations [55] (IMCS) can be adopted. In this work 1,000–25,000 test candidates are used. A larger N_{test} facilitates a much wider exploration of the domain but comes with a higher cost. Fortunately, building and evaluating the local surrogates can be executed in parallel. As each *test candidate* is a *potential training point location* during the next selection cycle, these terms will be used synonymously.

III. Selection of Training Points

Selection is the process that determines whether or not a test candidate becomes a training point at which the exact function (gradient and Hessian) is to be evaluated. This includes the following two steps:

(a) **Sorting the discrepancies:** Locations (test candidates) are prioritized based on calculated discrepancies between the global and local surrogate model predictions. A sorted

discrepancy function vector contains values in the order of decreasing discrepancy and is represented as $\boldsymbol{\delta}_{sort}(\boldsymbol{\xi})$.

(b) **Proximity check:** The distance, $\rho^{(j)}$, between each test candidate, $\boldsymbol{\xi}^{(j)}$, and its nearest existing training point, $\boldsymbol{x}^{*(j)}$, is calculated. Mathematically,

$$\rho^{(j)}(\boldsymbol{\xi}^{(j)}, \boldsymbol{x}^{*(j)}) = ||\boldsymbol{\xi}^{(j)} - \boldsymbol{x}^{*(j)}||_2, \quad j = 1, \ldots, N_{test}. \tag{5.3}$$

The mean value of all these distances measures the average proximity of a potential training point (belonging to the set of test candidates) to an existing training point and is written as,

$$\bar{\rho} = \frac{1}{N_{test}} \sum_{j=1}^{N_{test}} \rho^{(j)}. \tag{5.4}$$

Now, each test candidate from the ordered set, $\boldsymbol{\delta}_{sort}(\boldsymbol{\xi})$, starting with the one with the largest discrepancy is checked for proximity to already existing training points by calculating whether $\rho^{(j)} > c\bar{\rho}$, where c is a *control parameter* that can be specified by the user to relax or strongly emphasize the constraint. If a particular test candidate passes the check, it is added to the actual set of training points and the exact function (gradient and Hessian) is evaluated. After every successful selection of a training point the distances given by Eqns. (5.3) and (5.4) are updated.

The proximity check is repeated until N_{cyc} new training points have been selected at this selection cycle. The newly available training data can now be used in subsequent selection cycles of the framework to update the surrogate models.

Remark 4: The enforcement of a geometric constraint works under the assumption that the global surrogate model is sufficiently accurate within $c\bar{\rho}$ distance from an existing training data point. In other words, the main surrogate does not warrant any additional training

data within this radius, and the exact function can rather be evaluated at a more unexplored location. This criterion ensures that the training points are not clustered in one particular region and are sparse in other regions of the domain. As already mentioned in chapter I each function evaluation can be computationally very expensive, especially for high-fidelity physics-based simulations, and it is essential to effectively choose each new training point.

Remark 5: The availability of gradient and Hessian information affects the local as well as global surrogate model's approximation. It influences the discrepancy function δ driving the framework and helps the user in identifying the most viable locations to evaluate the derivative information.

Remark 6: Due to the enforcement of the geometric constraint the framework may not support *fast* optimizations as these constraints can prevent the placement of many training points close to the optimum. Though the framework is recommended for building globally accurate surrogate models, the geometric constraints do not severely restrict the applicability to optimizations as they are controllable by means of the control parameter c.

IV. Termination

Steps (II) and (III) are repeated to iteratively update the global and local surrogate models until any one of the following criteria is satisfied.

(a) **Reach desired accuracy:** The maximum absolute discrepancy (MAD) and/or root mean square discrepancy (RMSD) can be used to monitor the estimated accuracy of the current surrogate. A close agreement between MAD and MAE as well as RMSD and RMSE will be shown in sections 6.2 and 7.2. The process of selecting additional training data can be stopped when the desired accuracy is reached.

(b) **Exhaust computational budget:** When a maximum number of exact function (gradient and Hessian) evaluations is reached the user can stop the training point selection process.

Remark 7: The framework can simply be used for surrogate model error estimation by skipping the third step (Selection of Training Points).

5.2 Choice of Local Surrogate Model

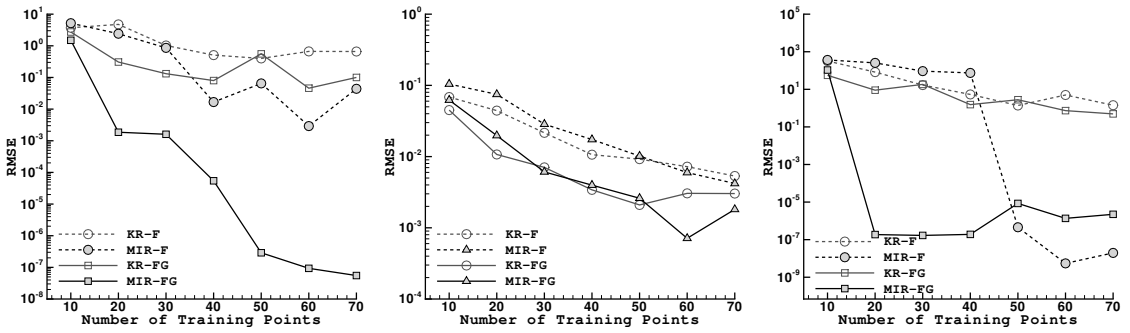
This section provides guidelines on choosing a good local surrogate model that can guide the framework effectively. The following discussions are directed on establishing the suitability of multivariate interpolation and regression (MIR) in serving as local surrogate model.

5.2.1 Ordinary Kriging and MIR

Figure 5.2 compares the original kriging and MIR on two-dimensional exponential, Runge, and Rosenbrock test functions (see section 6.1 for their definitions), where all training points are selected through LHS. The general observation is that MIR approximates all test functions except Runge better than kriging. The addition of gradient information for surrogate training (labeled FG, continuous lines) yields better results than training with function values alone (labeled F, dotted lines). The reader is referred to Wang *et al.* [48, 49] for a detailed comparison of MIR with other surrogate modeling methods and higher-dimensional test functions.

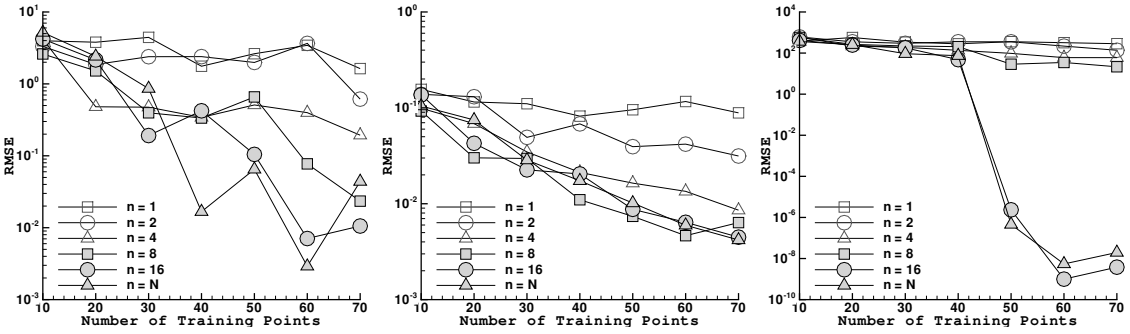
5.2.2 Effect of Taylor Order

Figure 5.3 shows the effect of different Taylor orders on the accuracy of the MIR approximated function value, $\widehat{f}$. It can be seen that a lower Taylor order, n, such as one



(a) Exponential (b) Runge (c) Rosenbrock

Figure 5.2: Plots of RMSE of the surrogate versus the number of training points using kriging and MIR on two-dimensional test functions.



(a) Exponential (b) Runge (c) Rosenbrock

Figure 5.3: Plots of MIR-RMSE versus the number of training points for different Taylor orders, n, on two-dimensional test functions.

or two produces a less accurate approximation, whereas a higher n leads to an improved approximation. For the Rosenbrock test function, after a certain number of points, the MIR expansion accurately models the function (close to machine precision). The problem with a higher Taylor order is that it increases the computational time in building the MIR surrogate significantly and there is not an easy way to choose the appropriate Taylor order a priori. It is recommended to use a Taylor order that is equal to the number of data points

(*i.e.,* $n = N$) when only function values are used for model training, or $3N$ if the data points also contain gradient information [48, 49].

Building the MIR surrogate becomes computationally very expensive with increasing number of training points. It can be seen from Figure 5.2 that MIR is able to produce comparatively good surrogates with just a few training points and hence are cheaper to build. It is generally acknowledged that, kriging has the capability to represent multi-modal functions and can effectively capture non-smooth functions [8]. Moreover, kriging supports the usage of both high- and low-fidelity training points *i.e.,* a variable-fidelity kriging surrogate [53, 55, 86–90] can be constructed. Due to these reasons kriging is preferred as the global surrogate model, whereas MIR is used to guide the training point selection process as well as to provide a metric for the accuracy of the global surrogate, using the framework discussed in section 5.1.

CHAPTER VI

IMPLEMENTATION RESULTS

This chapter presents the results obtained through the application of the proposed *training point selection and error estimation framework*. The discussions can be grouped as listed below:

- Validating the proposed error estimates for surrogate models (*root mean square discrepancy* and *maximum absolute discrepancy*) by comparing them with the actual error measures of the surrogate model (*root mean square error* and *maximum absolute error*),
- Demonstrating the superiority of the proposed "dynamic" training point selection versus other popular training point selection approaches such as LHS, low-discrepancy sequences and kriging MSE minimization in terms of *model accuracy* and *monotonicity*. Another motive is addressing the question of training point selection under the availability of gradient and Hessian information,
- Establishing the "non-intrusiveness" of the proposed framework by effective integration with two different surrogate models: the kriging and polynomial chaos expansions (PCE),

- Discussing results pertaining to Hessian-enhanced polynomial chaos which is one of the new contributions of this work,
- Presenting guidelines to users for a better combination of training point selection with higher-order derivative information, benchmarked based on *equivalent computational time* and their corresponding *accuracy*,
- Comparing the performance of kriging and polynomial chaos with each other in terms of *accuracy*, *robustness*, and *computational time*.

Data and results for the aforementioned discussions are shown for multi-dimensional analytical test functions which are defined next.

6.1 Analytical Test functions

Eqns. (6.1), (6.2) and (6.3) list the multidimensional exponential, Runge, and Rosenbrock test functions, respectively, which are used for evaluation purposes on an M-dimensional hypercube $[-2, 2]^M$.

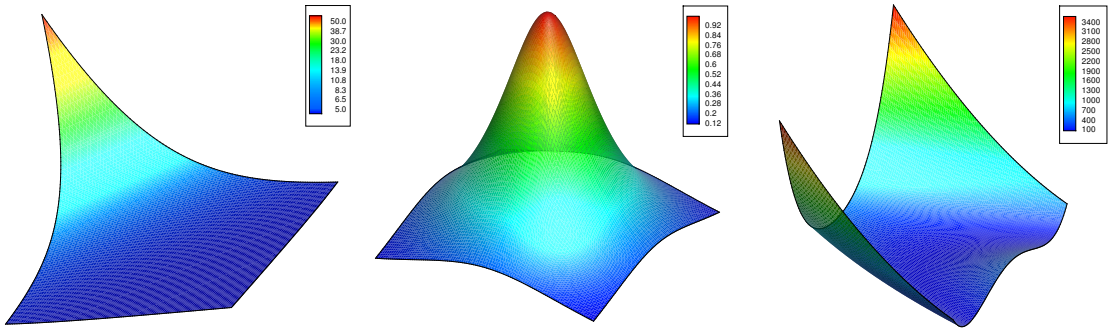
$$f_1(x_1, \ldots, x_M) = \mathrm{e}^{(x_1 + \ldots + x_M)} \tag{6.1}$$

$$f_2(x_1, \ldots, x_M) = \frac{1}{1 + x_1^2 + \ldots + x_M^2} \tag{6.2}$$

$$f_3(x_1, \ldots, x_M) = \sum_{i=1}^{M-1} \left[(1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2\right] \tag{6.3}$$

The choice of these three analytical test functions is due to the reasons outlined below. The first test function f_1 features an infinite Taylor's series expansion and it can not be captured "exactly" by any polynomial. The Runge function f_2 being a rational polynomial is known to pose difficulties for polynomial based response surfaces [91]. The Rosenbrock function is a popular test problem for gradient based optimizations, and being a fourth order polynomial it is captured exactly by any polynomial of order greater than or equal to

four. A two-dimensional surface contour plot for each of the above test functions is shown in Figure 6.1.



(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.1: Contour plots of analytical test functions in two dimensions where the contours are colored by the function values.

The root mean square error (RMSE) between the exact $f(\boldsymbol{x})$ and approximated function values $\widehat{f}(\boldsymbol{x})$ is calculated on an M-dimensional Cartesian mesh with N_t total nodes. Mathematically,

$$\text{RMSE} = \sqrt{\frac{1}{N_t}\sum_{i=1}^{N_t}(f^{(i)} - \widehat{f}^{(i)})^2}. \tag{6.4}$$

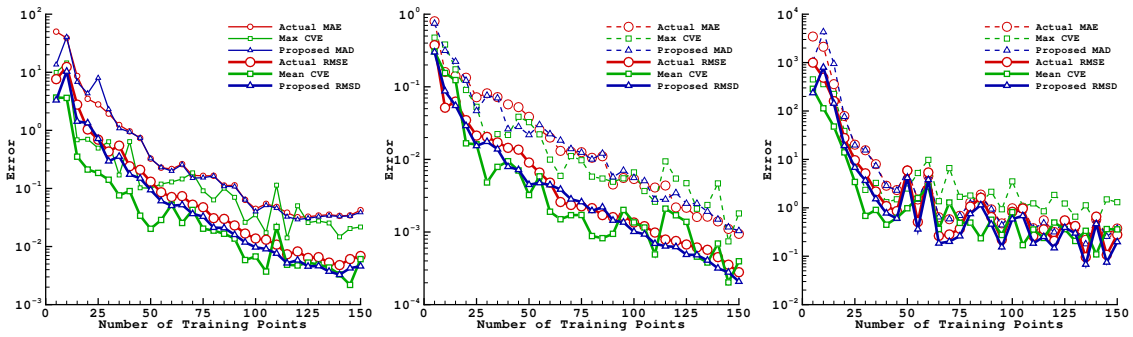
The RMSE is calculated on a Cartesian mesh with $10,201$ and $100,000$ nodes for two- and five-dimensional test cases, respectively.

6.2 Validation of Proposed Error Estimates

As discussed in section 5.1, the dynamic training point selection framework also provides an error estimate for the global (main) surrogate model through the maximum absolute discrepancy (MAD) and the root mean square discrepancy (RMSD).

6.2.1 Comparison with Actual Errors and Cross Validation

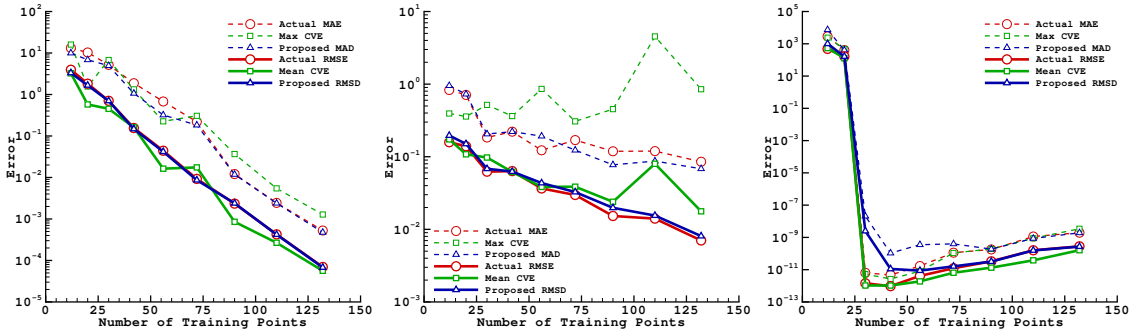
In this section, comparisons of: (i) the actual maximum absolute error (MAE or L_∞-norm) and the maximum cross validation error (Max–CVE) with the proposed MAD, and (ii) the actual root mean square error (RMSE or L_2-norm) and mean cross validation error (MCVE) with the proposed RMSD, are provided. Figures 6.2 and 6.3 show the above mentioned comparisons for kriging and PCE, respectively. A *leave-one-out cross validation* [79, 80] is employed in this study.



(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.2: A comparison of the proposed error measures (blue lines) with actual errors (red lines) and leave-one-out cross validation (green lines) on two-dimensional test functions using kriging.

The general observation is that both the proposed error measures (MAD and RMSD) feature an excellent agreement with the actual errors (MAE and RMSE) for all tested cases (exponential, Runge and Rosenbrock functions) and surrogate modeling approaches (kriging and PCE), with only a few occasional minor differences. This presented behavior shows great promise to validate the surrogate model without warranting exact function



(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.3: A comparison of the proposed error measures (blue lines) with actual errors (red lines) and leave-one-out cross validation (green lines) on two-dimensional test functions using PCE.

evaluations in applications of practical interest. Cross validation exhibits satisfactory tendencies in matching the actual errors but its overall performance is more chaotic. For all test cases a maximum of twenty-five nearest existing training points are used to build local approximations with MIR.

6.2.2 Comparison with Error Distributions in the Domain

In Figures 6.2 and 6.3 quantitative metrics have been used to validate the proposed error estimate for surrogate models. This section is intended to provide an insight into the spatial distribution of the *actual error,* ϵ, and the *proposed discrepancy,* δ. Figures 6.4, 6.5 and 6.6 show contour plots of the distribution of: (i) the local surrogate model error, $\epsilon_{local} = |f - \widehat{f}_{local}|$, (ii) the global kriging surrogate model error, $\epsilon_{global} = |f - \widehat{f}_{global}|$, and (iii) the proposed discrepancy function, $\delta = |\widehat{f}_{local} - \widehat{f}_{global}|$, for the exponential, Runge and Rosenbrock test functions, respectively.

The main assumption of the framework has been that the local surrogate models provide a more accurate representation of their corresponding sub-domain than the global surrogate

model. The local and global surrogate model errors shown in the leftmost and middle contour plots of Figures 6.4, 6.5 and 6.6 reveal that the local surrogate models are more accurate than the global surrogate model. For all the cases the local surrogate models use the closest 25 data points to predict the function value at a test location. For the Rosenbrock test function (see Figure 6.6) the local surrogate is indeed a "second global surrogate model" as all the available data (25 points) is used for its predictions.

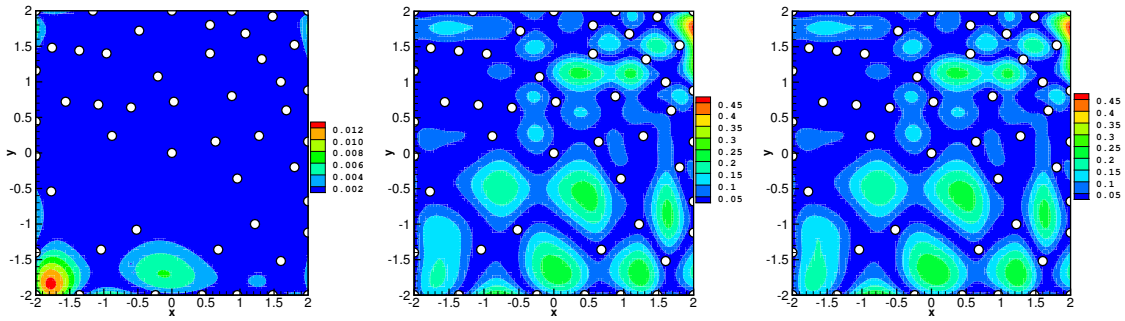


Figure 6.4: Contour plots for the exponential test function showing the distribution of: the local surrogate model error (left), the global kriging surrogate model error (middle) and the proposed discrepancy function (right). The global and local surrogate models are built with 50 and 25 training points (white circles), respectively.

The differences between the function predictions of the two surrogate models (local and global) is proposed as an approximation to the actual error in the global surrogate model as discussed in section 5.1. From Figures 6.4, 6.5 and 6.6 a close agreement can be noticed between the global surrogate model's actual error distribution (middle) and the proposed discrepancy function (right), which explains the excellent trends shown in Figures 6.2 and 6.3. Only for the Runge function (see Figure 6.5) can differences be visually seen between the actual error and the proposed discrepancy function. Similar plots for PCE are not shown here, but the behavior is clearly evident from the presented trends.

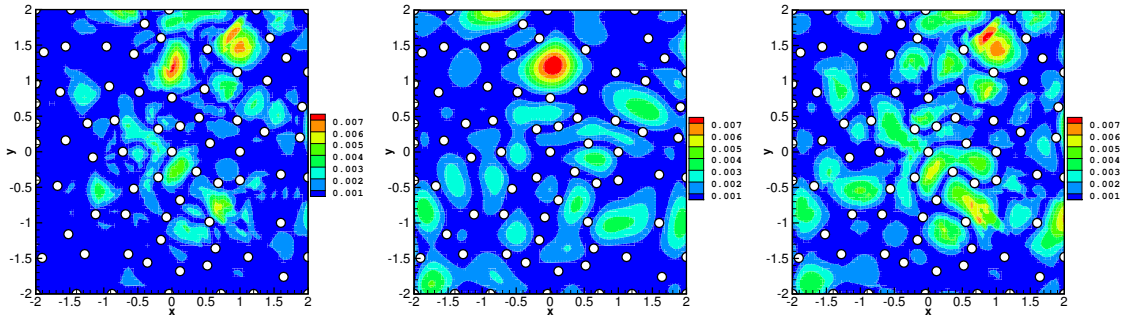


Figure 6.5: Contour plots for the Runge test function showing the distribution of: the local surrogate model error (left), the global kriging surrogate model error (middle) and the proposed discrepancy function (right). The global and local surrogate models are built with 75 and 25 training points (white circles), respectively.

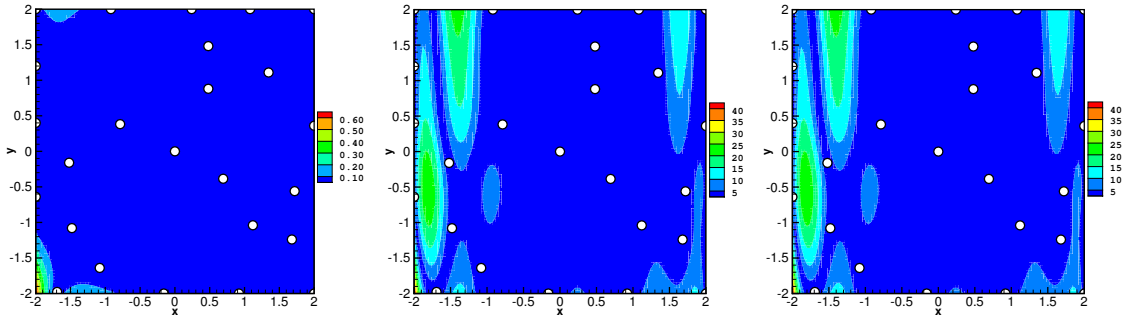


Figure 6.6: Contour plots for the Rosenbrock test function showing the distribution of: the local surrogate model error (left), the global kriging surrogate model error (middle) and the proposed discrepancy function (right). The global and local surrogate model(s) are both built with 25 training points (white circles).

Generally, it is possible for the local surrogate models to provide less accurate representation of the domain, especially when only a few training points are available. Therefore, it is advisable to use all the available training data to build local surrogate model(s) during the first few selection cycles. For example, in this work the local surrogate models for the two-dimensional cases use all the available training data until the size of the training data set increases beyond 25, after which it is fixed at 25 for computational efficiency purposes.

Nevertheless, the accuracy of the local surrogate models is *not a strict requirement*, as they can still be used as reference models to build and validate the global surrogate.

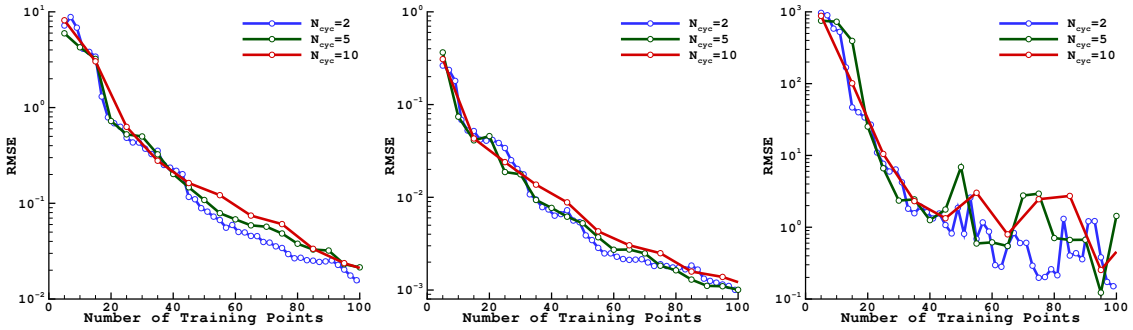
As a final remark, if the training point selection process is desired to be continued, the framework would choose points where the discrepancies shown in the rightmost contour plots of Figures 6.4, 6.5 and 6.6 are large (distance-constraint will also be checked before evaluating the expensive exact function).

6.3 Kriging Results

6.3.1 Number of Training Points per Cycle

The proposed dynamic training point selection features a progressive evolution of the training data set for the global surrogate model. The user specifies the number of training points (N_{cyc}) to be added at each iteration to the training data set - a factor that determines the rate at which the final training data set is evolved. In the case of PCE, the surrogate is built in steps of one polynomial order per selection cycle and the required number of additional points can be determined from Eq. (2.27) and the chosen oversampling factor which is two. In the case of kriging, the choice of N_{cyc} is left to the user as kriging does not mandate any requirements on the minimum number of points needed to build the surrogate. It is recommended to add a moderate number of training points per iteration to facilitate a *better evolution* of the training data set. Adding only a few points per cycle implies more computational burden since the kriging (also PCE or any response surface) has to be constructed more often to reach a fixed number of training points.

Figure 6.7 shows the effect of the number of training points added per iteration on the accuracy of the global surrogate for all three analytical test functions in two dimensions. The training can be done by choosing more points per cycle, to reduce the number of times



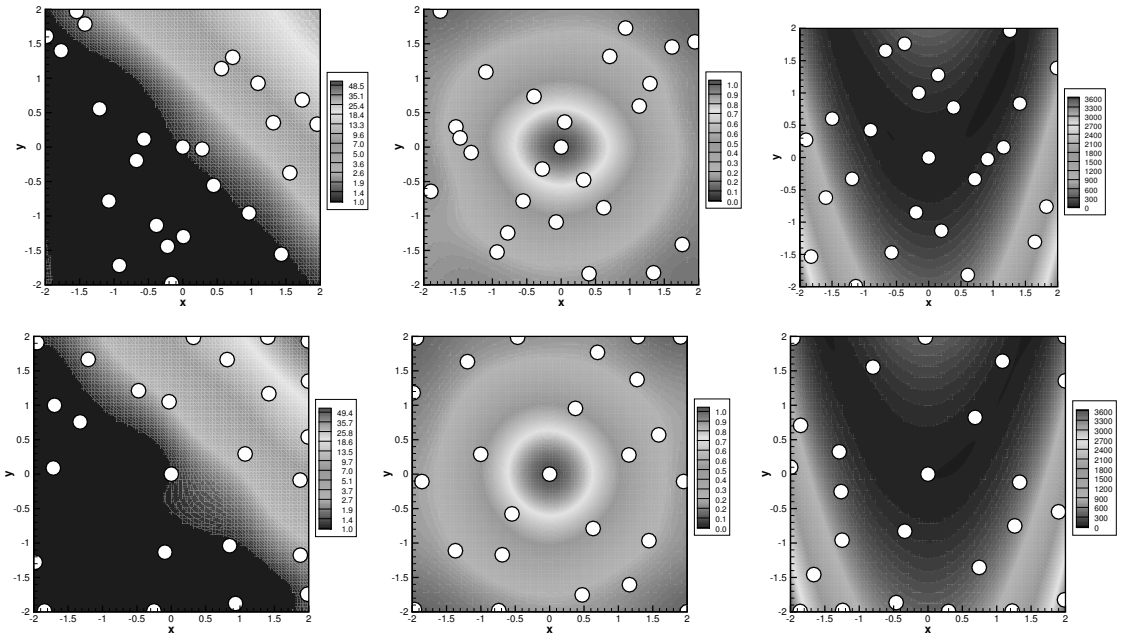
(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.7: The effect of the number of training points selected per cycle on the accuracy for two-dimensional test functions using kriging.

the surrogate model needs to be built, as the general monotonic behavior is preserved for all the tested cases (two, five and ten) and also because the trade-off with accuracy is found to be small. For the two-dimensional Rosenbrock test function, the kriging surrogate model does not present a monotone behavior after reaching about 40 training points, due to failed tuning of kriging hyper-parameters. A similar chaotic convergence behavior was observed for other polynomial test functions such as the quadratic and cubic ones (results are not shown here). It will be shown later (see section 6.3.3.2) that kriging exhibits the same behavior with all the tested training point selection approaches.

6.3.2 Training Point Distribution

Figure 6.8 shows typical training point distributions for all three test functions using LHS as well as the proposed dynamic method using kriging. In this example, five points were selected per iteration until reaching the final training data set with 25 points. It can be observed that the dynamic training strategy concentrates training points in regions where the curvature is high, near the peaks and bounds, and in regions where the points are sparse.



(a) Exponential (b) Runge (c) Rosenbrock

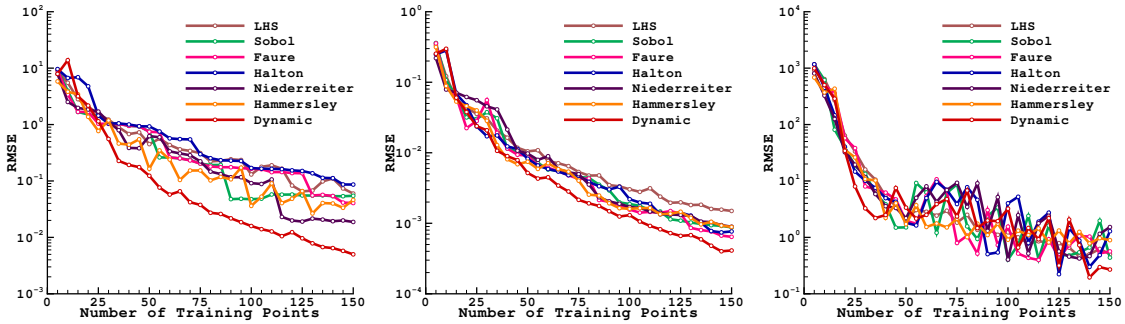
Figure 6.8: Training point distributions ($N = 25$) for two-dimensional test functions using LHS (top) and the dynamic method (bottom).

For example, the top row of Figure 6.8 (corresponding to latin hypercube sampling) has larger unsampled regions in the domain than the bottom row (dynamic training), where the training points are more spread throughout the domain. Such a strategy can save a lot of computational time when applied to high-fidelity simulations by reducing the number of required function evaluations to produce a globally accurate surrogate model. On the other hand, LHS tends to miss important locations as well as affects the matrix conditioning through too closely spaced points.

6.3.3 Accuracy of Dynamically Enhanced Kriging Surrogate Model

A quantitative comparison of the accuracy of the dynamically trained (and thereby enhanced) kriging surrogate model will be provided in the following paragraphs by means of RMSE comparisons.

6.3.3.1 Comparison with Low-discrepancy Sequences using Kriging



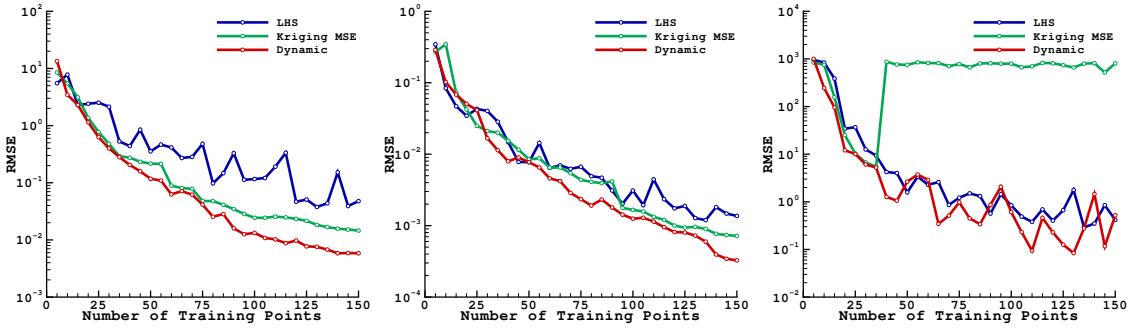
(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.9: A comparison of the dynamic training point selection with some quasi-Monte Carlo sequences and LHS on two-dimensional test functions using kriging.

The dynamic training point selection is compared with some quasi-Monte Carlo sequences such as Sobol, Faure, and Halton. LHS is also included in the comparisons to show its performance compared to other strategies. From Figure 6.9, it can be seen that the dynamic training point selection is better at producing accurate surrogate models than the other approaches. As discussed above, for the Rosenbrock test function all training point selection strategies suffer from chaotic convergence with kriging due to the choice of spatial correlation function.

6.3.3.2 Comparison with Kriging MSE Minimization

The kriging surrogate model provides an error estimate through the expected mean squared error (MSE) as described earlier. The MSE can be used to guide the training point selection process by adding training points at locations where the MSE is large. Figure 6.10 shows a comparison of kriging surrogate models built with LHS, minimizing MSE and the proposed framework for training point selection. For all three test functions, the proposed framework produces more accurate surrogate models.



(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.10: A comparison of the dynamic training point selection with minimizing the MSE method on two-dimensional test functions using kriging. LHS is also included to show its relative performance.

For the Rosenbrock function f_3, the MSE method features a poor convergence after about 40 training points and is even outperformed by LHS. In order to alleviate the non-smooth convergence behavior of kriging for the Rosenbrock test function (using LHS, Sobol, Faure, Halton, MSE, and the proposed framework) other spatial correlation functions have been employed and the results are displayed in Figure 6.11. By comparing the rightmost Figure 6.10 (using Wendland C4) with Figure 6.11 (using Gaussian and spline) it can be

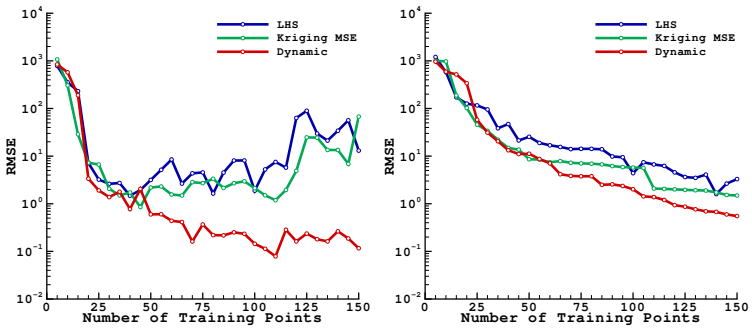


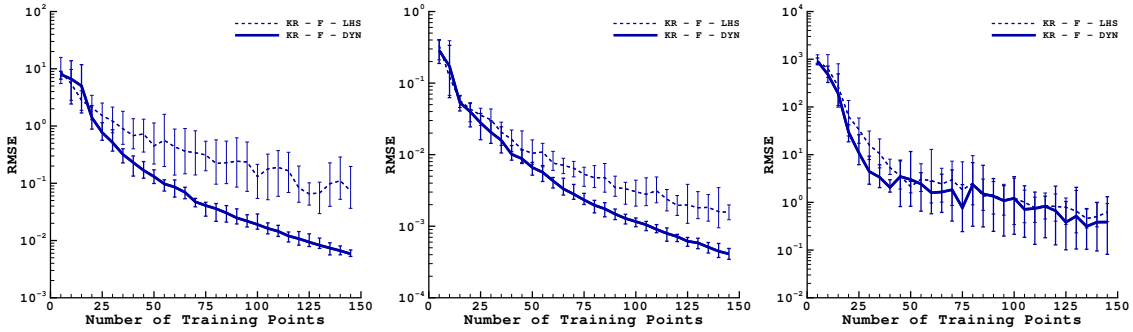
Figure 6.11: A comparison of accuracies of the kriging model built with Gaussian (left) and spline (right) spatial correlation functions for two-dimensional Rosenbrock test function.

noted that spline spatial correlation functions feature a much smoother convergence when compared to Wendland C4 and Gaussian spatial correlation functions. However, since a derivative-enhanced kriging model is employed in this work, Wendland C4 spatial correlation function [92] is used as default for the remainder of this article to ensure differentiability. Also, for further comparisons only LHS will be considered to reduce the complexity in presenting the results.

6.3.4 Dynamic versus LHS in 2D

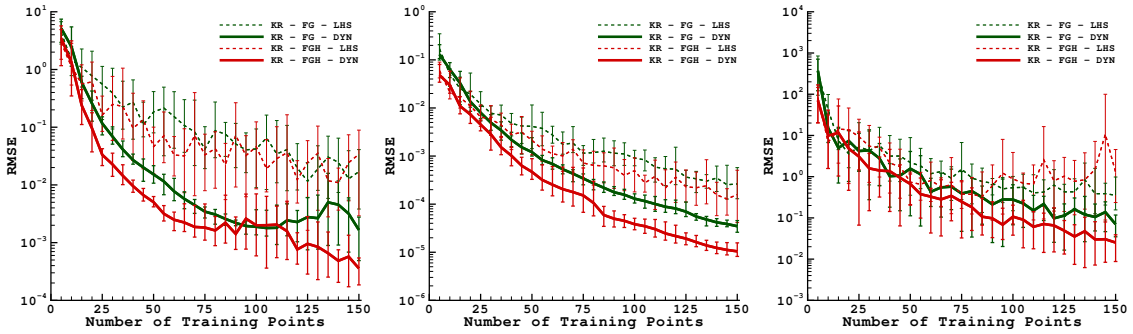
In order to account for the inherent randomness in LHS all two-dimensional results are averaged over ten separate runs and the mean results are presented with bounds referring to the best and worst case.

From Figure 6.12 it can be inferred that the dynamic method (shown with continuous lines) performs better than LHS (shown with dashed lines), both in terms of monotonicity and accuracy. The dynamic training point selection improves the accuracy of the surrogate by roughly an order of magnitude when compared to LHS, for the first two test functions.



(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.12: Plots comparing training point selections using LHS and the dynamic method on two-dimensional test functions (with function values only) with kriging. Five training points are selected per cycle.



(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.13: Plots comparing training point selections using LHS and the dynamic method on two-dimensional test functions (with derivative information) with kriging. Five training points are selected per cycle.

For the Rosenbrock test function (f_3), though not as distinct as for the other test functions, the lower bounds show that the dynamic method is still more accurate than LHS.

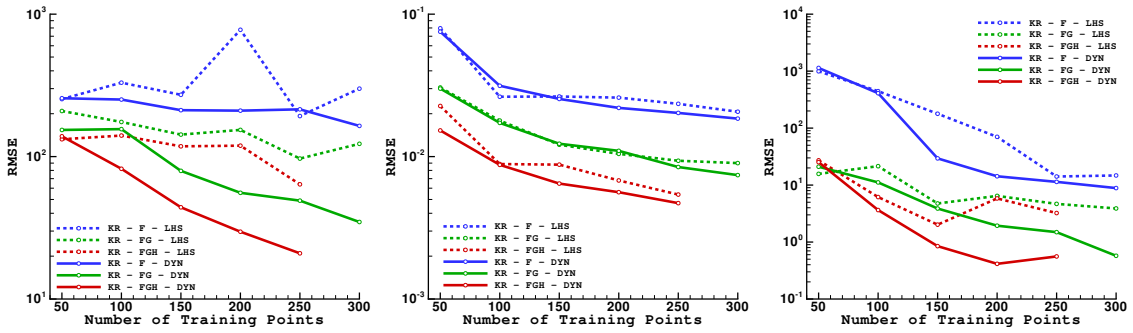
The training point selection in the presence of derivative information is shown separately in Figure 6.13. It can be noted that the addition of gradient and Hessian information helps to improve the accuracy of the response surface for both LHS and the dynamic training

point selection. The dynamic training point selection improves the accuracy even more by placing the higher-order derivative information in the most viable locations. This behavior can be observed across all test functions in Figure 6.13. Also, it can be seen that the LHS Hessian cases (FGH) of f_1 and f_3 (Figure 6.13 left and right, respectively), are not more accurate than the gradient cases (FG) as expected. But with the dynamic method, auxiliary information always improves the accuracy of the surrogate model, as it ably chooses the locations that would contain derivative information.

The smaller bounds on the RMSE for the dynamic method in the figures indicate that it is less random than LHS implying that only one run is sufficient whereas most random training point selection methods require multiple runs to ensure that "bad luck" does not impede the results. In most instances, the upper bounds of the dynamic method (corresponding to its worst approximation) are still better than the lower bounds of LHS (corresponding to its best approximation). Hence it can be expected that the dynamic training point selection method to produce more accurate surrogates for a fixed computational budget. The superiority of the dynamic method over other quasi-Monte Carlo sequences has already been demonstrated in Figure 6.9.

6.3.5 Dynamic versus LHS in 5D

Figure 6.14 shows the results for five-dimensional test functions. Due to the computational resources needed to build five-dimensional surrogate models, error bounds are not included. It can be seen that the dynamic method (shown as continuous lines) helps to improve the accuracy of the kriging surrogate compared to LHS (shown as dashed lines). The advantage of including derivative information is also more evident for these higher-dimensional test cases.



(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.14: Plots comparing training point selections using LHS and the dynamic method on five-dimensional test functions with kriging. Fifty training points are selected per cycle.

As a general remark, if the proposed dynamic training point selection (or any response-based method) is used, the surrogate is built progressively starting from an initial (small) number of training points, to achieve better accuracy for a fixed computational budget. The cost of building the surrogate model multiple times can generally be neglected in comparison with the computational cost of high-fidelity physics-based simulations. If the surrogate construction should become too expensive, it can be accelerated by selecting a larger number of additional training points per cycle for a minimal trade-off with accuracy as demonstrated in Figure 6.7.

6.4 Polynomial Chaos Results

Results pertaining to the polynomial chaos expansions are discussed in this section. An oversampling factor of two is enforced to improve the matrix conditioning (prevents accidental rank deficiency). The dynamic training point selection is initiated by building a second-order accurate PCE including a training point at the center of the domain. As the

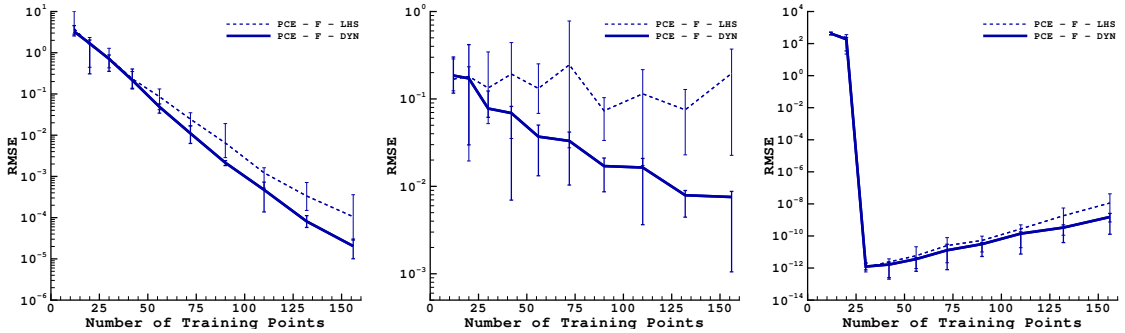
order of expansion increases, the required number of collocation points (training points) are picked dynamically.

6.4.1 Dynamic versus LHS in 2D

Figure 6.15 compares PCE with LHS (shown as dashed lines) with that of PCE with dynamic training point selection (shown as continuous lines) for all three test functions, along with error bounds similar to the results presented for kriging. As it can be seen, the convergence behavior of PCE with LHS is chaotic compared to that of the dynamic strategy which is monotonic instead. Although the rational Runge function (f_2) is known to pose difficulties for PCE [91], the dynamic method shows good convergence whereas LHS shows very poor results. Note that recently a rational polynomial chaos expansion scheme has been developed by Sheshadri *et al.* [91] to address the difficulty of PCE with such functions. The fourth-order polynomial Rosenbrock function (f_3) is captured exactly after the order of expansion reaches four; however, it suffers from polynomial over-fitting (using a model that is more sophisticated than needed) [93] as the order of expansion increases further. Overall, the dynamic training point selection consistently produces more accurate surrogates compared to LHS. Figure 6.16 shows the improved performance of the dynamic method for choosing training points that contain gradient (labeled FG) and Hessian information (labeled FGH). In particular, for the Runge function, the advantage of the proposed framework is very pronounced.

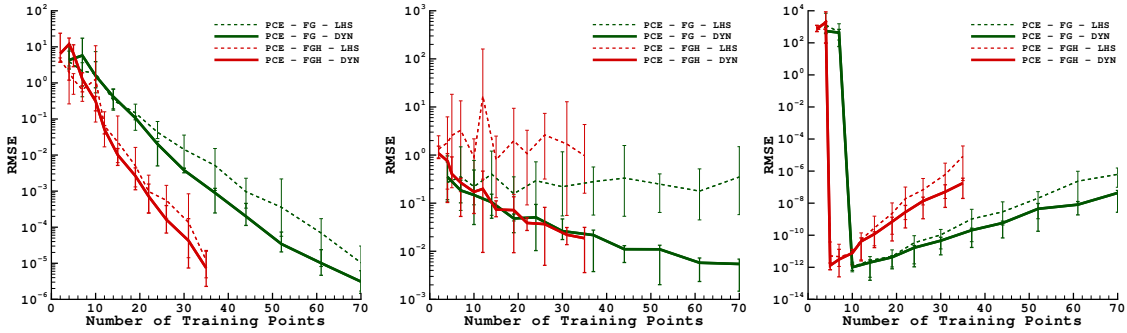
6.4.2 Dynamic versus LHS in 5D

Figure 6.17 the shows five-dimensional results. It is again evident that the dynamic method is more consistent in producing a good PCE surrogate compared to LHS which tends to show random fluctuations. When the function values alone are used, about 6000



(a) Exponential (b) Runge (c) Rosenbrock

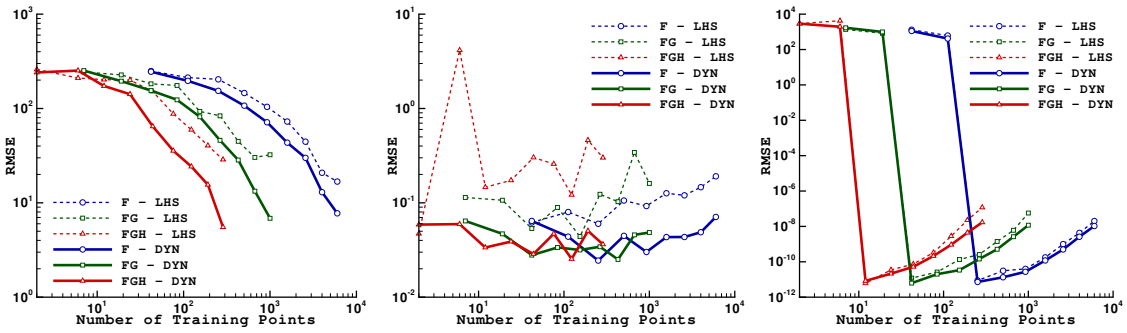
Figure 6.15: Plots comparing training point selections using LHS and the dynamic method on two-dimensional test functions (function values only) with PCE. The order of expansion, p, ranges from 2 to 11.



(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.16: Plots comparing training point selections using LHS and the dynamic method on two-dimensional test functions (with higher-order derivative information) with PCE. The order of expansion, p, ranges from 2 to 12.

training points are needed for a tenth order polynomial. If gradients are used, only about 1000 points are needed and only about 200 points are required with function, gradient and Hessian information to roughly obtain the same level of accuracy. This is because of the availability of additional training information as discussed in section 1.1.2.3.

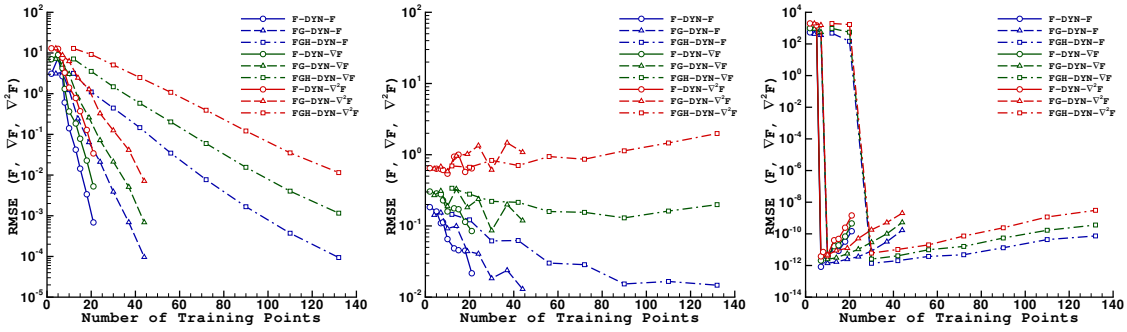


(a) Exponential (b) Runge (c) Exponential

Figure 6.17: Plots comparing training point selections using LHS and the dynamic method on five-dimensional test functions. The order of expansion, p, ranges from 2 to 10.

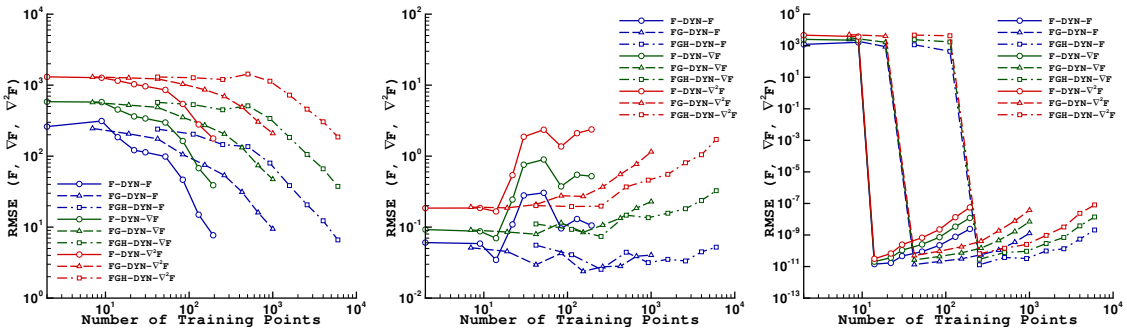
6.4.3 Validation of Gradients and Hessian from Surrogate

Figures 6.18 and 6.19 show the RMSE between the approximated function (shown as blue lines), gradients (shown as green lines) and Hessian (shown as red lines) with that of the exact values in two and five dimensions, respectively. It can be seen that the PCE produces good approximations of gradient and Hessian information that can be used for many applications including optimization and uncertainty quantification.



(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.18: RMSE in the approximated function, gradient and Hessian values from PCE for two-dimensional test functions.



(a) Exponential (b) Runge (c) Rosenbrock

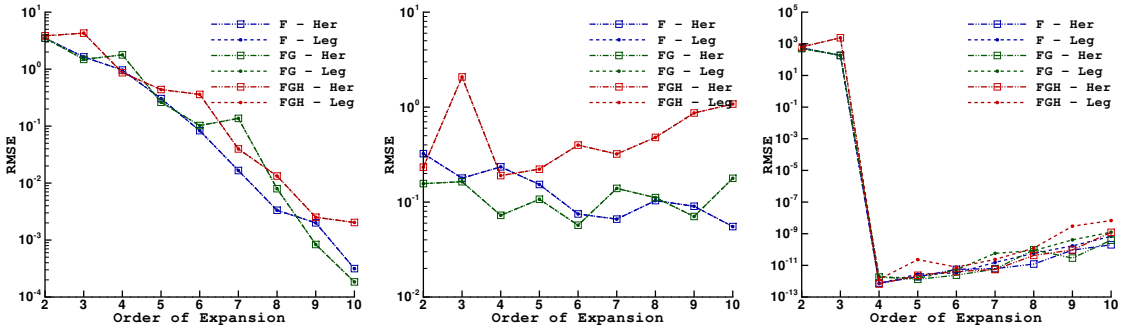
Figure 6.19: RMSE in the approximated function, gradient and Hessian values from PCE for five-dimensional test functions.

6.4.4 Choice of Orthogonal Polynomials

All the results shown above for PCE use Legendre orthogonal basis. Results in Figure 6.20 show the effect of using different orthogonal polynomials to construct the basis matrix ψ. For a reasonable comparison, the same set of training points (chosen via LHS) that were used with Legendre orthogonal basis were also used with Hermite orthogonal basis. Though the PCE coefficients $\boldsymbol{u}$ are found to be different when using Hermite polynomials as basis, it did not show any impact on the overall accuracy of the PCE surrogate model, as polynomial interpolation is unique. For f_3 small deviations can be seen only after reaching machine precision (not due to the choice of basis).

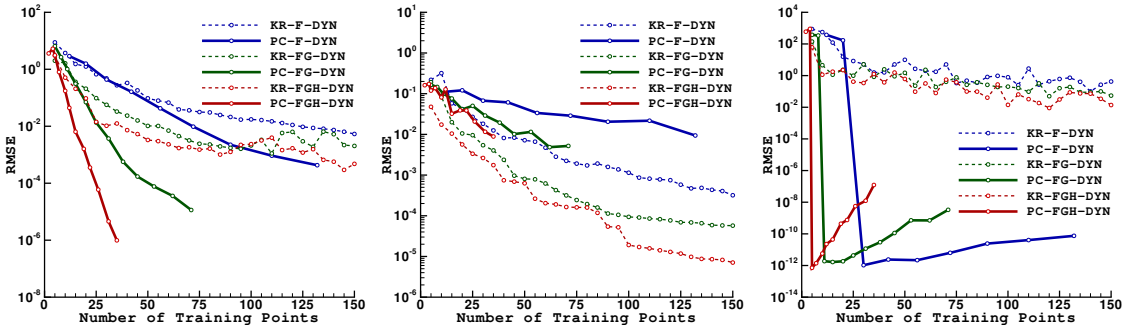
6.5 Comparison of Kriging and Polynomial Chaos

The purpose of this section is to compare the kriging and PCE surrogates, both of which are enhanced with the dynamic training point selection method on analytical test functions. In Figure 6.21 for the exponential function f_1, PCE exhibits a smoother as well as better convergence rate for all the three subcases F, FG and FGH, than kriging. Whereas



(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.20: A comparison between Legendre and Hermite basis functions for PCE in two dimensions.

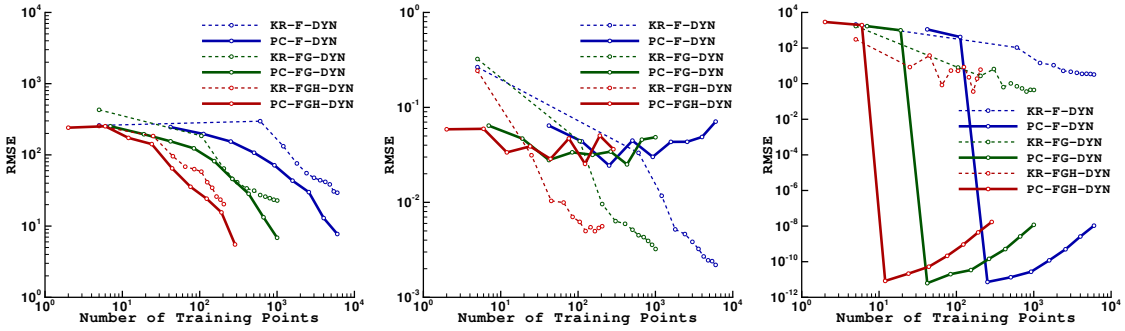


(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.21: A comparison between kriging and PCE on two-dimensional test functions.

for the rational Runge function f_2, the kriging is producing a better surrogate than PCE. The Rosenbrock function f_3 is captured exactly by PCE as expected, whereas kriging is converging much slower (as it is not a polynomial based surrogate). For all the kriging test cases five training points were added per cycle. Figure 6.22 compares kriging with PCE for the five-dimensional test cases. For f_1, PCE performs slightly better than kriging. However, for f_2, PCE produces poor results and f_3 is captured exactly as expected. In building the

kriging surrogate 600, 100, and 20 training points were selected per cycle for F, FG, and FGH test cases, respectively.



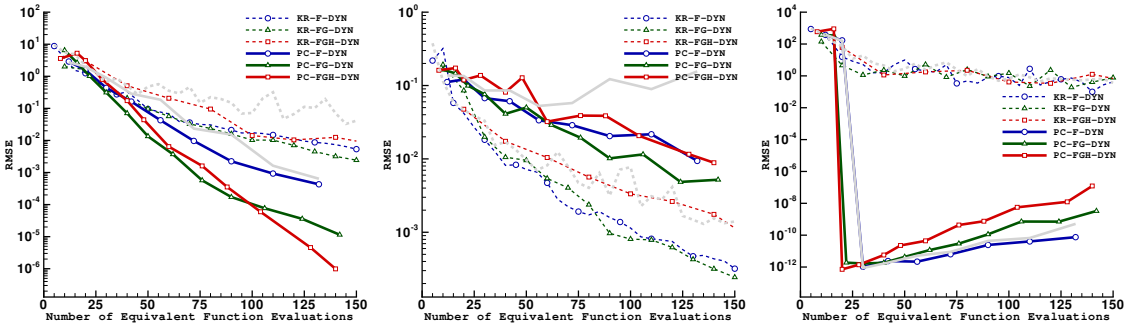
(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.22: A comparison between kriging and PCE on five-dimensional test functions.

6.5.1 On using Higher-order Derivative Information

The following discussion is aimed at providing guidelines for choosing an appropriate combination of training point selection method and incorporation of higher-order derivative information. Generally, the use of derivative information provides improved surrogate models for both kriging and polynomial chaos, irrespective of whether the training points are chosen dynamically or at random (occasionally LHS shows a deviant behavior). As discussed in section I, efficient gradient and Hessian calculation methods are available for high-fidelity physics-based simulations and they provide the means to reduce the effects of the "curse of dimensionality".

Figures 6.23 and 6.24 take into account the computational time for calculating the gradient and Hessian, and plot the model accuracy versus the number of equivalent exact

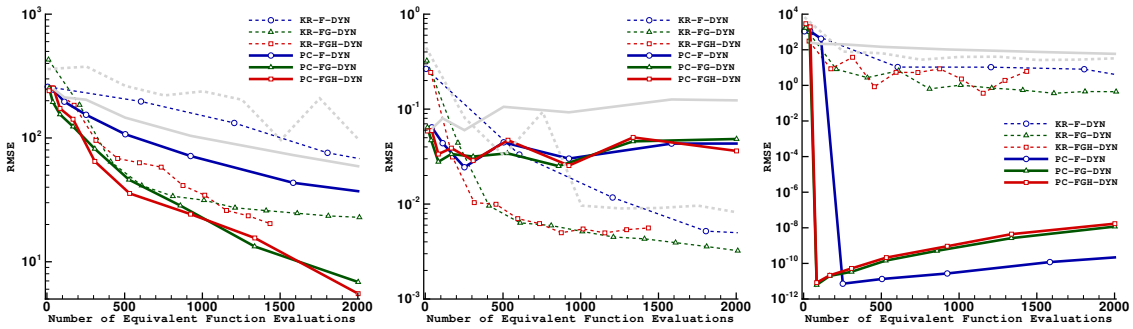


(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.23: RMSE versus the number of equivalent function evaluations for two-dimensional test functions using kriging and PCE with dynamic training point selection. Reference LHS results with function values only are shown with dashed and continuous gray lines corresponding to kriging and PCE, respectively.

function evaluations for two- and five-dimensional test functions, respectively. For example, the computational time for evaluating N data points with function values only (F), is the same as evaluating $\frac{N}{2}$ points with function and gradient values (FG), which in turn is the same as $\frac{N}{M+2}$ points with function, gradient and Hessian information (FGH), using the adjoint method as discussed in section 1.1.2.3, where M is the number of dimensions. From these figures it can be inferred that the gradient enhanced surrogates (FG) are computationally more efficient than the others (F or FGH). The Hessian enhancement does not yield convincing results as expected; however, it can be expected that the Hessian information in specific locations (such as where data points are sparsely distributed) can be advantageous than to add Hessian information to all training points as done here.

In summary, LHS and some low-discrepancy sequences have been shown to produce less accurate results than the proposed dynamic method. Additionally, the computational advantage of building gradient enhanced surrogate models is shown in Figures 6.23 and 6.24.



(a) Exponential (b) Runge (c) Rosenbrock

Figure 6.24: RMSE versus the number of equivalent function evaluations for five-dimensional test functions using kriging and PCE with dynamic training point selection. Reference LHS results with function values only are shown with dashed and continuous gray lines corresponding to kriging and PCE, respectively.

Hence, based on these observations, the dynamic training point selection in conjunction with the use of gradient information for the construction of surrogate models offers the best value.

6.5.2 Key Observations for Kriging and PCE

The following discussion summarizes the performances of kriging and PCE in terms of accuracy, robustness and computational time.

Accuracy: A definite conclusion could not be made on this aspect, as these surrogate models (kriging and PCE) perform differently for different classes of test functions. PCE features higher convergence rates for smooth and continuously differentiable functions (e.g., f_1) and polynomials (e.g., f_3). However, kriging is well suited for non-smooth, non-polynomial functions (f_2). A similar comparison for an aerodynamic test case featuring discontinuities will be shown in chapter VII.

Robustness: The general applicability of both kriging and PCE are compared below.

- PCE mandates a specific number of points for a particular order of expansion (due to its mathematical setup). It can be disadvantageous when working with a fixed computational budget or already existing data (e.g. experimental data).

- As the order of expansion increases for PCE, "wiggles" tend to appear in the PCE surrogate which is a typical behavior of higher-order polynomials which can affect the overall accuracy of the surrogate. The kriging surrogate model does not suffer from this major drawback as it is not a polynomial.

- PCE basis matrix is known to suffer from ill-conditioning (more pronounced in the regression approach). Though an oversampling factor of two or more is suggested, it does not guarantee the well-posedness of the problem.

- PCE produces good results for smooth and polynomial functions, but most physics-based simulations are neither smooth nor polynomials.

- Kriging supports the usage of both high- and low-fidelity training points [53,55,86–90] whereas PCE does not have this advantage yet.

Computational Time: PCE is cheaper to build and evaluate than kriging. This is due to comparatively intensive mathematical operations required for the construction of kriging as discussed in section 2.1. On the other hand, for PCE the most time-intensive operation is finding the inverse of the basis matrix $\boldsymbol{\psi}$ for a given set of training data.

CHAPTER VII

AERODYNAMIC DATABASE CREATION

In this chapter the benefits of the proposed training and validation framework are demonstrated for an aerodynamic problem. Finally, the kriging and polynomial chaos are applied to construct aerodynamic databases of drag and lift coefficients.

7.1 Flow Problem

The steady inviscid flow around a NACA 0012 airfoil governed by the Euler equations is solved by using a second-order accurate finite-volume approach [94, 95]. The computational mesh is shown in Figure 7.1.

The variations of the drag and lift coefficients with changes in Mach number ($0.5 \leq M_\infty \leq 1.5$) and angle of attack ($0^\circ \leq \alpha \leq 5^\circ$) are studied. An "exact" database is obtained from Euler flow solves on a Cartesian mesh of $N_t = 51 \times 51 = 2601$ equispaced nodes and is used to validate both the kriging and PCE surrogate models.

Figure 7.2 shows the gradients obtained using a discrete-adjoint approach (demonstrated to be accurate to machine precision [94, 95]) for two typical angles of attack, $\alpha = 1^\circ$ and 4°. It can be seen that the gradients are quite noisy (due to the transonic behavior of the flux limiters) and are hence counterproductive in the construction of the surrogate models.

Thus, gradients and Hessian are not used for surrogate training in this aerodynamic test case.

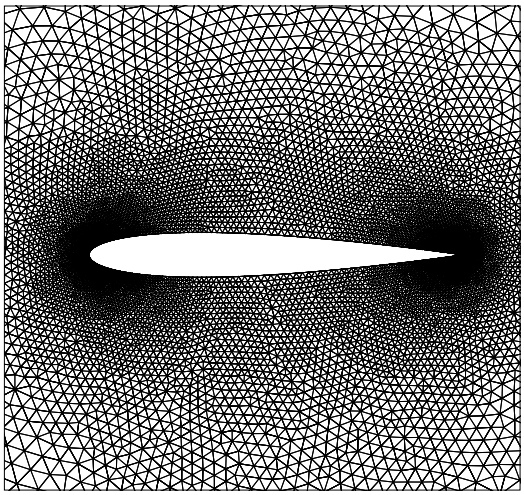


Figure 7.1: Computational mesh for NACA 0012 airfoil with 19,548 triangular elements.

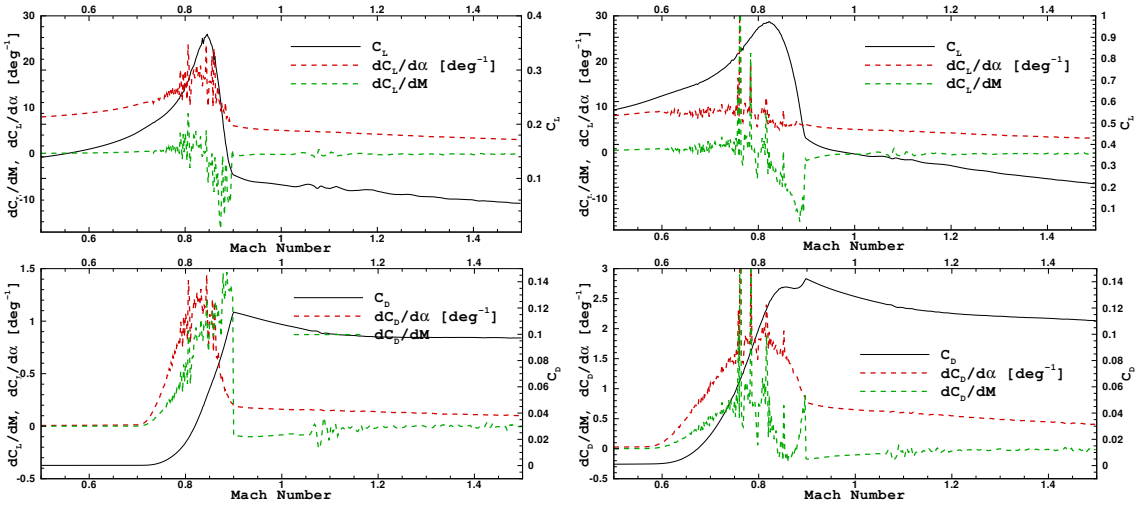


Figure 7.2: Noisy gradients for $\alpha = 1°$ (left) and $\alpha = 4°$ (right).

7.2 Validation of Proposed Error Estimates

7.2.1 Comparison with Actual Errors and Cross Validation

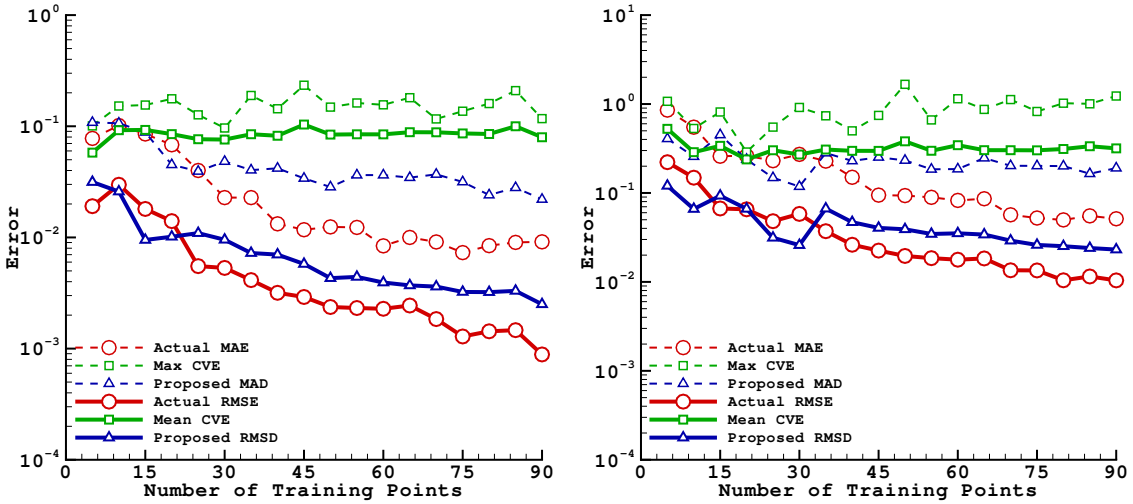


Figure 7.3: A comparison of proposed error measures (blue lines) with actual errors (red lines) and leave-one-out cross validation (green lines) for drag (left) and lift coefficients (right) using kriging.

In Figures 7.3 and 7.4, the proposed error estimates (MAD and RMSD) are compared with the actual errors (MAE and RMSE) as well as leave-one-out cross validations for kriging and PCE, respectively. The excellent agreement observed for analytical test functions can not be seen here. This is due to the reason that the kriging is better than MIR in approximating the non-smooth drag and lift functionals, thereby making the initial assumption of a more accurate local surrogate model invalid. This opens up the avenue for exploring other candidates for building local surrogate models (e.g. radial basis function, neural networks) that can approximate non-smooth functions as good as kriging (or better). Although not as accurate as kriging, the MIR produces reasonably accurate reference local surrogate models (see Figures 7.5 and 7.6). As a result, the proposed error measures are

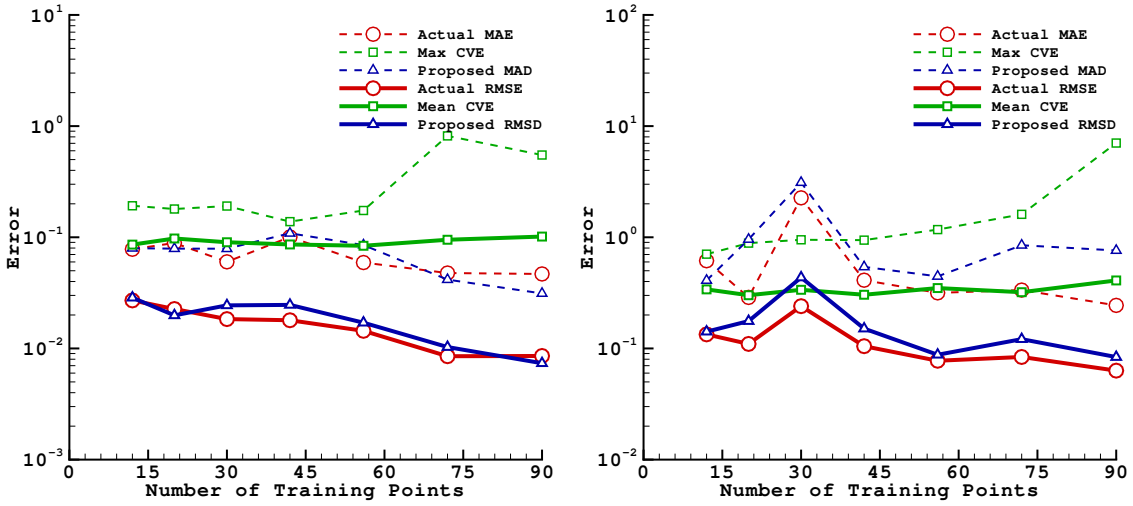


Figure 7.4: A comparison of proposed error measures (blue lines) with actual errors (red lines) and leave-one-out cross validation (green lines) for drag (left) and lift coefficients (right) using PCE.

much better in capturing the tendencies seen in global surrogate model convergence, when compared to the often used cross validation which shows misleading tendencies.

7.2.2 Comparison with Error Distributions in the Domain

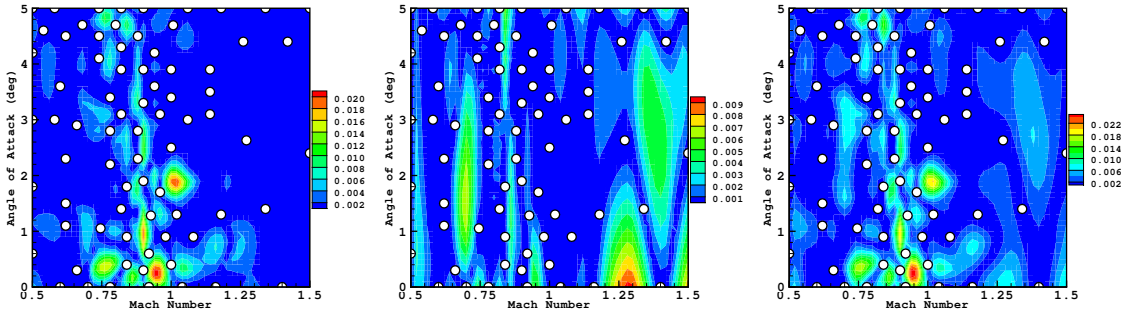


Figure 7.5: Contour plots for the drag coefficient showing the distribution of: the local surrogate model error (left), the global kriging surrogate model error (middle) and the proposed discrepancy function (right). The global and local surrogate models are built with 75 and 25 training points (white circles), respectively.

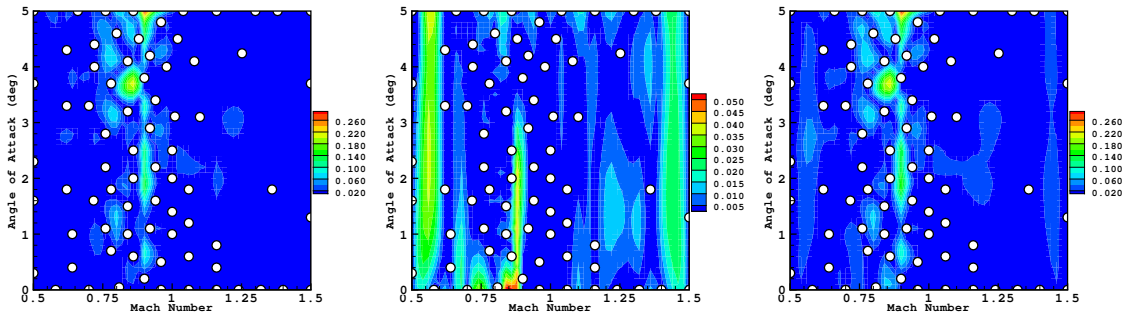


Figure 7.6: Contour plots for the lift coefficient showing the distribution of: the local surrogate model error (left), the global kriging surrogate model error (middle) and the proposed discrepancy function (right). The global and local surrogate models are built with 75 and 25 training points (white circles), respectively.

Figures 7.5 and 7.6 show the distribution of local (left) and global surrogate model error (middle), along with the proposed discrepancy function (right), for drag and lift coefficients, respectively, using kriging as the global surrogate model. The regions where the actual errors are high in the global kriging model (*i.e.,* the transonic regime) are predicted reasonably well by the discrepancy function, though the magnitudes are quite different (due to inadequate local models as mentioned above). By comparing the contour values of error, it can also be seen that the MIR local surrogate models are less accurate than the global kriging model for both the drag and lift coefficient. Another observation that can be made is that the dynamic training point selection method is adding a majority of the training points in the transonic region as expected. The kriging MSE minimization approach would not exhibit this behavior since MSE is a measure of space filling. If a better spread of training points is also preferred, the *control parameter* described in chapter V can be set to a higher value such as 1.1 or 1.2, that will enforce a more strict geometric constraint, or the vice versa when using the framework for optimizations (not studied here).

7.3 Construction of Aerodynamic Database

7.3.1 Comparing Training Point Selection Methods

Figure 7.7 compares the performance of the proposed dynamic training point selection with other DoE methods for kriging (left) and polynomial chaos (right), respectively. Here, ten separate runs are averaged and the mean, best and worst trends are shown. It can be seen that the dynamic training point selection performs better than LHS for both surrogate models and for both functionals. In Figure 7.7 (left), which corresponds to kriging,

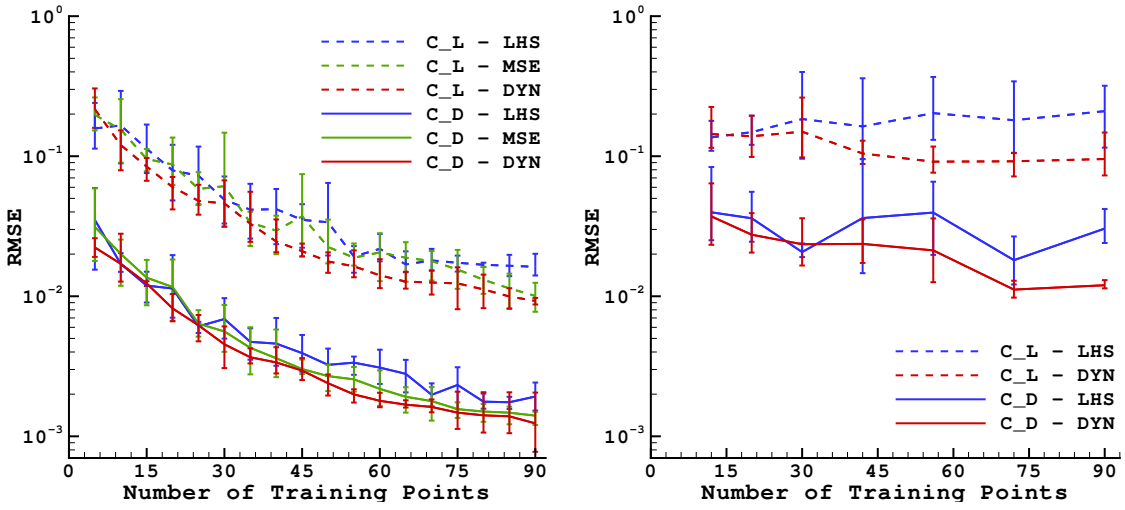


Figure 7.7: Plot of RMSE versus the number of training points for kriging (left) and PCE (right).

minimization of MSE is also compared with the other two methods. Though not as distinct as seen with analytical test functions, it can be observed that the dynamic method is consistently more monotonic and accurate than other compared methods. From Figure 7.7 (right), it can be noticed that the accuracy of PCE tends to deteriorate with increasing

order of the expansion when LHS is used. However, the dynamic training point selection prevents the early onset of such a behavior.

7.3.2 Drag and Lift Coefficient Hyper-surfaces

Figure 7.8 shows the exact and surrogate based contours for both the drag and lift coefficients. It can be inferred that the kriging model (shown in the middle column) is in good agreement with the exact model (shown in the left column) when compared to PCE (shown in the right column) which features high over- and undershoots in the domain. In addition, the kriging is able to capture the transonic behavior very well, demonstrating its ability to model non-smooth functions. Thus, for this test case the kriging proves to be a much better surrogate model than PCE.

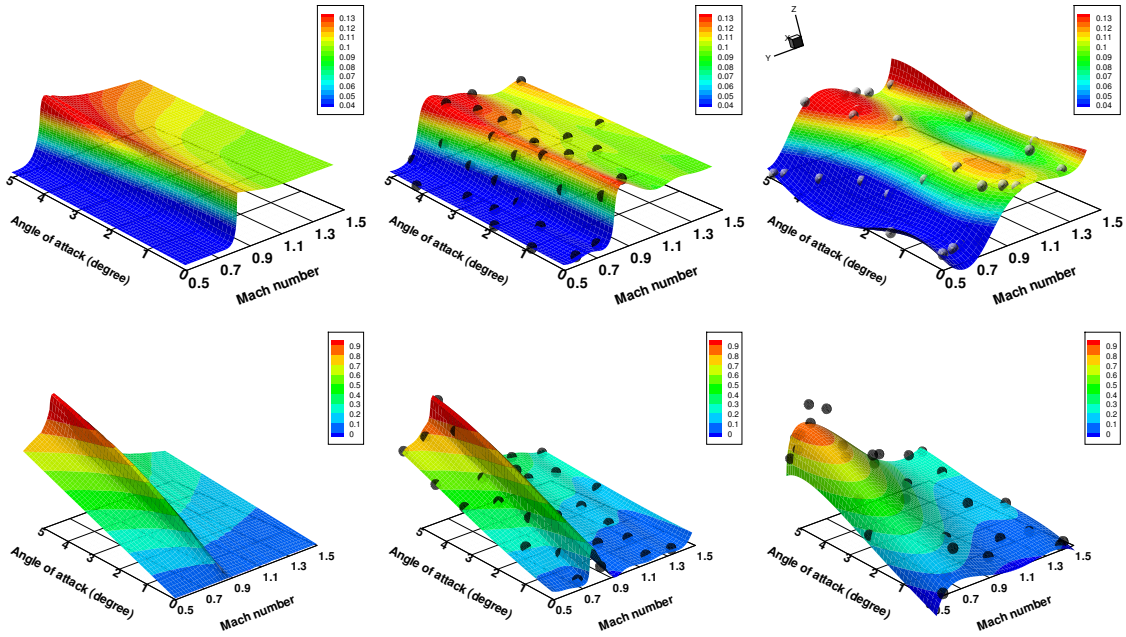


Figure 7.8: Contours of exact database (left), kriging (middle) and PCE (right) for drag (top) and lift coefficients (bottom) with 30 training points chosen with dynamic training point selection.

7.3.3 Variable-Fidelity Kriging Results

As mentioned in previous chapters, kriging supports the usage of both high- and low-fidelity training points [53,55,86–90]. The general idea is to combine trends from low-fidelity data (e.g., coarser meshes, less sophisticated models) with interpolations of high-fidelity data (e.g., finer meshes, better models, experimental data). This approach can help to reduce the time taken to build an accurate surrogate model, since the low-fidelity data can be obtained much faster. In this work a simple cokriging with cross-covariances between the low- and high-fidelity data is used [55, 89]. The low-fidelity data is calculated on a mesh with only 4,433 triangular elements (not shown), which is roughly four times cheaper to solve than the mesh shown in Figure 7.1 on which the high-fidelity data is calculated.

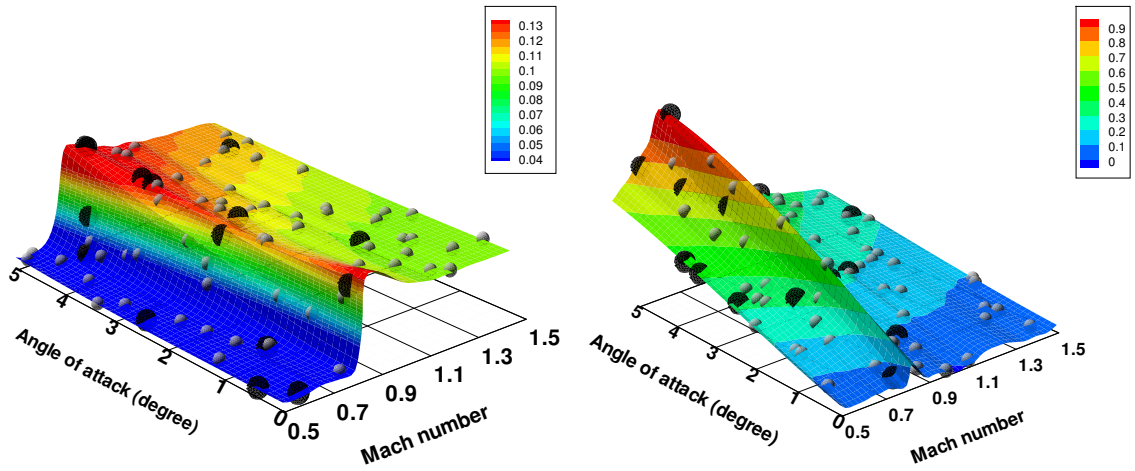


Figure 7.9: Kriging contour plots demonstrating the use of variable-fidelity data for drag (left) and lift (right) coefficients.

Figure 7.9 shows the variable-fidelity kriging contours for the drag and lift databases. Here, 15 high-fidelity and 60 low-fidelity training points were used in the construction of the kriging surrogate and the training point locations are shown as spheres, where the dark

spheres are high-fidelity training points picked dynamically and the smaller gray spheres are low-fidelity training points picked via LHS. The overall computational cost is roughly equivalent to the construction of the kriging surrogate model using 30 high-fidelity training points. An improvement in the accuracy of the model can be noticed when using the variable-fidelity data by comparing Figures 7.8 and 7.9 as well as from Table 7.1, for roughly the same computational cost.

Table 7.1: RMSE comparisons for different kriging models.

RMSE	High-fidelity (30 high-fidelity points)	Variable-fidelity (15 high-fidelity and 60 low-fidelity points)
Drag Coefficient C_D	0.39×10^{-2}	0.31×10^{-2}
Lift Coefficient C_L	0.35×10^{-1}	0.18×10^{-1}

CHAPTER VIII

DESIGN IN THE PRESENCE OF UNCERTAINTY

"A product should be designed in such a way that makes its performance insensitive to variation in variables beyond the control of the designer" [43]
—Genichi Taguchi

When a system is designed allowances must be made to accommodate likely variations that can disrupt the nominal performance, such as inaccuracies in modeling, uncertainties in manufacturing processes, operating environment and boundary conditions. When such variations or uncertainties are not accounted for in the design process, as in the case of a deterministic optimization practice, a degraded performance of the optimized design is inevitable.

In a heavily optimized design, the optimum solution tends to lie either at the extremum of the objective function or at the constraint boundary [43]. Thus, a deterministic optimum is a vulnerable solution with a greater likelihood of violating the design requirements: even small perturbations in the input can lead to a poor performance or failure of the design. To alleviate this, a factor of safety is traditionally incorporated into the constraints. For instance, in a structural design problem a stress constraint originally of the form $g(\boldsymbol{d}) = \frac{\sigma}{\sigma_{max}} - 1 \leq 0$ is treated as $g(\boldsymbol{d}) = F_s \cdot \frac{\sigma}{\sigma_{max}} - 1 \leq 0$, where $\boldsymbol{d}$ is the vector of design variables and F_s is the factor of safety. The factor of safety serves to move the optimum away from

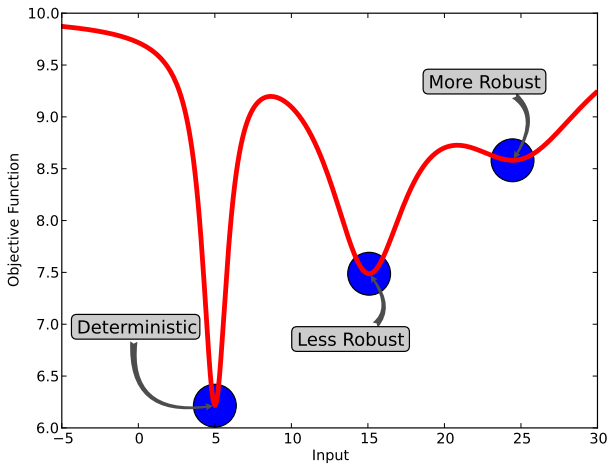


Figure 8.1: An illustration for the sensitivity of optimum designs to input variations.

the constraint boundary by a considerable distance, thereby preventing the design from an imminent failure. However, the reality is that the designer assigning a factor of safety is seldom aware of the real effects of uncertainty and predominantly produces an over- or under-conservative design leading to weight penalty or vulnerable products, respectively. With the continuous evolution of radically new types of design, it is increasingly difficult for a designer to assign an adequate factor of safety [43].

Due to these reasons, uncertainty quantification (UQ) has evolved as a field of interest, where the goal is to account for the effect of uncertainties through a modified optimization process known as *optimization under uncertainty* (OUU). OUU can be subdivided into two fields as *robust design optimization* (RDO) and *reliability based design optimization* (RBDO) [11–13]. Though these two fields share many common attributes, they differ in their objectives: RDO techniques are used to produce a design that is more robust (less sensitive) to design parameter anomalies, whereas the goal of RBDO is to minimize the probability of failure of the system. This work focuses on methods to produce robust designs

which involves finding an optimum that is less sensitive to input variability as opposed to a deterministic optimum that can exhibit a sharp change in the objective function value for minor perturbations in the inputs (see Figure 8.1). Usually, a robust solution is obtained at the expense of an increased cost function [43] (also illustrated in Figure 8.1).

Stages in OUU: There are three stages in design optimization under uncertainty [43]:

(i) *Identification, modeling and representation of uncertainties* to translate the available data into mathematical models that are either probabilistic or non-probabilistic in nature,

(ii) *Propagation of uncertainties* through computer models to quantify their impact on system performance,

(iii) *Formulation and solution of an optimization problem* with appropriate objective and constraint functions ensuring that the optimum solution is robust against uncertainties.

Brief overviews of these stages are provided in the remainder of this chapter. For an in-depth discussion on the subject, the reader is referred to works in the literature [14–18].

8.1 Uncertainty Modeling

The modeling of uncertainties begins with the treatment of inputs as random variables. Uncertainties can be classified as *aleatory* and *epistemic* uncertainties [14,43]. Aleatory uncertainties are the inherent randomness or variation in physical system, input parameters and variables, or operating environment [14]. For example, operating conditions are predominantly dissimilar to the ones used in design calculations and typically fluctuate around some mean value. Epistemic uncertainties arise due to the lack of knowledge or information

in any phase or activity of the modeling process. It is not an inherent property of the system and thus can be eliminated (or converted to aleatory form) when sufficient data becomes available. As an example, situations can arise where only the bounds or intervals of the uncertain random variables are known (e.g. manufacturing tolerances) whereas the underlying probability distribution or other statistical parameters within the interval are unknown (unlike aleatory random variables).

8.1.1 Probabilistic Modeling

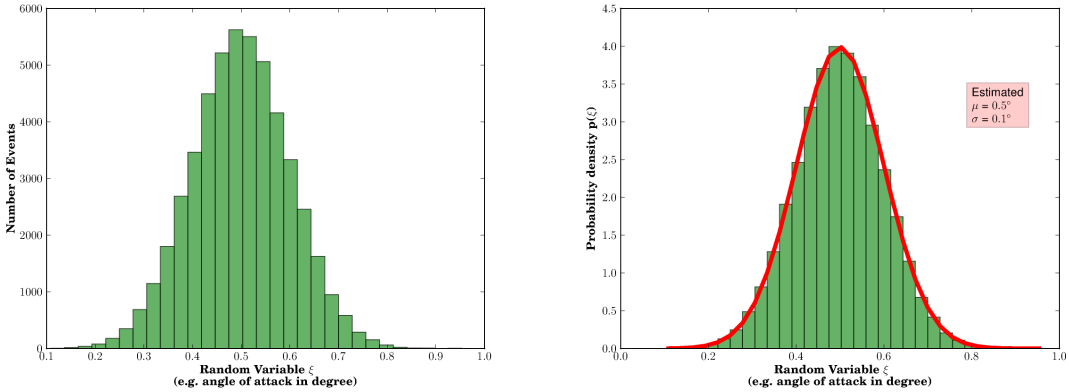


Figure 8.2: An illustration for the use of probabilistic methods. A histogram of the available angle of attack data (left) is shown with the fitted Gaussian probability density function (right) whose parameters (μ and σ) are estimated from the available data.

The use of probabilistic methods to model uncertainties is possible when sufficient data is available. When field data is available a probability density function can be fit to the available data. Alternatively, a distribution function (Gaussian, log-normal, exponential, etc.) can be assumed and its parameters can be estimated from the available data. Figure 8.2 shows an example of fitting the available angle of attack information to a normal distribution function. The mean and standard deviation are estimated from the data and

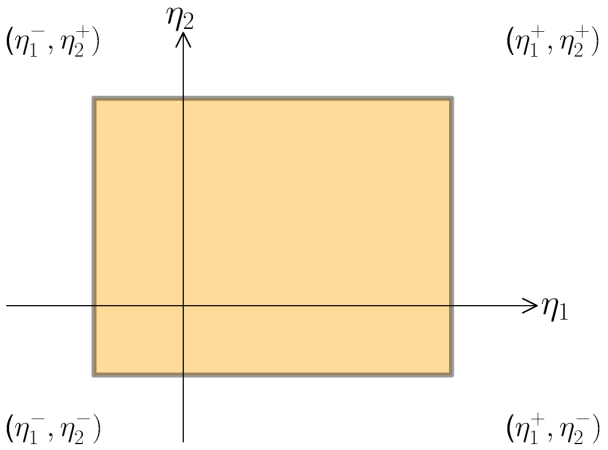


Figure 8.3: An illustration for bounds on input variables.

can be used to define the probability density function (PDF) of the random variable as $p(\xi) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\xi-\mu)^2}{2\sigma^2}}$ (Gaussian PDF). The angle of attack (and similarly any other variables) can now be treated as random variable in optimization formulations.

Distributions should be assumed with caution when only limited data is available to the designer for assessment [43]. For example, a normal distribution can not be assumed for Young's modulus, as a Gaussian distribution supports $[-\infty, +\infty]$ and a zero probability would mean a negative Young's modulus which is unrealistic. In reliability calculations, the probability of failure is estimated near the tail end of the distribution, which can be very erroneous if a wrong distribution structure is assumed.

8.1.2 Non-Probabilistic Modeling

The use of probability theory to model the distribution of input uncertainties is questionable when not enough information is available. This naturally leads into non-probabilistic

approaches such as possibility theory, interval analysis, convex modeling and evidence theory [43]. The simplest non-probabilistic approach is perhaps the interval representation of input uncertainties. The input random variable is represented by the interval $[\eta^-, \eta^+]$ where η^- and η^+ denote the lower and upper bounds on the input random variable, respectively. This scenario is illustrated with a two-variable example in Figure 8.3. The random process can take any value within the specified interval but the underlying probability distribution is unknown. The input bounds are processed into the analysis model to construct bounds on the output quantity of interest [43].

In summary, probabilistic approaches are apt for modeling aleatory uncertainties featuring an abundance of data and non-probabilistic approaches are suitable for epistemic uncertainties suffering a data scarcity.

8.2 Uncertainty Propagation

In this section, approaches for the propagation of input uncertainties are discussed. The goal is to quantify the uncertainties and model the input–output relationship through numerical methods. The aleatory variables are denoted as $\boldsymbol{\xi}$ and realizations of aleatory variables from their probability distribution are represented as $\boldsymbol{\alpha}$. The epistemic variables are denoted as $\boldsymbol{\eta}$ and their realizations within the specified bounds are denoted as $\boldsymbol{\beta}$.

8.2.1 Propagation of Aleatory Uncertainties

Sufficient input data is generally available for the analysis of aleatory uncertainties. Thus, probabilistic methods which mandate multitude of realizations are commonly used for computing the statistics based on the input probability distribution. In other words, the distribution type of the input random variables (e.g. $\boldsymbol{\alpha} \sim \mathcal{N}(\bar{\boldsymbol{\xi}}, \boldsymbol{\sigma}_{\boldsymbol{\xi}}^2)$) are known, whereas

the functional dependence $f(\boldsymbol{\xi})$ on these random variables is not known, and are modeled using numerical simulations.

Monte Carlo Simulation: The simplest approach to achieve uncertainty propagation is the Monte Carlo simulation (MCS). When information such as the input mean, standard deviation and PDF of design variables and other parameters (collectively known as inputs) are available, the statistics of the output function can be computed using MCS. In this method, numerous samples (realizations) $\boldsymbol{\alpha}^{(j)}$ are generated from the distribution $p(\boldsymbol{\xi})$ of the input random variables and the response function or simulation code is evaluated. This leads to the following estimates for the mean:

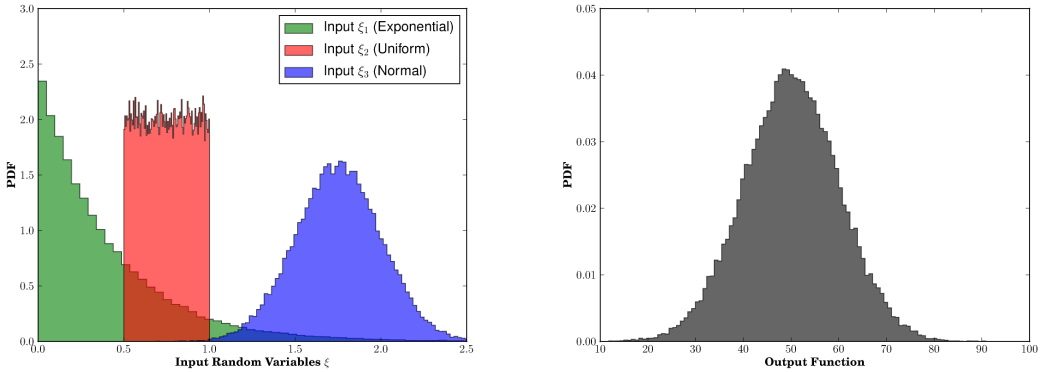
$$\bar{f} = \mu_f = \frac{1}{\widetilde{N}} \sum_{j=1}^{\widetilde{N}} f(\boldsymbol{\alpha}^{(j)}), \tag{8.1}$$

and variance of the output quantity:

$$\sigma_f^2 = \vartheta_f = \frac{1}{\widetilde{N}} \sum_{j=1}^{\widetilde{N}} (f(\boldsymbol{\alpha}^{(j)}) - \bar{f})^2, \tag{8.2}$$

where $\widetilde{N}$ is the number of Monte Carlo samples. MCS can be used on any output function $f(\boldsymbol{\xi})$ and is hence non-intrusive in nature.

Inexpensive Monte Carlo Simulation: It is known that repeated evaluations of the exact function, $f(\boldsymbol{\xi})$, is prohibitively expensive or impractical most of the times. To overcome this problem, surrogate models can be built and inexpensively probed to yield approximated output function values $\widehat{f}(\boldsymbol{\xi})$ for the calculation of approximate means and variances. The required number of function values (training points) for building an accurate surrogate is usually way less than the number required for statistically converged Monte Carlo simulation. In this work, the domain over which the surrogate model is built is taken to be *three standard deviations* in all aleatory input dimensions from the specified mean value



(a) Set of input variables with distributions

(b) Distribution of the output function

Figure 8.4: An illustration for modeling the input-output relationship of uncertainties.

i.e., the surrogate domain is: $[\bar{\boldsymbol{\xi}} \pm 3 \cdot \boldsymbol{\sigma_\xi}]$. This implies that for normally distributed input variables more than 99 % of all samples (aleatory realizations) fall within the surrogate domain and the less accurate extrapolation capabilities of the surrogate model only need to be employed for a small fraction of the samples. A larger domain can be specified for the surrogate (e.g. 6σ from mean) to account for even remote possibilities, but is not used in this work (recommended for reliability calculations). In this work, polynomial chaos and kriging surrogate models described in chapter II and trained using the dynamic framework described in chapter V are employed for aleatory uncertainty propagation. Both surrogate models have the capability to incorporate higher-order derivative information (gradients and Hessian), however, the surrogates are built using function values only to reduce the complexity in presenting the results.

8.2.2 Propagation of Epistemic Uncertainties

Epistemic uncertainties represent the lack of knowledge about the appropriate value to use [16]. For instance, manufacturers typically provide intervals in terms of tolerances (e.g.

$\pm$ 0.5 mm for length of a bolt), but the exact values are not known or cannot be guaranteed whatsoever. Here, the goal is to have bounds on the output quantity of interest or to determine worst case scenarios (e.g. maximum possible constraint violation, least possible lift), in order to minimize the sensitivity or variation of the design with respect to these uncertainties. In a situation where only the input interval $I(\boldsymbol{\eta}) = [\boldsymbol{\eta}^-, \boldsymbol{\eta}^+] = [\bar{\boldsymbol{\eta}} - \boldsymbol{\tau}, \bar{\boldsymbol{\eta}} + \boldsymbol{\tau}]$ is known, the above assessment can be accomplished in most straightforward manner by either one of the two methods discussed below.

Sampling: An extensively sampling of the interval $I(\boldsymbol{\eta})$ can be done and the analysis of the resulting outputs $f(\boldsymbol{\eta})$ is carried out to determine the extreme values (worst and best cases). However, the computational burden can be prohibitive in the case of high-fidelity physics-based simulations and for higher-dimensional spaces. As a remedy, a surrogate model over the domain $I(\boldsymbol{\eta})$ can be constructed (similar to aleatory uncertainties), which can then be sampled using inexpensive Monte Carlo simulations (IMCS). However, with increasing number of variables, building an accurate surrogate model requires thousands of simulation outputs (referred to as the curse of dimensionality) and quickly becomes prohibitively expensive as well.

Bound-Constrained Optimization: A bound- or box-constrained optimization (BCO) [96–98] can be employed to find the worst and best behavior of the constraint/objective function within the specified interval $I(\boldsymbol{\eta})$. A gradient-based BCO scales mildly with the number of input variables, making it computationally more attractive than MCS for quantifying the effect of epistemic uncertainties, particularly for larger problems. In BCO, the problem of finding the extreme value of the function, f^*, (and the constraints, g_i^*) within the interval

$I(\boldsymbol{\eta})$ can be cast as follows:

$$\begin{aligned} &\underset{\boldsymbol{\beta}}{\text{minimize/maximize}} \quad f = f(\boldsymbol{\eta}), \\ &\text{subject to} \qquad\qquad\quad \boldsymbol{\beta} \in I(\boldsymbol{\eta}) = [\bar{\boldsymbol{\eta}} - \boldsymbol{\tau}, \bar{\boldsymbol{\eta}} + \boldsymbol{\tau}]. \end{aligned} \tag{8.3}$$

In most cases the extremum occurs at either the upper or lower bound of the interval due to the quasi-linearity of the typically small space described by $I(\boldsymbol{\eta})$. Thus, BCO typically takes only about five to ten simulation output and gradient evaluations to reach f^* and is used throughout this work to propagate epistemic uncertainties. An L-BFGS [99, 100] algorithm which utilizes function and gradient information is used to solve the BCO problem given by Eq. (8.3).

8.2.3 Propagation of Mixed Uncertainties

Table 8.1 summarizes the four typical methods that can be employed for the propagation of mixed epistemic and aleatory uncertainties along with their corresponding approximate simulation requirements. The computational requirements can be interpreted assuming an approximate range of values for: (*i*) the number of Monte Carlo sample points ($\widetilde{N} = 10^5 - 10^8$), (*ii*) the number of surrogate training points ($N = 50 - 5000$), and (*iii*) the number of simulation output evaluations (with gradients) for a BCO ($n = 10 - 100$). The most straightforward way to propagate mixed uncertainties is to carry out a

Table 8.1: Methods for optimization under mixed uncertainties along with their simulation requirements per iteration.

Method	Propagation Method		No. of Evaluations		Total per iteration
	Aleatory	Epistemic	Aleatory	Epistemic	
1	MCS	MCS	$\widetilde{N}_1$	$\widetilde{N}_2$	$\widetilde{N}_1\widetilde{N}_2$
2	MCS	BCO	$\widetilde{N}$	n	$\widetilde{N} \cdot n$
3	IMCS	IMCS	N_1	N_2	$N_1 \cdot N_2$
4	IMCS	BCO	N	n	$N \cdot n$

nested-sampling approach (Method 1), where for each aleatory random variable realization, $\boldsymbol{\alpha}^{(i)},\ i = 1, 2, \ldots, \widetilde{N}$, drawn from its probability distribution $p(\boldsymbol{\xi})$, a Monte Carlo sampling (or LHS for a better search performance) has to be performed over the epistemic variable realizations $\boldsymbol{\beta}^{(j)},\ j = 1, 2, \ldots, \widetilde{N}$, to determine the extreme behavior. Method 2 uses BCO for epistemic uncertainties and is less expensive than Method 1, but can still represent an enormous computational endeavor for the aleatory uncertainties, and is hence impractical for high-fidelity simulations. It can be seen that the last two methods employing surrogate models for uncertainty propagation are several orders of magnitude cheaper. Method 3 turns out to be the cheapest for smaller problems ($M = M_\xi + M_\eta \approx 1 - 6$), but can easily suffer from the curse of dimensionality and thus lacks robustness, whereas it can be inferred that Method 4 is still computationally manageable for bigger problem sizes. Thus, this work preferably employs the IMCS-BCO approach (Method 4) for the propagation of mixed uncertainties in a robust optimization problem. The employed IMCS-BCO framework has been developed by Lockwood *et al.* [96–98] and Rumpfkeil [101]. A detailed discussion of steps involved is given in section 8.3.3.

The computational requirements in Table 8.1 are given for just one iteration of the numerical solution for the robust optimization problem. If the optimizer requires $\mathcal{K}$ iterations to converge, the number in the last column has to be multiplied with $\mathcal{K}$ to obtain an approximation for the number of simulation evaluations needed. As a last remark, a deterministic gradient-based optimization requires only on the order of $2\mathcal{K}$ (one function and one gradient evaluation per iteration) simulation output evaluations to reach the optimum.

In the mixed uncertainty problem, the trial design variable vector $\boldsymbol{d}$ is comprised of both aleatory and epistemic components *i.e.,* $\boldsymbol{d} = [\bar{\boldsymbol{\xi}}, \bar{\boldsymbol{\eta}}]$, where $\bar{\boldsymbol{\xi}}$ represents the mean of aleatory uncertainties and $\bar{\boldsymbol{\eta}}$ refers to the midpoint of the epistemic uncertainty bounds. In this

work the aleatory uncertainties are assumed to be *statistically independent* and *normally distributed* with $\boldsymbol{\alpha} \sim \mathcal{N}(\bar{\boldsymbol{\xi}}, \boldsymbol{\sigma}_{\boldsymbol{\xi}}^2)$. Equations for correlated and/or non-normally distributed aleatory variables can also be derived; however, the analysis and resulting equations become more complex [102] and are beyond the scope of this work. In addition, the assumed input standard deviation $\boldsymbol{\sigma}$ for aleatory variables as well as the upper and lower bounds for epistemic variables defined by $\boldsymbol{\tau}$ are treated as fixed throughout the optimization for simplicity (could be easily changed).

8.3 Optimization Problem Formulation

8.3.1 Deterministic Optimization

A conventional constrained optimization problem for an objective function, $\mathcal{J}$, that is a function of input variables, $\boldsymbol{d}$, state variables, $\boldsymbol{q}(\boldsymbol{d})$, and simulation outputs, $f(\boldsymbol{d}) = F(\boldsymbol{q}(\boldsymbol{d}), \boldsymbol{d})$, can be written as:

$$\begin{aligned} \underset{\boldsymbol{d}}{\text{minimize}} \quad & J = J(f, \boldsymbol{q}, \boldsymbol{d}), \\ \text{subject to} \quad & R(\boldsymbol{q}, \boldsymbol{d}) = 0, \\ & g(f, \boldsymbol{q}, \boldsymbol{d}) \leq 0. \end{aligned} \tag{8.4}$$

Here, the state equation residuals, R, are expressed as an equality constraint, and other system constraints, g, are represented as general inequality constraints. In the case where the input variables are precisely known all functions dependent on $\boldsymbol{d}$ are deterministic. However, due to input uncertainties all functions in Eq. (8.4) can no longer be treated deterministically.

8.3.2 Robust Optimization

The setup of a robust optimization problem under mixed uncertainties is discussed below.

Objective Function: A robust objective function can be written in terms of the mean values of the functional outputs μ_{f*} and variance σ^2_{f*}. The robust optimization problem is minimizing the weighted sum of mean extremum and variance of the function. Mathematically, the objective function assumes the form:

$$\mathcal{J} = w_1 \mu_{f*} + w_2 \, \sigma^2_{f*}, \tag{8.5}$$

where w_1 and w_2 are some user specified weights. In this work, the weights w_1 and w_2 are set to one. The asterisk $(*)$ refers to the extremum of the BCO problem.

Constraint Functions: The state equation residual equality constraint, R, is deemed satisfied for all values of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. The inequality constraints can be cast into a probabilistic statement such that the probability that the constraints are satisfied is greater than or equal to a desired or specified probability P_k. The constraints are written as a function of mean values and their standard deviations [103, 104]:

$$g^r = g(\mu_{f*}, \boldsymbol{q}, \boldsymbol{\xi}, \boldsymbol{\eta}) + k\sigma_{f*} \leq 0, \tag{8.6}$$

where k is the number of standard deviations σ_{g*} that the constraint g must be displaced in order to achieve the required P_k.

Problem Formulation: Lastly, the deterministic optimization problem given by Eq. (8.4) can be recast into a robust design optimization problem [102, 105] as follows:

$$\begin{aligned} \underset{\boldsymbol{\xi},\boldsymbol{\eta}}{\text{minimize}} \quad & \mathcal{J} = \mathcal{J}(\mu_{f*}, \sigma^2_{f*}, \boldsymbol{q}, \boldsymbol{\xi}, \boldsymbol{\eta}), \\ \text{subject to} \quad & R(\boldsymbol{q}, \boldsymbol{\xi}, \boldsymbol{\eta}) = 0, \\ & g^r = g(\mu_{f*}, \boldsymbol{q}, \boldsymbol{\xi}, \boldsymbol{\eta}) + k\sigma_{f*} \leq 0. \end{aligned} \tag{8.7}$$

Optimization Software: The software package IPOPT (Interior Point OPTimizer) [106] for large-scale nonlinear constrained optimization is used for the solution of the robust optimization problem given by Eq. (8.7). IPOPT allows users to impose bound or box constraints on the design variables, which can be very helpful in ensuring the stability of the simulation output analysis by preventing the exploration of too extreme regions of the design space.

8.3.3 Robust Optimization Framework

The steps involved in robust optimization under mixed uncertainties [96–98, 101] are detailed here (see Figure 8.5). Surrogate models are built to propagate aleatory uncertainties and bound-constrained optimizations are used to propagate epistemic uncertainties.

1. Initialize: The main optimizer IPOPT (for the robust optimization problem discussed in section 8.3.2) provides a trial design variable vector $\boldsymbol{d} = [\bar{\boldsymbol{\xi}}, \bar{\boldsymbol{\eta}}]$ at each iteration, based on which the surrogate domain is determined as: $[\bar{\boldsymbol{\xi}} \pm 3 \cdot \boldsymbol{\sigma}_{\boldsymbol{\xi}}]$, from which surrogate training point locations $\boldsymbol{\alpha}^{(i)},\ i = 1, \ldots, N$, are selected (using the dynamic training point selection framework [107, 108]). The interval for the bound-constrained optimization for epistemic random variables is represented as $I(\boldsymbol{\eta}) = [\boldsymbol{\eta}^-, \boldsymbol{\eta}^+] = [\bar{\boldsymbol{\eta}} - \boldsymbol{\tau}, \bar{\boldsymbol{\eta}} + \boldsymbol{\tau}]$.

2. Propagate epistemic effects: For each surrogate training point $\boldsymbol{\alpha}^{(i)},\ i = 1, \ldots, N$, a BCO problem is solved for determining the worst or best epistemic realization $\boldsymbol{\beta}^{*(i)}$ within the interval $I(\boldsymbol{\eta})$. Mathematically, this refers to the determination of the extremum f^* of the output function within the interval $I(\boldsymbol{\eta})$:

$$\begin{aligned} &\underset{\boldsymbol{\beta}}{\text{minimize/maximize,}} && f = f(\boldsymbol{\alpha}^{(i)}, \boldsymbol{\beta}) \\ &\text{subject to} && \boldsymbol{\beta} \in I(\boldsymbol{\eta}). \end{aligned} \tag{8.8}$$

The aleatory variables remain fixed during the BCO process, while the epistemic variables are allowed to change within the specified bounds. This completes the propagation of epistemic uncertainties. The BCO problem needs only a few exact function and gradient evaluations to reach the extremum.

3. Obtain surrogate training data: The exact function $f(\boldsymbol{\xi}, \boldsymbol{\eta})$ is evaluated at $(\boldsymbol{\alpha}^{(i)}, \boldsymbol{\beta}^{*(i)})$, $i = 1, \ldots, N$, and the data is used to train the surrogate model. Note that, if the number of aleatory variables is large, the surrogate suffers from the curse of dimensionality, *i.e.,* tens of thousands of BCO results may be required as input training data to the surrogate.

4. Propagate aleatory effects: Once the surrogate model is built using the training data, it can be probed inexpensively to yield the output statistics (e.g. mean μ_{f*} and variance σ^2_{f*}).

5. Update objective/constraints: The aleatory statistics are used to update the objective function and constraints defined in section 8.3.2.

Steps (1) to (5) are continued until meeting user-specified stopping criteria for the robust optimization loop.

8.3.4 Gradient Evaluation

8.3.4.1 Aleatory Gradients

The gradient of the objective function with respect to design variables associated with aleatory uncertainties (random variables) is given by [101]:

$$\frac{d\mathcal{J}}{d\boldsymbol{\xi}} = \frac{\partial \mathcal{J}}{\partial \mu_{f*}} \frac{d\mu_{f*}}{d\boldsymbol{\xi}} + \frac{\partial \mathcal{J}}{\partial \vartheta_{f*}} \frac{d\vartheta_{f*}}{d\boldsymbol{\xi}} = w_1 \frac{d\mu_{f*}}{d\boldsymbol{\xi}} + w_2 \frac{d\vartheta_{f*}}{d\boldsymbol{\xi}}, \tag{8.9}$$

where the mean μ_{f*} and variance ϑ_{f*} are computed using the surrogate model (kriging or polynomial chaos). The mean extremum of the simulation output is approximated as:

$$\mu_{f*} \approx \frac{1}{\widetilde{N}} \sum_{k=1}^{\widetilde{N}} \widehat{f*}(\boldsymbol{\alpha}^k). \tag{8.10}$$

The derivative of the mean extremum with respect to aleatory variables can be calculated as:

$$\frac{d\mu_{f*}}{d\boldsymbol{\xi}} \approx \frac{1}{\widetilde{N}} \sum_{k=1}^{\widetilde{N}} \frac{d\widehat{f*}(\boldsymbol{\alpha}^k)}{d\boldsymbol{\alpha}^k} \frac{d\boldsymbol{\alpha}^k}{d\boldsymbol{\xi}} = \frac{1}{\widetilde{N}} \sum_{k=1}^{\widetilde{N}} \frac{d\widehat{f*}(\boldsymbol{\alpha}^k)}{d\boldsymbol{\alpha}^k}, \tag{8.11}$$

where $\frac{d\widehat{f*}(\boldsymbol{\alpha}^k)}{d\boldsymbol{\alpha}^k}$ is obtained from the surrogate models. Likewise, the variance and its derivative can be approximated as follows:

$$\vartheta_{f*} \approx \left(\frac{1}{\widetilde{N}} \sum_{k=1}^{\widetilde{N}} \widehat{f*}^2(\boldsymbol{\alpha}^k) \right) - \mu_{f*}^2 \tag{8.12}$$

$$\frac{d\vartheta_{f*}}{d\boldsymbol{\xi}} \approx \left(\frac{2}{\widetilde{N}} \sum_{k=1}^{\widetilde{N}} \widehat{f*}(\boldsymbol{\alpha}^k) \frac{d\widehat{f*}(\boldsymbol{\alpha}^k)}{d\boldsymbol{\alpha}^k} \right) - 2\mu_{f*} \frac{d\mu_{f*}}{d\boldsymbol{\xi}}. \tag{8.13}$$

8.3.4.2 Epistemic Gradients

The gradient of the objective function with respect to design variables associated with epistemic uncertainties (random variables) is given by:

$$\frac{d\mathcal{J}}{d\boldsymbol{\eta}} = \frac{\partial \mathcal{J}}{\partial \mu_{f*}} \frac{d\mu_{f*}}{d\boldsymbol{\eta}} + \frac{\partial \mathcal{J}}{\partial \vartheta_{f*}} \frac{d\vartheta_{f*}}{d\boldsymbol{\eta}} = w_1 \frac{d\mu_{f*}}{d\boldsymbol{\eta}} + w_2 \frac{d\vartheta_{f*}}{d\boldsymbol{\eta}}. \tag{8.14}$$

In this case, the calculation of $\frac{d\mu_{f*}}{d\boldsymbol{\eta}}$ and $\frac{d\vartheta_{f*}}{d\boldsymbol{\eta}}$ is not simple because moving the midpoint of the epistemic intervals will lead in general to different extrema for the training points and thus to a different surrogate model, which when sampled provides different values for μ_{f*} and ϑ_{f*}. In comparison, the aleatory gradient was easy to obtain because the same surrogate model is used and only the change in sample points (random realizations) has to be accounted. In this work the following approximations are used [101]:

$$\frac{d\mu_{f*}}{d\boldsymbol{\eta}} \approx \left. \frac{df}{d\boldsymbol{\eta}} \right|_{(\boldsymbol{\xi}=\bar{\boldsymbol{\xi}}, \boldsymbol{\eta}=\bar{\boldsymbol{\eta}})} \quad \text{and} \quad \frac{d\vartheta_{f*}}{d\boldsymbol{\eta}} \approx 0, \tag{8.15}$$

i.e., the derivative of the mean extremum μ_{f*} with respect to the epistemic variables $\boldsymbol{\eta}$, is approximated by the derivative of the function f with respect to $\boldsymbol{\eta}$, evaluated at the mean values of the aleatory variables and midpoints of the interval for the epistemic variables. Generally, this derivative is non-zero since for the epistemic optimizations via BCO, the extreme value is typically encountered at the interval bound. Since the variances are small in comparison with the mean values, their sensitivities are neglected: $\frac{d\vartheta_{f*}}{d\boldsymbol{\eta}} \approx 0$.

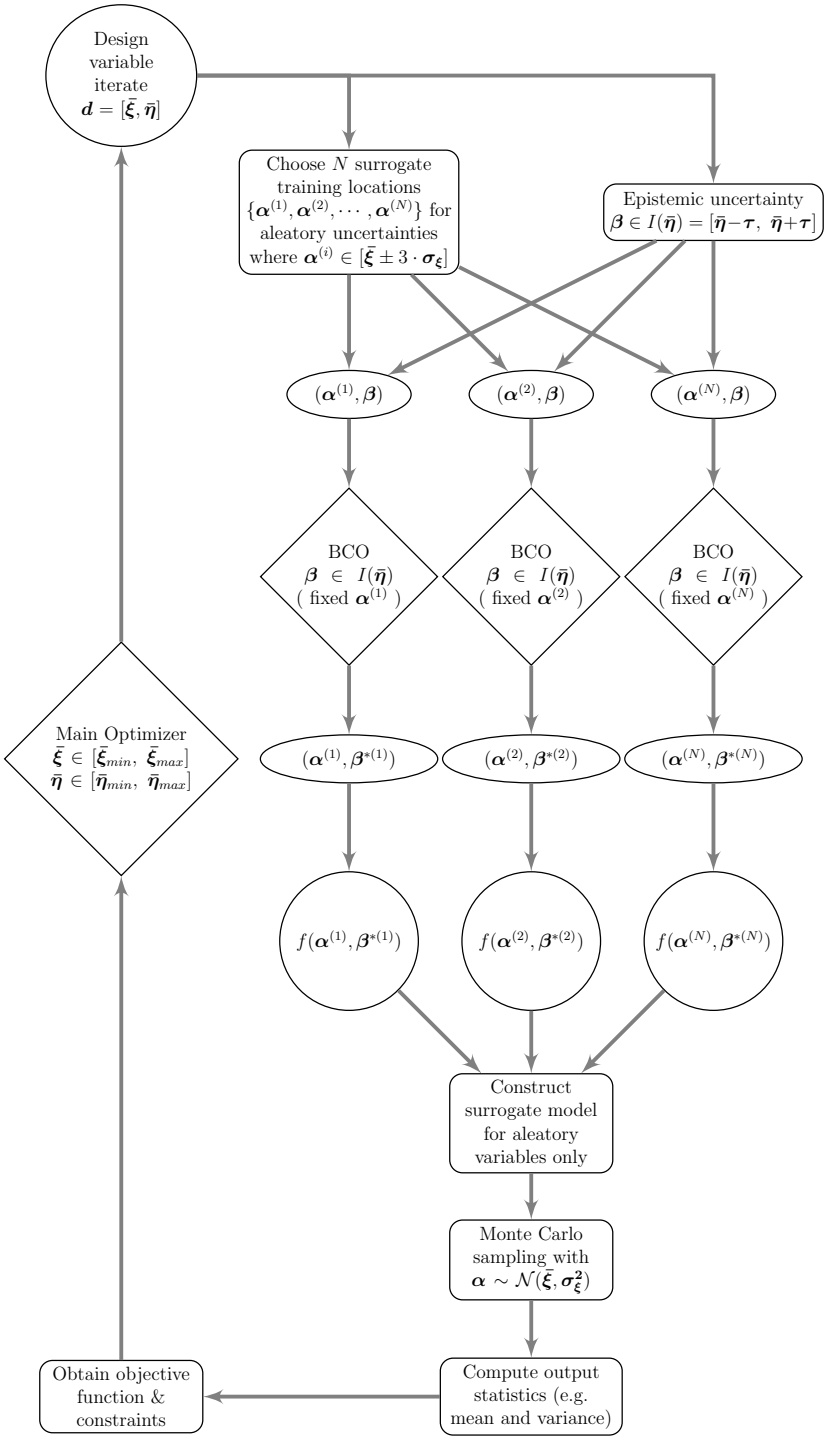


Figure 8.5: Framework for robust optimization under mixed epistemic and aleatory uncertainties.

CHAPTER IX

ROBUST OPTIMIZATION RESULTS

This chapter presents the results pertaining to the robust optimization of structural and aerodynamic designs.

9.1 Three-bar Truss Design

In this section the robust optimization of a three-bar truss is discussed.

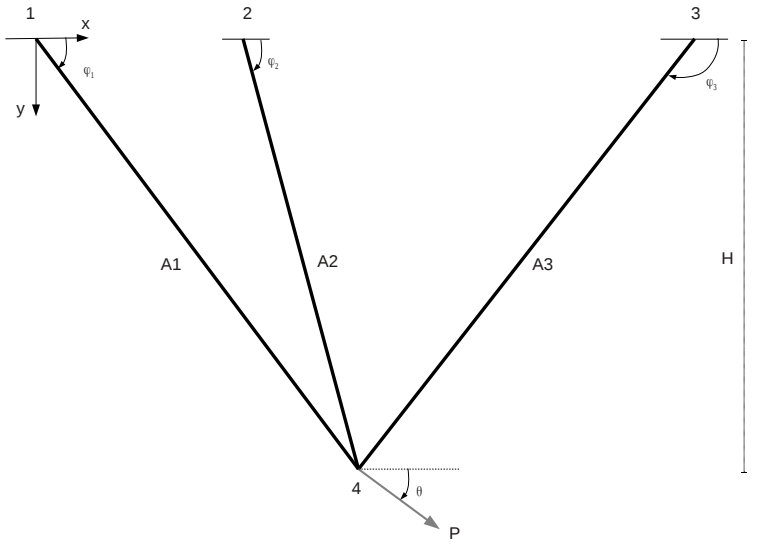


Figure 9.1: A schematic diagram of the three-bar truss structure.

9.1.1 Deterministic Problem

The truss shown in Figure 9.1 is subjected to a load inclined at an angle θ from the horizontal, which puts bars 1 and 2 under tension and bar 3 under compression. The nodes are represented with numbers 1 through 4. The goal is to minimize the total weight W of the structure. The mathematical formulation of the problem is given below in Eq (9.1).

$$
\begin{aligned}
\underset{\boldsymbol{d}}{\text{minimize}} \quad & W = \frac{A_1\gamma H}{\sin(\phi_1)} + \frac{A_2\gamma H}{\sin(\phi_2)} + \frac{A_3\gamma H}{\sin(\phi_3)}, \\
\text{subject to} \quad & g_1 = \frac{\sigma_1}{\sigma_{1_{max}}} - 1 \leq 0, \\
& g_2 = \frac{\sigma_2}{\sigma_{2_{max}}} - 1 \leq 0, \\
& g_3 = \frac{\sigma_3}{\sigma_{3_{max}}} - 1 \leq 0, \\
& g_4 = -\frac{\sigma_1}{\sigma_{1_{max}}} - 1 \leq 0, \\
& g_5 = -\frac{\sigma_2}{\sigma_{2_{max}}} - 1 \leq 0, \\
& g_6 = -\frac{\sigma_3}{\sigma_{3_{max}}} - 1 \leq 0, \\
& g_7 = \frac{Q_{4x}}{Q_{4x_{max}}} - 1 \leq 0, \\
& g_8 = \frac{Q_{4y}}{Q_{4y_{max}}} - 1 \leq 0, \\
\text{bounds} \quad & 0.25\ in^2 \leq\ A_1,\ A_2,\ A_3\ \leq 5.0\ in^2, \\
& 30^\circ \leq\ \phi_1\ \leq 60^\circ, \\
& 60^\circ \leq\ \phi_2\ \leq 120^\circ, \\
& 120^\circ \leq\ \phi_3\ \leq 150^\circ.
\end{aligned}
\tag{9.1}
$$

The problem has a total of six design variables $\boldsymbol{d} = [A_1, A_2, A_3, \phi_1, \phi_2, \phi_3]$, *i.e.,* the areas ($A_1$, A_2 and A_3) and the orientations (ϕ_1, ϕ_2 and ϕ_3) of the bars with respect to the horizontal. The structure has to be designed to withstand a total of 8 constraints $g_i(\boldsymbol{d})$. It is to be noted that the constraints are normalized with respect to their allowable values (denoted with subscript max). The first three, the next three, and the last two constraints

impose tensile stress, compressive stress and displacement requirements, respectively. The axial stresses and nodal displacements used in Eq. (9.1) are calculated using a finite element procedure described in Appendix A. The other parameters used for this problem are listed in Table 9.1.

Table 9.1: Design data for three-bar truss.

Quantity	Description	Value	Unit
P	Load	30000	lb
θ	Loading angle	50	deg
E	Young's modulus	10^7	psi
γ	Weight density	0.1	lb/in^3
H	Reference length (projection on $y-$axis)	10	in
$\sigma_{1_{max}}$	Allowable axial stress on bar 1	5000	psi
$\sigma_{2_{max}}$	Allowable axial stress on bar 2	10000	psi
$\sigma_{3_{max}}$	Allowable axial stress on bar 3	5000	psi
$u_{4x_{max}}$	Allowable x-displacement at 4	0.005	in
$u_{4y_{max}}$	Allowable y-displacement at 4	0.005	in
ϵ_1	Constraint violation tolerance	10^{-3}	-
ϵ_2	Norm of design change $\|\Delta \boldsymbol{d}\|$	10^{-3}	-

9.1.2 Robust Optimization Problem

The robust optimization problem involves minimizing the following objective function:

$$\begin{aligned} &\underset{\boldsymbol{\xi},\boldsymbol{\eta}}{\text{minimize}} \quad \mathcal{J} = \mu_W + \sigma_W^2, \\ &\text{subject to} \quad g_i^r = \mu_{g_i} + k\sigma_{g_i} \leq 0, \quad \text{for } i = 1, \ldots, 8, \end{aligned} \tag{9.2}$$

i.e. the minimization of an equally weighted sum of the mean and variance of the weight subject to eight constraints. The area design variables A_i are assumed to have epistemic uncertainties with bounds $\tau_i = \pm 0.1\, in^2$ and the orientations ϕ_i are assumed to have aleatory uncertainties with standard deviation $\sigma_i = 1^\circ$. The input aleatory uncertainties are modeled

as $\alpha^{(j)} \sim \mathcal{N}(\mu_{\phi_i}, \sigma^2_{\phi_i})$ and the epistemic uncertainties are represented as an interval $\beta^{(j)} \in [A_i - \tau_i, A_i + \tau_i]$. All other input parameters are kept fixed throughout the optimization.

Surrogate Models: The kriging surrogate model is built with seventy training points. The polynomial chaos surrogate is a fourth order polynomial with an oversampling factor of two which also requires seventy training points. The training points are chosen via the dynamic training point selection framework [107, 108]. Note that each training data f^* comes from solving a BCO problem as discussed in section 8.2.2.

9.1.3 Optimization Results

9.1.3.1 Deterministic and Robust Designs

Table 9.2 compares the robust design optima with the deterministic optimum. From the optimum weights, it can be inferred that the deterministic design is the best in terms of lightness of the structure, but lacks robustness. A deterministic design with no assumed factor of safety is 15% lighter than a highly robust design specified by $k = 4$. However, a deterministic design with a small factor of safety of 1.3 is 29% heavier than a highly robust design specified by $k = 4$. The designers can capitalize such a behavior for over-conservative designs that are in use today or the ones that need to be built in the future. A design corresponding to $k = 0$ with a weight of 14.65 ± 0.24 has 50% chances of violating the constraints and is not so robust compared to a design corresponding to $k = 3$ with a weight of 16.54 ± 0.25 that has less than one percent probability of violating the constraints. As the desired robustness specified with k increases, an increase in the objective function value can be seen, meaning that robustness is obtained at the expense of additional weight to the structure. The designer can carry out a trade-off study between the weight of the

structure and the required robustness specified with k or P_k. It can be seen that the kriging and polynomial chaos based results agree very closely for all the tested cases.

Table 9.2: Optimization results for three-bar truss problem.

Type	k	P_k	A_1 in^2	A_2 in^2	A_3 in^2	ϕ_1 deg	ϕ_2 deg	ϕ_3 deg	μ_W lb	σ_W lb	C_v -	No. of F/FG Evals. & Iterations
Initial design	-	-	2.0	2.0	2.0	45.0	90.0	135.0	7.66	-	-	-
Det $F_s = 1.0$	-	-	5.00	1.42	2.30	37.6	60.0	150.0	14.45	-	-	108/108-12
Det $F_s = 1.3$	-	-	5.00	4.95	5.00	39.5	60.0	143.6	22.00	-	-	126/126-14
Robust-KR	0	0.5000	5.00	1.45	2.37	37.7	60.0	150.0	14.65	0.24	0.0162	17559/17559-12
Robust-PC	0	0.5000	5.00	1.45	2.37	37.7	60.0	150.0	14.65	0.24	0.0162	17615/17615-12
Robust-KR	1	0.8413	5.00	1.66	2.66	37.5	60.0	149.3	15.41	0.24	0.0159	21963/21963-14
Robust-PC	1	0.8413	5.00	1.66	2.66	37.5	60.0	149.3	15.41	0.24	0.0159	20555/20555-13
Robust-KR	2	0.9772	5.00	1.84	2.92	37.5	60.0	148.6	16.02	0.25	0.0155	23594/23594-13
Robust-PC	2	0.9772	5.00	1.84	2.92	37.5	60.0	148.6	16.02	0.25	0.0155	33555/33555-18
Robust-KR	3	0.9986	5.00	1.99	3.15	37.5	60.0	148.2	16.54	0.25	0.0153	20771/20771-12
Robust-PC	3	0.9986	5.00	1.99	3.15	37.5	60.0	148.2	16.54	0.25	0.0153	17938/17938-12
Robust-KR	4	0.9999	5.00	2.13	3.36	37.6	60.0	147.9	17.00	0.26	0.0151	31178/31178-17
Robust-PC	4	0.9999	5.00	2.13	3.36	37.6	60.0	147.9	17.00	0.26	0.0151	19500/19500-12

It can also be noted that area A_1 is pushed to its upper limit for all designs, while the other two areas (A_2 and A_3) and orientations generally govern the robustness of the structure.

Since the standard deviation is always associated with a mean it is advantageous to use a dimensionless number, the coefficient of variation C_v (also known as relative standard deviation or relative amount of uncertainty) [11] that measures the extent of variability in relation to the mean of the output. It can be used as a metric of comparison with different data sets or designs that involve different units, or different assumed input mean and standard deviations. A decrease in the coefficient of variation can be observed across the robust designs.

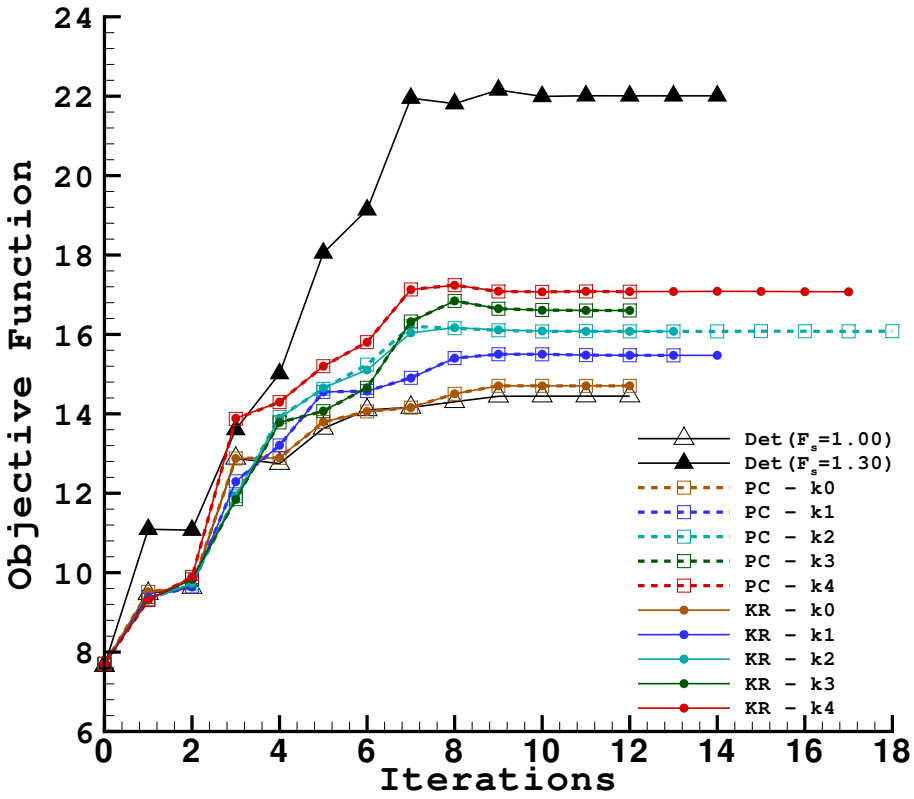


Figure 9.2: Change in the objective function with the number of optimizer iterations.

9.1.3.2 Simulation Requirements

Table 9.2 also presents the number of exact function and gradient evaluations needed to reach the final design. Here, each constraint evaluation is counted towards the total number of simulations. The box-constrained optimization takes $2-3$ exact function and gradient evaluations for this test case. On average, the kriging and polynomial chaos took roughly the same number of function and gradient evaluations to reach the optimum. Figure 9.2 plots the change in the objective function with the number of optimizer iterations.

Table 9.3: Constraint status for three-bar truss problem.

Type	k	g_1^r	g_2^r	g_3^r	g_4^r	g_5^r	g_6^r	g_7^r	g_8^r
Initial design	-	$0.12 \cdot 10^{1}$	$-0.43 \cdot 10^{0}$	$-0.21 \cdot 10^{1}$	$-0.32 \cdot 10^{1}$	$-0.16 \cdot 10^{1}$	$0.69 \cdot 10^{-1}$	$0.14 \cdot 10^{0}$	$0.23 \cdot 10^{1}$
Det $F_s = 1.0$	-	$-0.14 \cdot 10^{0}$	$-0.41 \cdot 10^{0}$	$-0.12 \cdot 10^{1}$	$-0.19 \cdot 10^{1}$	$-0.16 \cdot 10^{1}$	$0.82 \cdot 10^{0}$	$-0.48 \cdot 10^{-8}$	$-0.17 \cdot 10^{-8}$
Det $F_s = 1.3$	-	$-0.29 \cdot 10^{0}$	$-0.58 \cdot 10^{0}$	$-0.13 \cdot 10^{1}$	$-0.17 \cdot 10^{1}$	$-0.14 \cdot 10^{1}$	$0.71 \cdot 10^{0}$	$-0.46 \cdot 10^{0}$	$-0.34 \cdot 10^{-7}$
Robust-KR	0	$-0.15 \cdot 10^{0}$	$-0.41 \cdot 10^{0}$	$-0.12 \cdot 10^{1}$	$-0.18 \cdot 10^{1}$	$-0.16 \cdot 10^{1}$	$-0.82 \cdot 10^{0}$	$-0.56 \cdot 10^{-6}$	$-0.87 \cdot 10^{-5}$
Robust-PC	0	$-0.15 \cdot 10^{0}$	$-0.41 \cdot 10^{0}$	$-0.12 \cdot 10^{1}$	$-0.18 \cdot 10^{1}$	$-0.16 \cdot 10^{1}$	$-0.82 \cdot 10^{0}$	$-0.84 \cdot 10^{-5}$	$-0.78 \cdot 10^{-5}$
Robust-KR	1	$-0.17 \cdot 10^{0}$	$-0.42 \cdot 10^{0}$	$-0.11 \cdot 10^{1}$	$-0.18 \cdot 10^{1}$	$-0.15 \cdot 10^{1}$	$-0.79 \cdot 10^{0}$	$0.90 \cdot 10^{-4}$	$0.49 \cdot 10^{-4}$
Robust-PC	1	$-0.17 \cdot 10^{0}$	$-0.42 \cdot 10^{0}$	$-0.11 \cdot 10^{1}$	$-0.18 \cdot 10^{1}$	$-0.15 \cdot 10^{1}$	$-0.79 \cdot 10^{0}$	$-0.12 \cdot 10^{-4}$	$0.38 \cdot 10^{-4}$
Robust-KR	2	$-0.19 \cdot 10^{0}$	$-0.43 \cdot 10^{0}$	$-0.11 \cdot 10^{1}$	$-0.18 \cdot 10^{1}$	$-0.15 \cdot 10^{1}$	$0.77 \cdot 10^{0}$	$-0.73 \cdot 10^{-4}$	$0.13 \cdot 10^{-3}$
Robust-PC	2	$-0.19 \cdot 10^{0}$	$-0.43 \cdot 10^{0}$	$-0.11 \cdot 10^{1}$	$-0.18 \cdot 10^{1}$	$-0.15 \cdot 10^{1}$	$0.77 \cdot 10^{0}$	$-0.16 \cdot 10^{-4}$	$-0.12 \cdot 10^{-3}$
Robust-KR	3	$-0.20 \cdot 10^{0}$	$-0.43 \cdot 10^{0}$	$-0.11 \cdot 10^{1}$	$-0.17 \cdot 10^{1}$	$-0.15 \cdot 10^{1}$	$0.76 \cdot 10^{0}$	$0.28 \cdot 10^{-3}$	$-0.12 \cdot 10^{-3}$
Robust-PC	3	$-0.20 \cdot 10^{0}$	$-0.43 \cdot 10^{0}$	$-0.11 \cdot 10^{1}$	$-0.17 \cdot 10^{1}$	$-0.15 \cdot 10^{1}$	$0.76 \cdot 10^{0}$	$0.16 \cdot 10^{-3}$	$-0.76 \cdot 10^{-4}$
Robust-KR	4	$-0.21 \cdot 10^{0}$	$-0.43 \cdot 10^{0}$	$-0.11 \cdot 10^{1}$	$-0.17 \cdot 10^{1}$	$-0.14 \cdot 10^{1}$	$-0.74 \cdot 10^{-3}$	$0.90 \cdot 10^{-4}$	$0.20 \cdot 10^{-3}$
Robust-PC	4	$-0.21 \cdot 10^{0}$	$-0.43 \cdot 10^{0}$	$-0.11 \cdot 10^{1}$	$-0.17 \cdot 10^{1}$	$-0.14 \cdot 10^{1}$	$-0.74 \cdot 10^{-3}$	$0.75 \cdot 10^{-3}$	$-0.36 \cdot 10^{-3}$

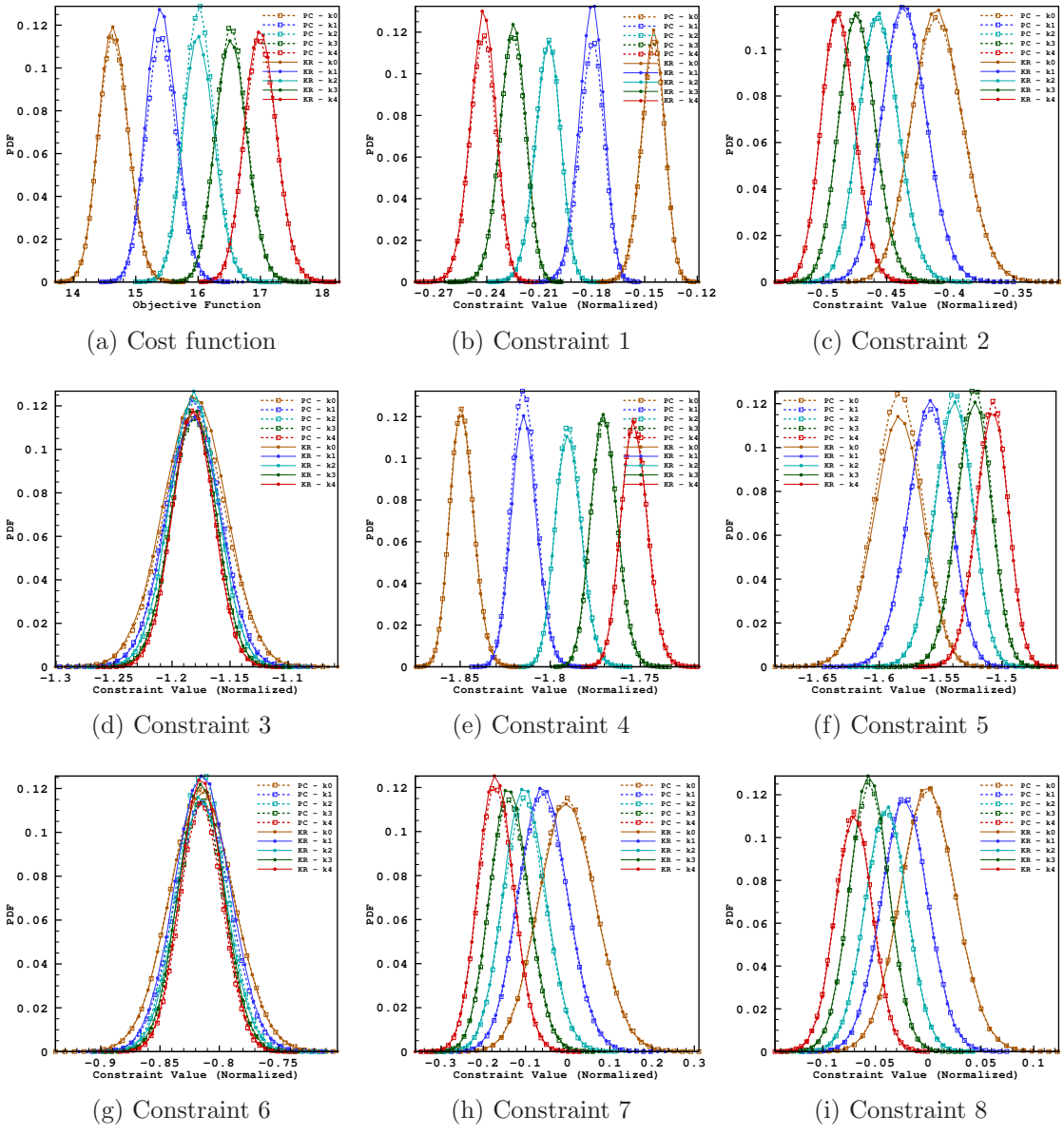
9.1.3.3 Constraint Status

Table 9.3 displays the status of all eight constraints at the initial design, deterministic optimum, and robust optimum. Here, a positive value for g represents a constraint-violation, whereas a negative value means that the constraint is satisfied. It can be inferred that constraints 1, 7 and 8 are the ones that significantly affect the design throughout the optimization (tight or ϵ-active constraints). The kriging and polynomial chaos based values are a little different for these tight constraints, yet within the specified tolerance to ensure that the constraints are not violated. All other constraints are inactive and both the surrogate models provide same values to these constraints.

9.1.3.4 Output PDF and CDF

Figures 9.3 and 9.4 show the probability density function (PDF) and cumulative distribution function (CDF) of the objective function (weight) as well as the constraints (normalized) at their optimum designs. As the desired robustness specified with k increases, an increase in the objective function value by means of a shift to the right can be seen. The robust optimization problem formulation serves to move the constraint values k standard

deviations away from a potential violation which is evident from the PDF and CDF of the constraints. It can be seen that a design corresponding to $k = 4$ has less than 1% chances of constraint violation, whereas $k = 0$ features 50% chances of constraint violation due to the effect of input uncertainties.



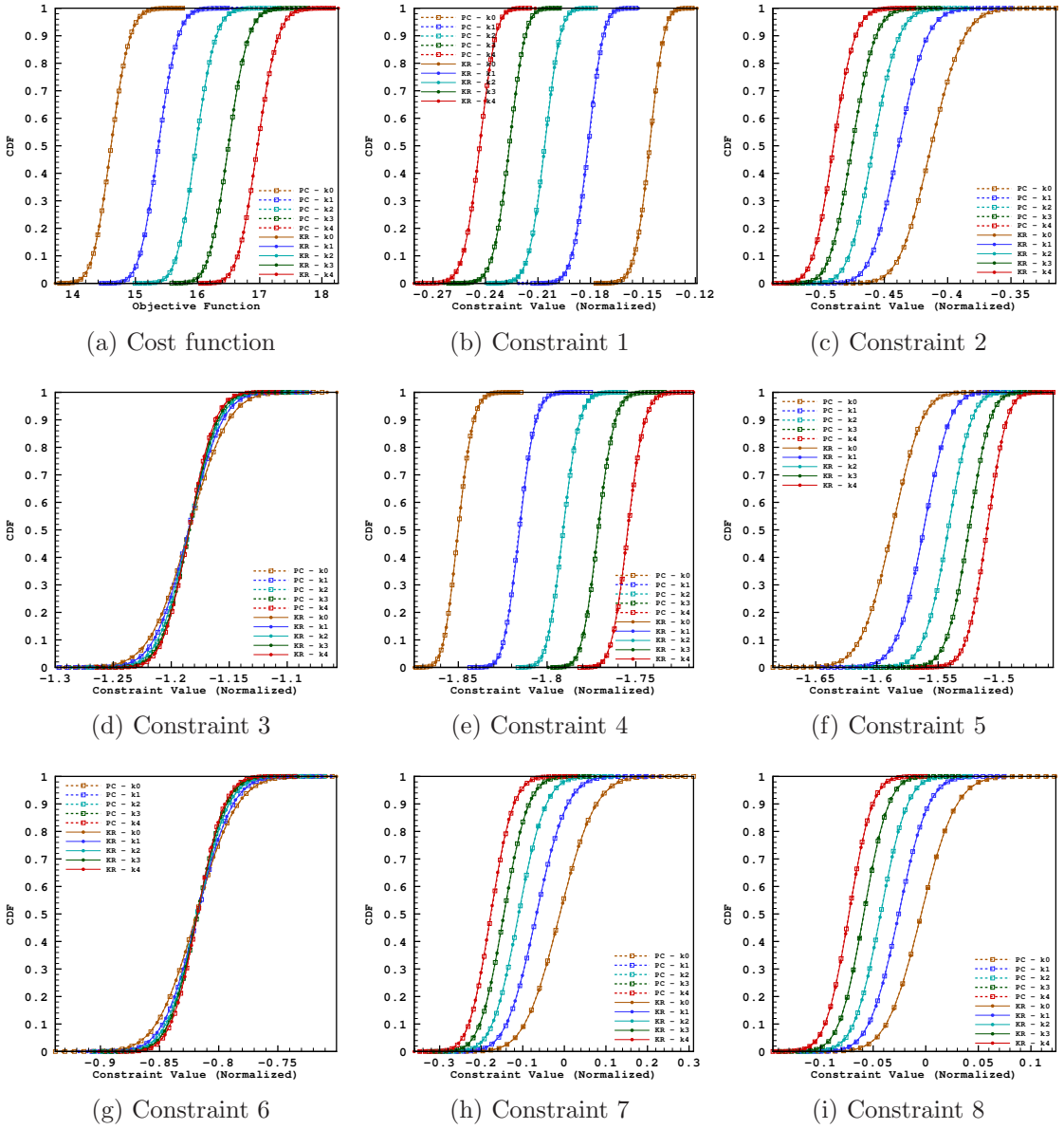
(a) Cost function (b) Constraint 1 (c) Constraint 2

(d) Constraint 3 (e) Constraint 4 (f) Constraint 5

(g) Constraint 6 (h) Constraint 7 (i) Constraint 8

Figure 9.3: Probability density function of objective and constraint functions at robust optimum designs.

It can also be seen that the spread of values is reduced as the robustness increases (compare PDFs of $k = 0$ and $k = 4$ cases), which shows that the design is less sensitive to uncertainties/variations in the input. Overall both surrogate models produce comparable distributions apart from occasional differences.



(a) Cost function (b) Constraint 1 (c) Constraint 2

(d) Constraint 3 (e) Constraint 4 (f) Constraint 5

(g) Constraint 6 (h) Constraint 7 (i) Constraint 8

Figure 9.4: Cumulative distribution function of objective and constraint functions at robust optimum designs.

9.2 Cantilever Beam Design

This section describes the robust optimization of a cantilever beam.

9.2.1 Problem Description

A cantilever beam of rectangular cross-section is subjected to a bending moment $\mathcal{M}$ ($N \cdot mm$) and shear force $\mathcal{V}$ (N). The bending stress in the beam is calculated as $\sigma = 6\mathcal{M}/bd^2$ (N/mm^2) and the average shear stress is calculated as $\tau = 3\mathcal{V}/2bd$ (N/mm^2), where b is the width and d is the depth of the beam. The maximum allowable stress in the form of bending, σ_{allow}, is 10 N/mm^2 and the maximum allowable shear, τ_{allow}, is 2 N/mm^2. The goal is to minimize the cross-sectional area A (mm^2) of the beam. The mathematical formulation of the problem is given below:

$$\begin{aligned} \underset{b,d}{\text{minimize}} \quad & A(b,d) = bd, \\ \text{subject to} \quad & g_1(b,d,\mathcal{M}) = \frac{6\mathcal{M}}{bd^2\sigma_{allow}} - 1 \leq 0, \\ & g_2(b,d,\mathcal{V}) = \frac{3\mathcal{V}}{2bd\tau_{allow}} - 1 \leq 0, \\ & g_3(b,d) = \frac{d}{2b} - 1 \leq 0, \\ \text{bounds} \quad & 100\ mm \ \leq \ b,d \ \leq \ 600\ mm, \end{aligned} \tag{9.3}$$

where the constraints g_1 and g_2 enforce bending and shear stress requirements, respectively, while g_3 imposes an aspect-ratio requirement for the rectangular cross-section. All the constraints are represented in standard normalized form. The design variables are the width and depth of the beam.

9.2.2 Robust Optimization Problem

The two allowable stresses (σ_{allow} and τ_{allow}) in Eq. (9.3) are assumed to be precise (hence kept fixed) whereas the remaining parameters are assumed to have uncertainties and are therefor treated as random variables as listed in Table 9.4. The bending moment

Table 9.4: Data and assumed uncertain parameters for cantilever beam design problem.

Random Variable	Description	Uncertainty Type	τ_{min}	τ_{max}	Mean	Standard Deviation	Unit
b	Breadth	Epistemic	-10	10	-	-	mm
d	Width	Epistemic	-10	10	-	-	mm
$\mathcal{M}$	Bending Moment	Aleatory	-	-	$40 \cdot 10^6$	40000	$N \cdot mm$
$\mathcal{V}$	Shear Force	Aleatory	-	-	$150 \cdot 10^3$	1500	N

and shear force are assumed to have normally distributed aleatory uncertainties with specified mean and standard deviations as shown in Table 9.4. Only the constraints g_1 and g_2 which are functions of the bending moment and shear force are influenced by these aleatory variables. Unlike the three-bar truss problem where all random variables were also considered as design variables, the cantilever beam problem involves random variables which are not design variables, but their effects will be considered in the optimization procedure. The robust optimization problem can be written as:

$$\begin{aligned} \underset{b,d}{\text{minimize}} \quad & A(b,d) = \mu_A + \sigma_A^2, \\ \text{subject to} \quad & g_1^r(b,d,\mathcal{M}) = \mu_{g_1} + k\sigma_{g_1} \leq 0, \\ & g_2^r(b,d,\mathcal{V}) = \mu_{g_2} + k\sigma_{g_2} \leq 0, \\ & g_3^r(b,d) = \mu_{g_3} + k\sigma_{g_3} \leq 0. \end{aligned} \tag{9.4}$$

In this problem only the epistemic random variables (width and depth) govern the cost function and the aspect-ratio constraint and therefore the output standard deviations are unavailable for these functions: σ_A and $\sigma_{g_3} = 0$.

Surrogate Models: The robust optimization results will be compared using both kriging and polynomial chaos. The kriging surrogate model is built with 20 training points and the polynomial chaos metamodel is a third order polynomial which is also built with 20 training points.

9.2.3 Optimization Results

Table 9.5 presents the optimization results. It can be seen that the objective function value increases with the desired robustness, for example, the cross-sectional area increases by roughly 17% for a design corresponding to $k = 4$ compared to a deterministic design with no factor of safety. However, the robust design ($k = 4$) has 29% less cross-sectional area (hence lighter) than a deterministic design with a factor of safety of 1.5.

Table 9.5: Optimization results for cantilever beam design problem.

Type	k	P_k	Width b mm	Depth d mm	Area A $\cdot 10^3$ mm^2	No. of F/FG Evals. & Iterations
Initial Design	-	-	300	300	90.0	-
Det ($F_s = 1.0$)	-	-	335.5	335.4	112.5	33/33-7
Det ($F_s = 1.5$)	-	-	595.5	283.4	168.7	45/45-8
Robust-KR	0	0.5000	347.4	343.4	126.3	7046/3523-7
Robust-PC	0	0.5000	347.4	343.4	126.3	7917/7917-8
Robust-KR	1	0.8413	349.7	344.5	127.5	7146/3573-7
Robust-PC	1	0.8413	349.7	344.5	127.5	8037/8037-8
Robust-KR	2	0.9772	398.5	305.4	128.8	7686/3843-7
Robust-PC	2	0.9772	398.5	305.4	128.8	9661/9661-9
Robust-KR	3	0.9986	386.5	317.8	130.0	8694/4347-8
Robust-PC	3	0.9986	386.5	317.8	130.0	11669/11669-10
Robust-KR	4	0.9999	356.6	347.5	131.1	7286/3643-7
Robust-PC	4	0.9999	356.6	347.5	131.1	8196/8196-8

Figure 9.5 shows all three constraints plotted along with the objective function contours. The objective function is parallel to the constraint g_2, therefore, any point on the cure A–B is a feasible deterministic optimum. At point A, the constraints g_2 and g_3 are active; at point B, the constraints g_1 and g_2 are active; while any point on the curve A–B has the constraint g_2 active. Through robust optimization, the optimum solution is moved by a distance of k standard deviations away from the deterministic solution, which is shown by

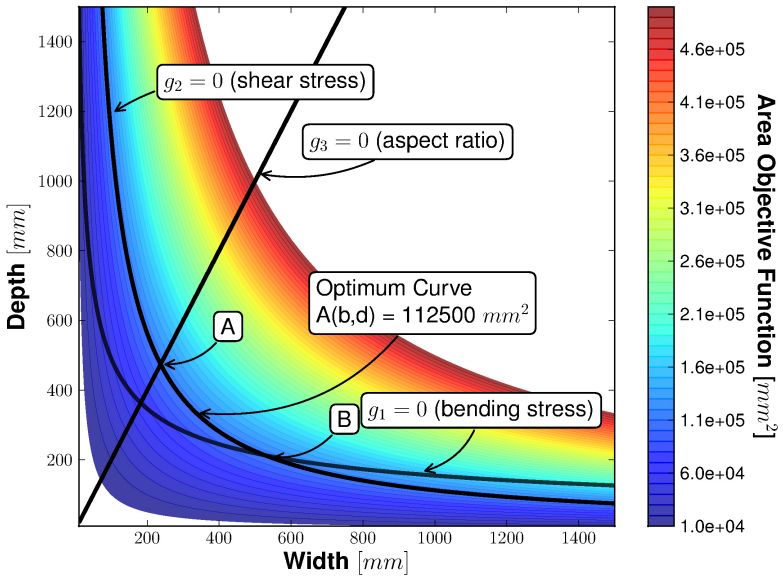


Figure 9.5: Graphical solution to the minimum area beam design problem.

an increment in the objective function values in Table 9.5. However, the robust optimization accounts for the exact amount of uncertainty in the problem and achieves a reduced cost function compared to deterministic designs employing an arbitrary factor of safety which could easily be over-conservative or under-conservative.

9.2.3.1 Simulation Requirements

The box-constrained optimizations took only 2 to 3 function and gradient evaluations to reach the extremum. On average the robust optimization takes about 7500 function and gradient evaluations (includes the constraint evaluations), when compared to the deterministic optimization whose simulation needs are many folds lesser. Simulation requirements for the polynomial chaos method is roughly 20% higher than that of kriging.

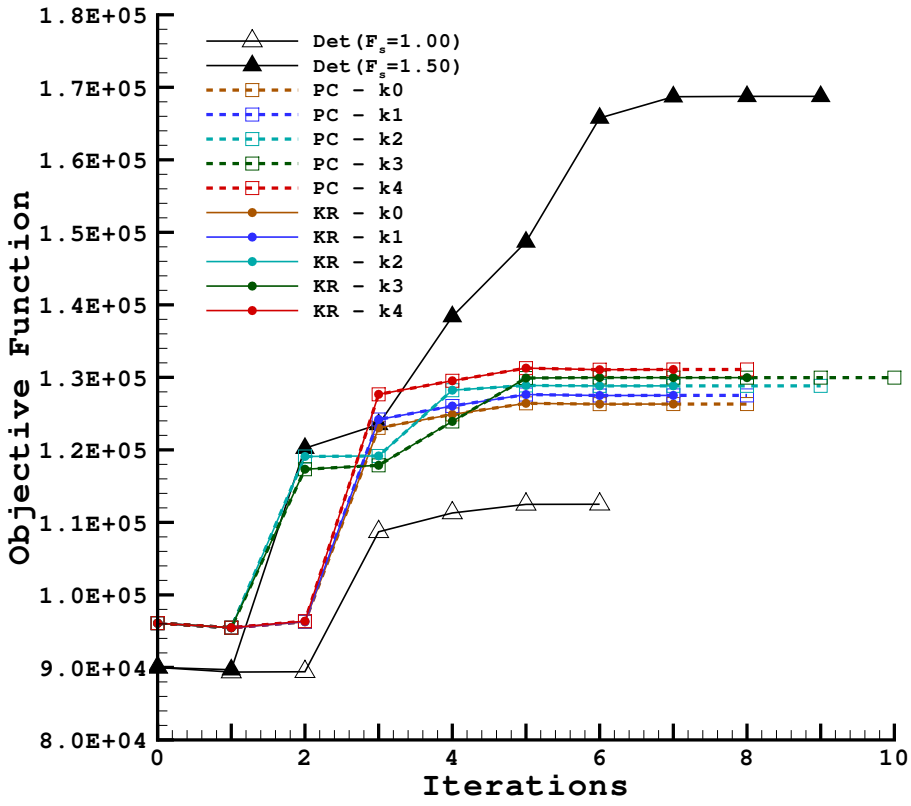


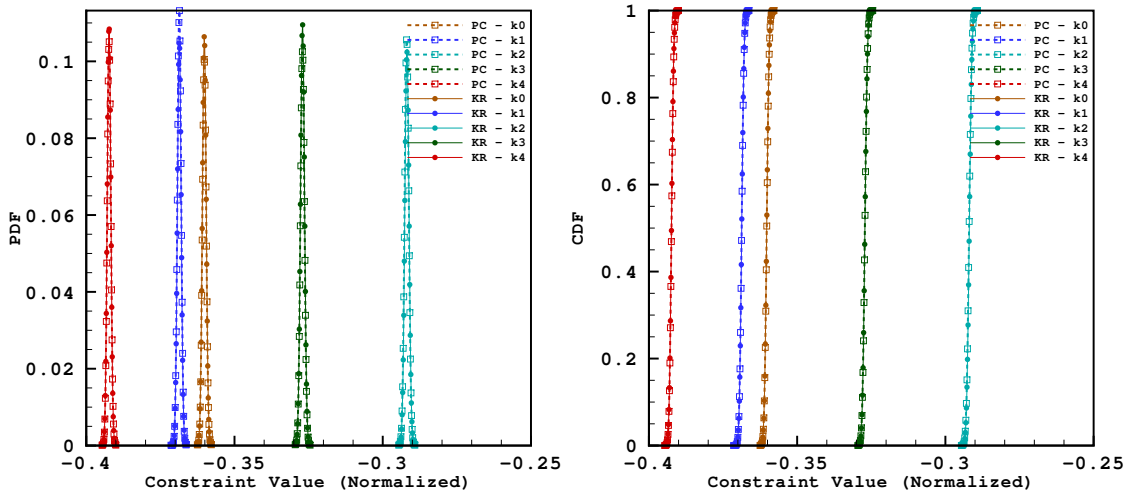
Figure 9.6: Optimization history for the beam design problem.

Figure 9.6 plots the change in the objective function with the number of optimization iterations for different tested cases (deterministic and robust).

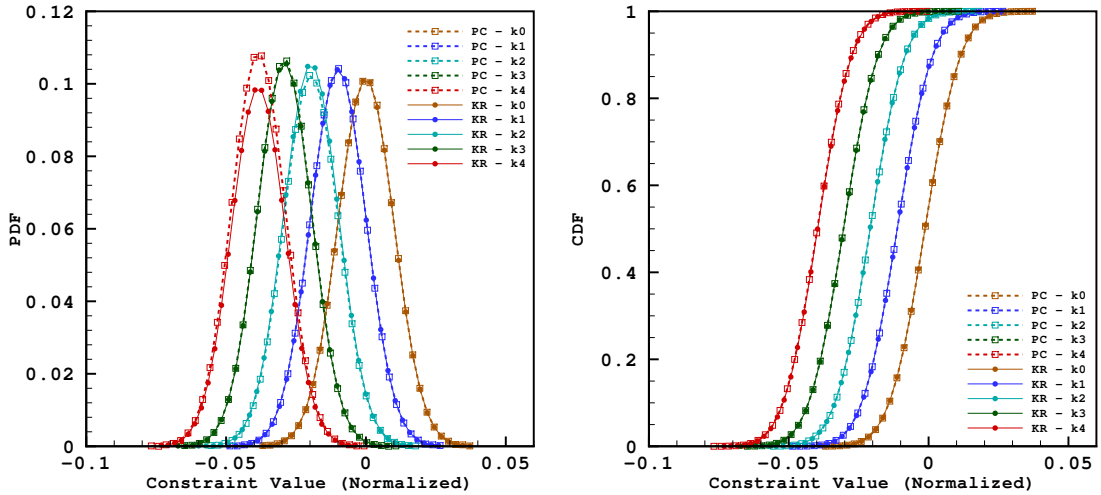
9.2.3.2 Output PDF and CDF

Figure 9.7 shows the probability density and cumulative distribution functions of constraints g_1 and g_2. The PDFs show the spread of possible values taken by the constraints corresponding to different robust designs, whereas the CDFs show the probability of obtaining a specified value or less. Knowing the spread of values helps a designer to make informed decisions about the performance of the design. For example, a robust design corresponding

to $k = 4$ (red lines) features negligible occurrence of $g_i > 0$ (signifies a constraint violation). It can also be observed that the constraint values are normally distributed.



(a) PDF (g_1) (b) CDF (g_1)



(c) PDF (g_2) (d) CDF (g_2)

Figure 9.7: Output PDF (left) and CDF (right) of constraint g_1 and g_2.

9.3 Airfoil Design

In this section the robust optimization of an airfoil is discussed.

9.3.1 Aerodynamic Analysis

The steady inviscid flow around an airfoil governed by the Euler equations is solved by using a second-order accurate finite-volume approach [94, 95]. The computational mesh is shown in Figure 9.8. Hicks-Henne sine bump functions [109] are used to control the shape of the airfoil resulting from perturbations of shape design variables. The resulting deformation of the mesh is calculated via a linear tension spring analogy [36, 110].

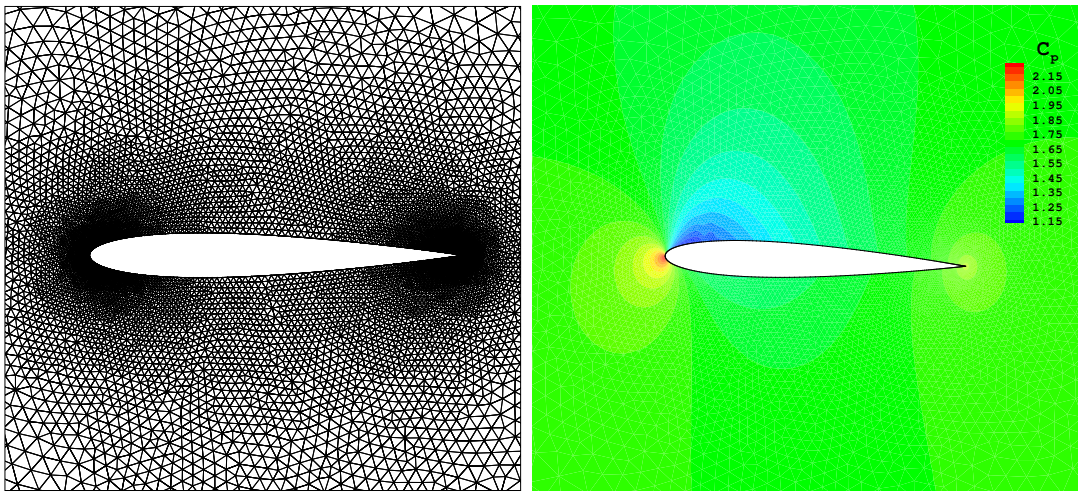


Figure 9.8: Computational mesh for NACA 0012 airfoil with 19, 548 elements (left), pressure distribution at $\alpha = 2.0^\circ$ and M=0.65 (right).

9.3.2 Robust Optimization Problem

Seven shape design variables are placed on the upper surface and seven on the lower surface of the airfoil (at 20%, 30%, 40%, 50%, 60%, 80%, and 90% chord). The bounds on

the flow variables, angle of attack and Mach number, are taken as $0^\circ \leq \alpha \leq 4^\circ$ and $0.6 \leq M \leq 0.78$. All fourteen shape design variables are assumed to have epistemic uncertainties

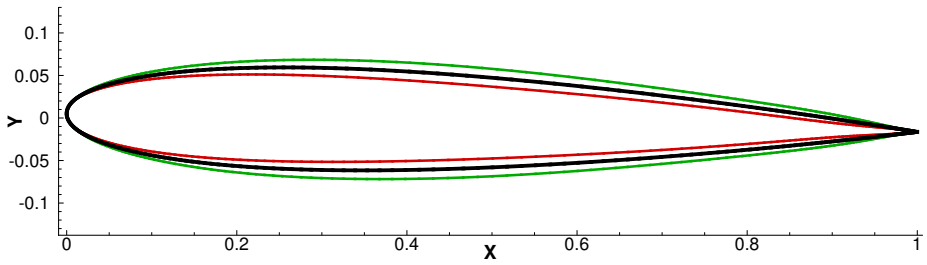


Figure 9.9: The NACA 0012 airfoil (in black) and airfoils resulting from perturbations of ± 0.0025 (in red and blue).

and the two flow variables are assumed to have aleatory uncertainties.

Table 9.6: Data for robust optimization of airfoil.

Random Variable	Description	Uncertainty Type	τ_{min}	τ_{max}	Standard Deviation
$\eta_{1,2,13,14}$	Shape design variables	Epistemic	-0.00125	0.00125	-
η_{3-12}	Shape design variables	Epistemic	-0.01	0.01	-
ξ_α	Angle of attack	Aleatory	-	-	0.1°
ξ_M	Mach number	Aleatory	-	-	0.01

Figure 9.9 shows the baseline NACA 0012 airfoil used as the initial (starting) design and the airfoils resulting from perturbations of the fourteen shape design variables within the bounds specified in Table 9.6. The initial value of the angle of attack is 2° and the Mach number is 0.65. The mathematical formulation of the problem is given as follows,

$$\begin{aligned} \underset{\boldsymbol{\xi},\boldsymbol{\eta}}{\text{minimize}} \quad & \mathcal{J} = \mu_{C_{d_{max}}} + \sigma^2_{C_{d_{max}}}, \\ \text{subject to} \quad & g = (\mu_{C_{l_{min}}} + k\sigma_{C_{l_{min}}}) - {C_l}^+ \geq 0, \end{aligned} \tag{9.5}$$

where C_l^+ refers to a target lift coefficient of 0.6, and $C_{l_{min}}$ and $C_{d_{max}}$ are the least possible lift and highest possible drag within the specified set of epistemic variables, respectively.

Surrogate models: The kriging surrogate model is built with 11 training points. The polynomial chaos surrogate is a second-order polynomial with an over sampling factor of two forming a regression surface which requires 12 training points. The surrogate models are built only over the assumed aleatory variables: the angle of attack and Mach number. The domain of the surrogate model is three standard deviations wide from the mean values $\bar{\boldsymbol{\xi}}$ provided by the main optimizer (IPOPT) at every iteration.

9.3.3 Optimization Results

Table 9.7: Optimization results for airfoil design problem.

Type	k	P_k	$\mu_{c_{d_{max}}}$	$\sigma^2_{c_{d_{max}}}$	$\mu_{c_{l_{min}}}$	$\sigma_{c_{l_{min}}}$	α	M	No. of F/FG Evals. & Iterations
Initial	-	-	$4.72 \cdot 10^{-4}$	-	0.335	-	2.000°	0.650	
Deterministic	-	-	$1.17 \cdot 10^{-3}$	-	0.600	-	2.510°	0.600	49/49-24
Robust-KR	0	0.5000	$2.72 \cdot 10^{-3}$	$2.03 \cdot 10^{-7}$	0.600	$1.84 \cdot 10^{-2}$	2.013°	0.600	844/844-23
Robust-PC	0	0.5000	$2.62 \cdot 10^{-3}$	$5.80 \cdot 10^{-8}$	0.600	$1.82 \cdot 10^{-2}$	2.389°	0.600	675/6751-16
Robust-KR	1	0.8413	$2.93 \cdot 10^{-3}$	$3.07 \cdot 10^{-7}$	0.619	$1.86 \cdot 10^{-2}$	2.065°	0.600	434/434-13
Robust-PC	1	0.8413	$2.73 \cdot 10^{-3}$	$2.50 \cdot 10^{-7}$	0.618	$1.84 \cdot 10^{-2}$	3.058°	0.600	434/434-15
Robust-KR	2	0.9772	$3.10 \cdot 10^{-3}$	$4.46 \cdot 10^{-7}$	0.637	$1.88 \cdot 10^{-2}$	2.179°	0.600	831/831-19
Robust-PC	2	0.9772	$3.20 \cdot 10^{-3}$	$8.58 \cdot 10^{-7}$	0.637	$1.89 \cdot 10^{-2}$	2.193°	0.600	710/710-22
Robust-KR	3	0.9986	$3.28 \cdot 10^{-3}$	$6.23 \cdot 10^{-7}$	0.657	$1.90 \cdot 10^{-2}$	2.301°	0.600	650/650-21
Robust-PC	3	0.9986	$3.25 \cdot 10^{-3}$	$9.83 \cdot 10^{-7}$	0.658	$1.92 \cdot 10^{-2}$	2.352°	0.600	1145/1145-21
Robust-KR	4	0.9999	$3.56 \cdot 10^{-3}$	$9.50 \cdot 10^{-7}$	0.677	$1.93 \cdot 10^{-2}$	2.421°	0.600	620/620-15
Robust-PC	4	0.9999	$3.65 \cdot 10^{-3}$	$1.25 \cdot 10^{-6}$	0.677	$1.93 \cdot 10^{-2}$	2.427°	0.600	2104/2104-36

Table 9.7 compares the robust design optima with the deterministic optimum. The average drag and mean angle of attack increase as the desired probability of achieving the target lift C_l^+ is increased. The optimum solution is sought from the optimizer at a distance of k-standard deviations away from the lift-constraint hyperplane. As a result, the amount

of lift produced is higher as more robustness is expected from the design *i.e.,* the additional lift produced defines the robustness of the design and the design will be less prone to failure (violation of the lift-constraint). On the contrary, during a deterministic optimization an optimum design is sought at the constraint boundary, that can very well violate design requirements when the underlying variables are not representative of the ones considered during optimization. Another inference is that robustness is achieved at the expense of the objective function (drag penalty). Also by observing the optimum aleatory variables on the right, it can be seen that the Mach number remains the same for all designs (at its lower bound), whereas the angle of attack varies.

9.3.3.1 Airfoil shape

Figure 9.10 shows the original, deterministic and robustly optimized ($k = 4$, with polynomial chaos) airfoils. It can be inferred that the deterministically optimized airfoil (shown in blue) looks very thin compared to the robustly optimized airfoil (shown in red).

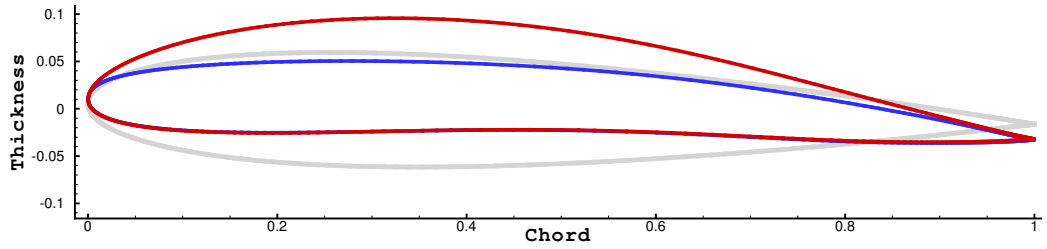
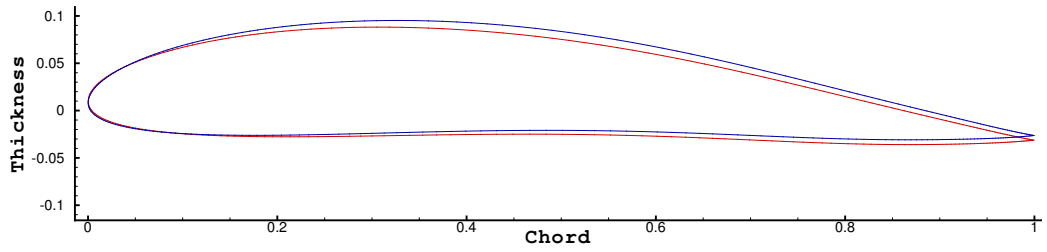
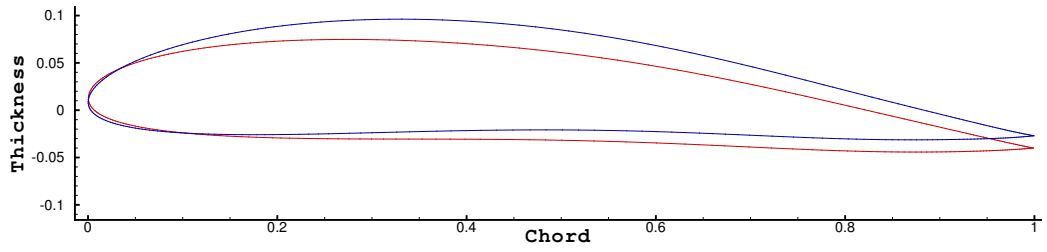


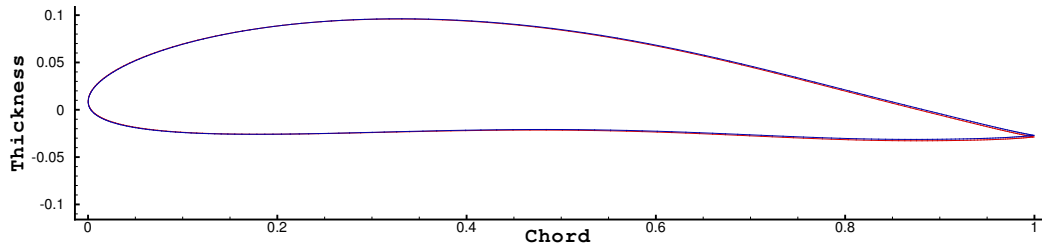
Figure 9.10: Original NACA 0012 (gray), deterministic (blue) and robust with $k = 4$ (red) airfoils produced using polynomial chaos. The kriging produced very similar airfoils (hence not shown).



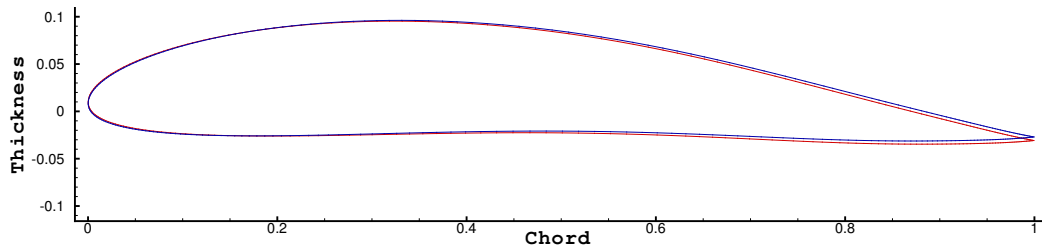
(a) Robust Airfoils $k = 0$



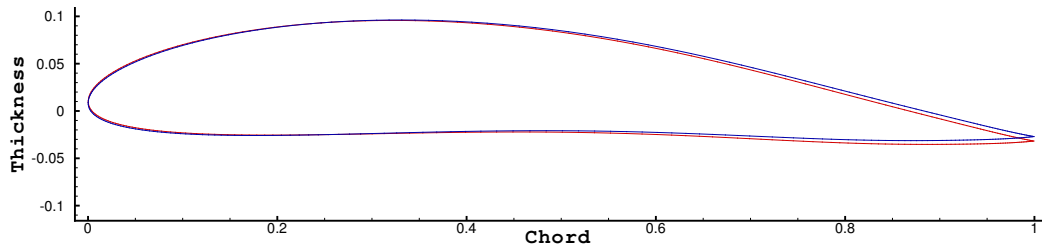
(b) Robust Airfoils $k = 1$



(c) Robust Airfoils $k = 2$



(d) Robust Airfoils $k = 3$



(e) Robust Airfoils $k = 4$

Figure 9.11: Plots showing the shape (also angle of attack) of different robust airfoils. Red and blue lines correspond to polynomial chaos and kriging, respectively.

The robust airfoils corresponding to increasing k are shown in Figure 9.11. Except for the first two cases ($k = 0$ and $k = 1$), the epistemic shape design variables attain similar values for both models. For $k = 1$ case, the kriging based robust airfoil is thicker and has a lower angle of attack, whereas the polynomial chaos based airfoil is thinner and attains the target lift with an increased angle of attack.

Overall, the kriging surrogate based robust designs (shown as blue lines) tend to have a lower angle of attack than its polynomial chaos counterpart (shown as red line). This behavior can be observed across all five robust designs. In general, it may be advantageous for an airplane to fly faster rather than having an increased angle of attack for generating more lift. The solution to the Euler equations ignores important viscous effects, such as boundary layers, wakes and flow separation. If a Navier-Stokes solver is used, it can be expected that the optimizer places a greater emphasis on the shape optimization rather than the flow parameter optimization.

9.3.3.2 Simulation Requirements

The last column of Table 9.7 presents the number of function and gradient evaluations needed as well as the number of iterations taken by the optimizer to converge. Here a single flow solve provides the lift (constraint) and drag (objective) values. For this airfoil optimization test case, the robust optimization needs on average roughly 1000 flow and adjoint solutions, compared to close to 50 evaluations needed for the deterministic optimization, placing a roughly 20 times higher simulation requirement on the designer. The box-constrained optimizations took 2 to 3 flow and adjoint solutions to determine the worst possible lift and the highest possible drag within the specified epistemic uncertainty bounds. For the aleatory uncertainty propagation, though kriging and polynomial chaos involve the same amount of training information (eleven and twelve points respectively), at the end of

the main optimization, the latter's simulation requirements are 50% higher than that of the kriging (on average). This shows that kriging is more effective for non-smooth functions such as this aerodynamic test case.

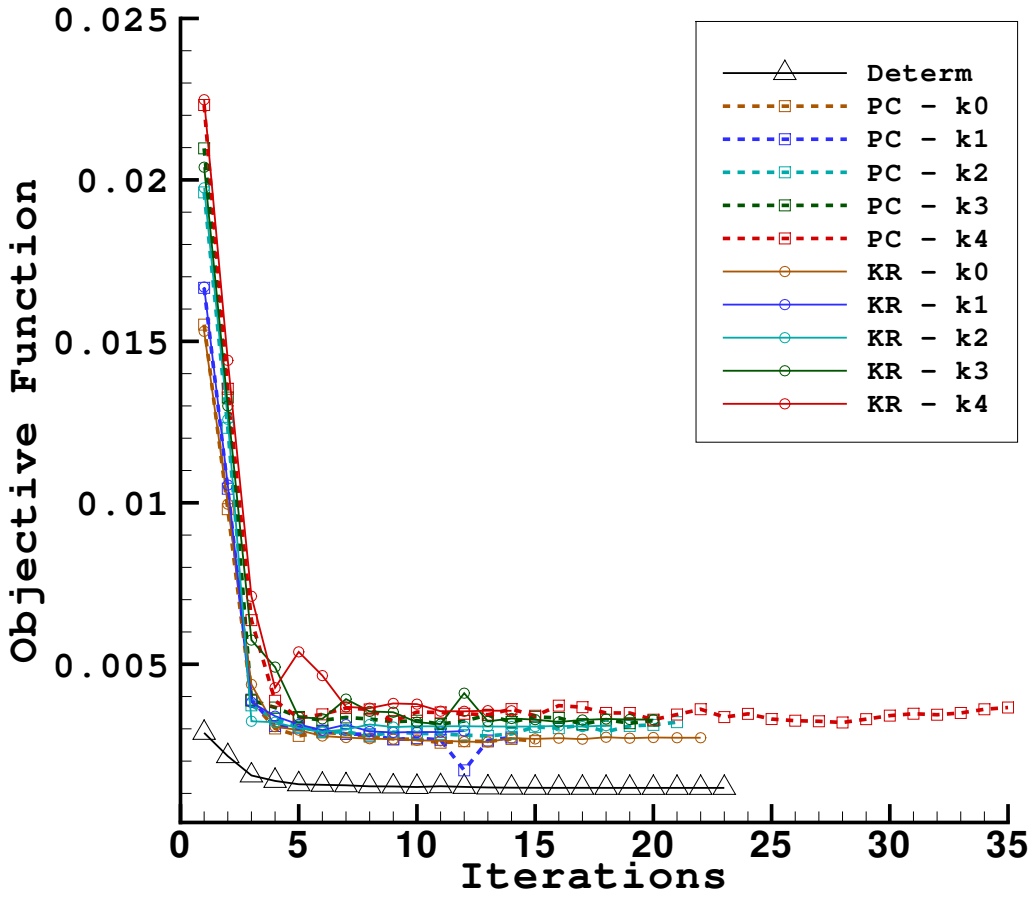


Figure 9.12: Optimizer iteration history for airfoil design problem.

Figure 9.12 plots the change in the objective function with the number of optimizer iterations.

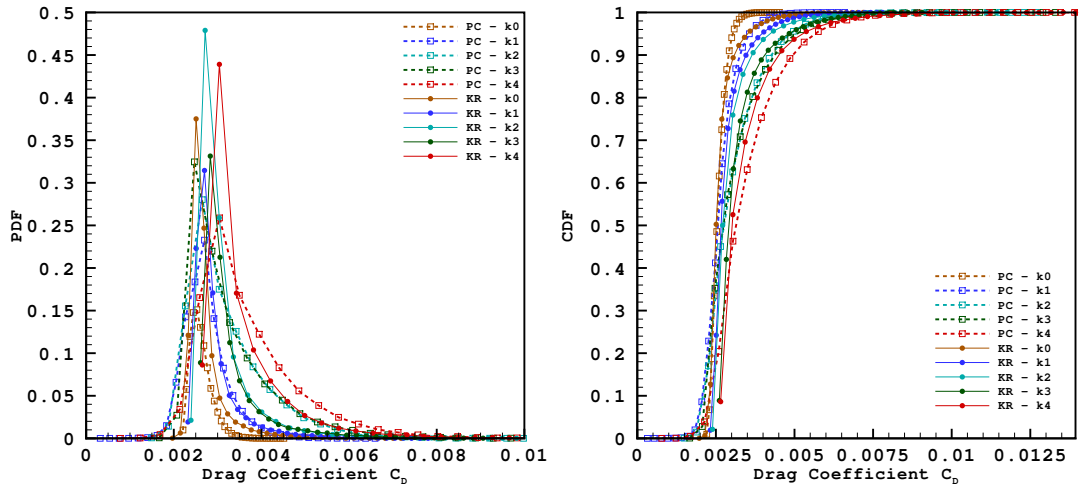


Figure 9.13: PDF and CDF of the drag coefficient at optimum designs.

9.3.3.3 Output PDF and CDF at the Optimum

Figures 9.13 and 9.14 show the PDF as well as CDF of the drag and lift coefficients, respectively, at several robust designs using kriging and polynomial chaos. The PDFs shown in the left shows the distribution of possible drag and lift coefficient values due to the effect of uncertainties, whereas the CDFs show the probability of obtaining a specified value or less. For example, the distribution of drag (see the left of Figure 9.13) helps the designer to construct confidence bounds on possible drag values. Similarly, the probability that the target lift coefficient ($C_L^+ = 0.6$) is not attained is 50% for the $k = 0$ case and is less than 1% for the $k = 4$ case. As the required robustness increases, the distributions shift to the right, which signifies a higher lift generation as well as drag penalty. A Gaussian distribution is seen for the lift coefficient, whereas the distribution of drag coefficient resembles a log-normal one.

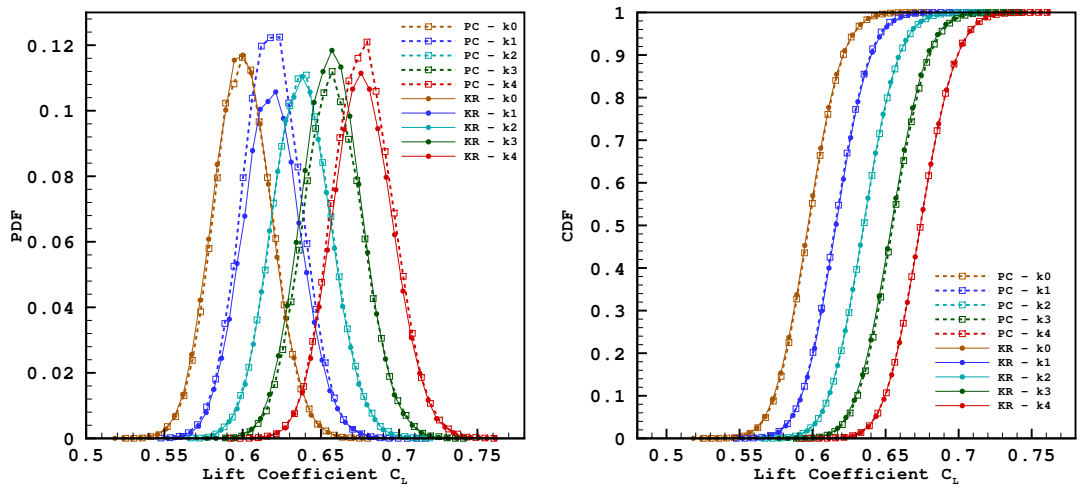


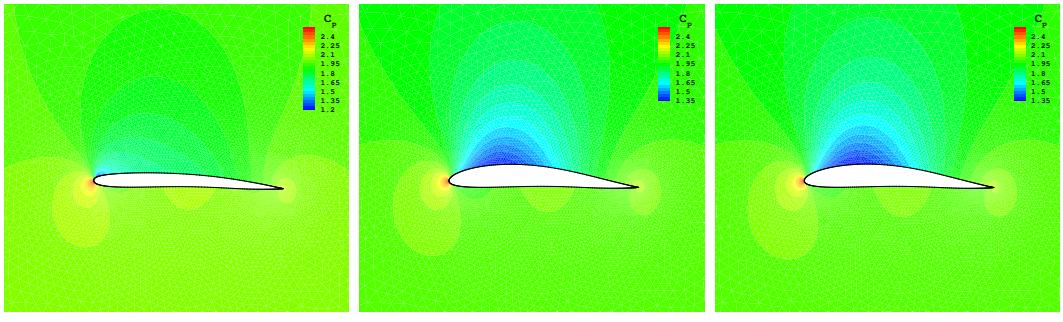
Figure 9.14: PDF and CDF of the lift coefficient at optimum designs.

9.3.3.4 Pressure Contours at the Optimum

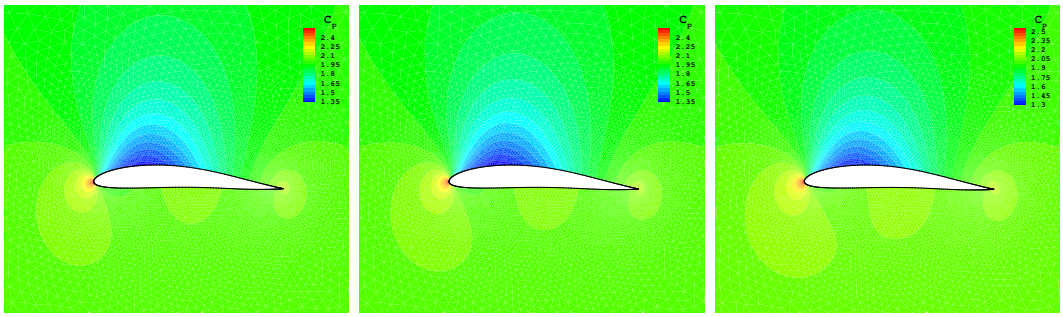
Figures 9.15 and 9.16 show the pressure distribution around deterministically and robustly optimized airfoils using kriging and polynomial chaos, respectively. It can be seen that the pressure distributions are very similar among the robust airfoils, whereas a distinct difference can be noticed between the robust and deterministic ones.

9.3.4 Validation with Exact Monte Carlo Simulation

Here, validations for the IMCS-BCO approach are provided by a selective comparison of the $k = 1$ case with exact Monte Carlo simulation (MCS) and BCO *i.e.,* the surrogate models are replaced with exact function evaluations (Euler flow solutions). Due to the expense of the Euler flow solutions, only 3000 Monte Carlo samples are used for this test. For each Monte Carlo sample, a BCO problem is solved and statistics obtained are presented in Tables 9.8 and 9.9.



(a) Deterministic (b) Robust $k = 0$ (c) Robust $k = 1$



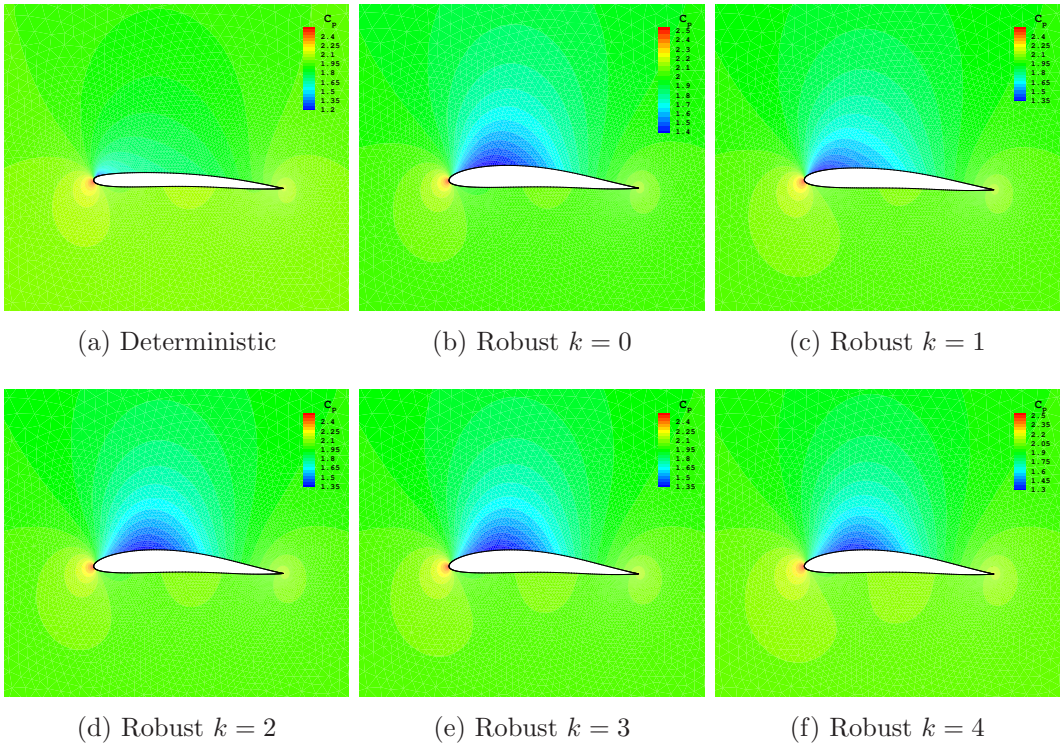
(d) Robust $k = 2$ (e) Robust $k = 3$ (f) Robust $k = 4$

Figure 9.15: Contour plots of pressure coefficients C_p at different optimum designs using kriging.

Table 9.8: Validations for $k = 1$ robust case at the initial design (NACA 0012, $\alpha = 2^\circ$ and $M_\infty = 0.65$) for kriging and polynomial chaos.

Type	$\mu_{c_{d_{max}}}$	$\sigma^2_{c_{d_{max}}}$	$\mu_{c_{l_{min}}}$	$\sigma_{c_{l_{min}}}$	No. of Function/ Gradient Evals.
IMCS-BCO (Kriging)	$8.85 \cdot 10^{-4}$	$4.32 \cdot 10^{-9}$	0.186	$1.72 \cdot 10^{-2}$	38/38
IMCS-BCO (PC)	$9.27 \cdot 10^{-4}$	$3.97 \cdot 10^{-8}$	0.186	$1.72 \cdot 10^{-2}$	36/36
MCS-BCO	$8.98 \cdot 10^{-4}$	$2.98 \cdot 10^{-8}$	0.186	$1.72 \cdot 10^{-2}$	6153/6153

Overall, it can be noticed that the surrogate models produce reasonably accurate statistics for a fraction of the computational cost compared to MCS. Also, kriging is more accurate than polynomial chaos in predicting the statistics.



(a) Deterministic (b) Robust $k = 0$ (c) Robust $k = 1$

(d) Robust $k = 2$ (e) Robust $k = 3$ (f) Robust $k = 4$

Figure 9.16: Contour plots of pressure coefficients C_p at different optimum designs using PCE.

Table 9.9: Validations for $k = 1$ robust case at the final design for kriging (robust shape, $\alpha = 2.065°$ and $M_\infty = 0.6$) and polynomial chaos (robust shape, $\alpha = 3.058°$ and $M_\infty = 0.6$).

Type	$\mu_{c_{d_{max}}}$	$\sigma^2_{c_{d_{max}}}$	$\mu_{c_{l_{min}}}$	$\sigma_{c_{l_{min}}}$	No. of Function / Gradient Evals.
IMCS-BCO (Kriging)	$2.93 \cdot 10^{-3}$	$3.07 \cdot 10^{-7}$	0.619	$1.86 \cdot 10^{-2}$	23/23
MCS-BCO	$2.73 \cdot 10^{-3}$	$2.50 \cdot 10^{-7}$	0.618	$1.84 \cdot 10^{-2}$	23/23
IMCS-BCO (PC)	$2.96 \cdot 10^{-3}$	$2.86 \cdot 10^{-7}$	0.619	$1.86 \cdot 10^{-2}$	6153/6153
MCS-BCO	$2.65 \cdot 10^{-3}$	$2.90 \cdot 10^{-8}$	0.620	$1.81 \cdot 10^{-2}$	6152/6152

CHAPTER X

CONCLUSIONS

10.1 Summary of Results

A dynamic training point selection framework has been proposed and applied to two different surrogate models: the kriging and polynomial chaos expansions. It provides a better choice of training point locations for both surrogate models since it inherits the characteristics of both domain- and response-based training point selection methods, *i.e.,* a geometric criterion is used to spread out the points and the points are chosen based on discrepancies between local and global surrogate approximated response values. Comparisons with latin hypercube sampling and other domain-based training point selection approaches show that more monotone convergence behavior and better accuracies are achieved.

The framework also addresses the question of training point selection in the presence of higher-order derivative information: the local surrogate models use the available derivative information in approximating the function values and influence the training point selection via the discrepancy function. As a result, it is shown that the dynamic method ably chooses better locations to evaluate the gradient and Hessian information than LHS. In this process,

polynomial chaos has been newly enhanced to incorporate Hessian information. The computational advantage of using higher-order derivative information for the construction of surrogate models in conjunction with the proposed training point selection is also discussed.

The framework also introduces a local as well as global measure of surrogate model accuracy, where the global error measure (RMSD) is the mean of all local error measures, δ. The proposed MAD and RMSD show great promise for measuring surrogate model error in applications of practical interest where it is intractable to calculate the actual errors (MAE or RMSE). This is demonstrated by an excellent agreement between the proposed measures and actual errors for a variety of analytical test functions for both surrogate models. However, for the aerodynamic test function, the proposed error measures are not as accurate as they are for analytical test functions, due to the difficulties of the MIR local surrogate in approximating non-smooth functions. The error estimate does not require additional exact function evaluations and the extra computational overhead of building and evaluating local surrogate models is negligible compared to most high-fidelity physics-based function evaluations.

This work also compares the performances of kriging and polynomial chaos surrogate models on analytical and aerodynamic test problems in terms of model accuracy. The performances of both surrogate models are also assessed when applied to uncertainty quantification and robust optimization under uncertainty (mixed epistemic/aleatory) on structural and aerodynamic test problems.

10.2 Novel Contributions

1. **Training Point Selection:** A new framework for surrogate model training point selection is the main contribution of this work. The dynamic method is shown to

perform better than commonly used methods for DoE. Many methods exist in the literature for training point selection in the absence of derivative information. This work also allows better choices of locations where to evaluate the gradients as well as Hessian.

2. **Error Estimation:** This work proposes an error estimate to the surrogate models in the form of a discrepancy function. This function is shown to exhibit close agreement with the actual distribution of error.

3. **Hessian Enhancement of PCE:** The polynomial chaos (regression procedure) is newly enhanced to incorporate Hessian information as additional fitting conditions.

10.3 Recommendations for Future Work

In summary, the presented framework has shown promise to improve training point selection as well as estimate the error of the resulting surrogate model. The following is recommended as future research avenues.

10.3.1 On the Proposed Framework

- The suitability of the training point selection method for building globally accurate surrogate models has been explored in this work, however, its suitability pertaining to surrogate-based optimization needs investigation. The framework currently ensures at least one mean distance (see Eq. (5.4)) between a new and existing training point. The proposed framework can be adapted for optimizations, for example, by means of tuning the *control parameter* discussed in section 5.1 which controls the placement of training points closer to an existing training point (if needed by the optimizer). On the other hand, the proposed error estimate driven by the discrepancy function ($\delta =$

$|\widehat{f}_{global} - \widehat{f}_{local}|$) can readily be used for optimizations, but this is also not investigated in this work. A straightforward investigation of the suitability of the error estimation framework for validating the statistics (e.g. mean and variance) produced by the global surrogate model is also recommended as a research avenue.

- The assumption that a local surrogate model is more accurate than a global one becomes less reasonable in some test cases such as the aerodynamic problem studied in section VII. It is recommended to study other potential candidates for local surrogate models (e.g. neural networks [85], radial basis function [84]), particularly for non-smooth functions. Also, considerable efforts have gone into tuning the MIR local surrogate model in terms of the parameters which influence the approximation [48,49]. A surrogate model that is relatively insensitive to the function to be modeled and without any tunable parameters will be an attractive candidate to serve as a local surrogate model. Along the same line of discussion, the use of two or more local surrogate models for added fidelity is also a recommended topic of investigation.

- The training point selection and error estimation framework has been applied to kriging and polynomial chaos. The application of the framework to other existing surrogate modeling methods can also be considered.

10.3.2 On Optimization Under Uncertainty

This work identifies the following as potential research areas in applying the surrogate models to uncertainty quantification and optimization under uncertainty.

- The gradient approximation for epistemic variables discussed in section 8.3.4.2 needs further refinement.

- In this work the random variables are assumed to be *uncorrelated* as well as *normally* distributed. This naturally leads into the investigation of correlated random variables and other input distributions.

- The OUU framework (IMCS-BCO), can be applied to complex problems of interest, for example, structural optimization of a wing, by coupling the OUU framework with specialized finite-element solvers.

BIBLIOGRAPHY

[1] Pironneau, O., "On Optimum Design in Fluid Mechanics," *Journal of Fluid Mechanics*, Vol. 64, No. 1, 1974, pp. 97–110. 2, 9

[2] Jameson, A., "Optimum Aerodynamic Design Using Control Theory," Computational Fluid Dynamics Review, Hafez, M., Oshima, K. (eds), Wiley: New York, pp. 495-528, 1995. 2, 9

[3] Errico, R. M., "What is an adjoint model?" *Bulletin of the American Meteorological Society*, Vol. 8(11), 1997, pp. 2577–2591. 2, 9

[4] Wiener, N., "The homogenous chaos," *American Journal of Mathematics*, Vol. 60, 1938, pp. 897–936. 2, 22

[5] Xiu, D. and Karniadakis, G. E., "Modeling Uncertainty in Flow Simulations via Generalized Polynomial Chaos," *Journal of Computational Physics*, Vol. 187(1), 2003, pp. 137–167. 2, 22

[6] Cressie, N., "The Origins of Kriging," *Mathematical Geology*, Vol. 22, No. 3, 1990, pp. 239–252. 2, 13

[7] Koehler, J. R. and Owen, A. B., "Computer Experiments," *Handbook of Statistics, Ghosh, S., Rao, C.R., (Eds.)*, pp. 261-308, 1996. 2

[8] Yamazaki, W., Rumpfkeil, M. P., and Mavriplis, D. J., "Design Optimization Utilizing Gradient/Hessian Enhanced Surrogate Model," AIAA Paper, 2010-4363, 2010. 2, 9, 10, 19, 21, 63

[9] Kiris, C., Housman, J., and et. al., M. G., "Best Practices for Aero-Database CFD Simulations of Ares V Ascent," 49th AIAA Aerospace Meeting, Orlando, Florida, 2011. 3

[10] Freeman, Jr., D. C., Talay, T., and Austin, R., "Prediction of High-Speed Aerodynamic Characteristics Using Aerodynamic Preliminary Analysis System (APAS)," Reusable Launch Vehicle Technology Program, IAF 96-V.4.01, 1996. 3

[11] Arora, J. S., *Optimization of Structural and Mechanical Systems*, World Scientific Publishing Co. Pte. Ltd., 2007. 6, 14, 16, 33, 99, 120

[12] Agarwal, H., *Reliability Based Design Optimization: Formulations and Methodologies*, Ph.D. thesis, University of Notre Dame, 2004. 6, 99

[13] Padmanaban, D., *Reliability-Based Optimization for Multidisciplinary System Design*, Ph.D. thesis, University of Notre Dame, 2003. 6, 99

[14] Alvin, K. F., Oberkampf, W. L., Rutherford, B. M., and Diegert, K. V., "Methodology for Characterizing Modeling and Discretization Uncertainties in Computational Simulations," Tech. Rep. SAND2000-5015, Sandia National Laboratories, 2000. 6, 100

[15] Pilch, M., Trucano, T. G., and Helton, J. C., "Ideas Underlying Quantification of Margins and Uncertainties (QMU): A white paper," Tech. Rep. SAND2006-5001, Sandia National Laboratories, 2006. 6, 100

[16] Helton, J. C., Johnson, J. D., Oberkampf, W. L., and Storlie, C. B., "A sampling-based computational strategy for the representation of epistemic uncertainty in model predictions with evidence theory," Tech. Rep. SAND2006-5557, Sandia National Laboratories, 2006. 6, 100, 105

[17] Diegert, K., Klenke, S., Novotny, G., Paulsen, R., Pilch, M., and Trucano, T., "Toward a More Rigorous Application of Margins and Uncertainties within the Nuclear Weapons Life Cycle - A Sandia Perspective," Tech. Rep. SAND2007-6219, Sandia National Laboratories, 2007. 6, 100

[18] Helton, J. C., Oberkampf, J. D. J. W. L., and Sallaberry, C. J., "Representation of Analysis Results Involving Aleatory and Epistemic Uncertainty," Tech. Rep. SAND2008-4379, Sandia National Laboratories, 2008. 6, 100

[19] Ghate, D. and Giles, M. B., "Inexpensive Monte Carlo uncertainty analysis," *Recent Trends in Aerospace Design and Optimization, Tata McGraw-Hill, New Delhi*, 2006, pp. 203–210. 6, 9, 10

[20] Fang, K. T., Lin, D., and Winker, P., "Uniform design: Theory and application," *Technometrics*, Vol. 42, No. 3, 2000, pp. 237–248. 7

[21] Metropolis, N. and Ulam, S., "The Monte Carlo method," *Journal of the American Statistical Association*, Vol. 44, 1949, pp. 335–341. 7, 34

[22] McKay, M. D., Conover, W. J., and Beckman, R. J., "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics*, Vol. 21, No. 2, 1979, pp. 239–245. 7, 34

[23] Xiu, D., *Numerical Methods for Stochastic Computations: A Spectral Method Approach*, Princeton University Press. 7, 13, 22, 23, 24, 25, 39

[24] Maître, O. P. L. and Knio, O. M., *Spectral Methods for Uncertainty Quantification, Scientific Computation*, Springer. 7, 22, 23, 24, 25, 39

[25] Wong, T. T., Luk, W. S., and Heng, P. A., "Sampling with Hammersley and Halton points," *J. Graph. Tools*, Vol. 2, No. 2, Nov. 1997, pp. 9–24. 7, 37

[26] Sobol, I., *A primer for the Monte Carlo Method*, CRC Press, 1994. 7, 34, 37, 58

[27] Rosenbaum, B. and Schulz, V., "Comparing sampling strategies for aerodynamic Kriging surrogate models," *Journal of Applied Mathematics and Mechanics*, Vol. 92, 2012, pp. 852–868. 7, 13

[28] Mehmani, A., Chowdhury, S., Zhang, J., and Messac, A., "Regional Error Estimation of Surrogates (REES)," AIAA Paper, 2012-5707, 2012. 7, 8, 51

[29] Alexandrov, N. M., Dennis, J. E., Lewis, R. M., and Torczon, V., "A Trust-Region Framework for Managing the Use of Approximation Models in Optimization," *Structural Optimization*, Vol. 15, 1998, pp. 16–23. 7, 43

[30] Roderick, O., Anitescu, M., and Fischer, P., "Polynomial Regression Approaches Using Derivative Information for Uncertainty Quantification," *J. of Nuclear Science and Engineering*, Vol. 164, No.2, 2010, pp. 122–139. 7, 26, 27

[31] Cheng, H. and Sandu, A., "Collocation least-squares polynomial chaos method." SCS/ACM, 2010. 7

[32] Bozdogan, H., "Akaike's Information Criterion and Recent Developments in Information Complexity," *J. Math. Psychol.*, Vol. 44, No. 1, March 2000, pp. 62–91. 8

[33] Queipo, N., Haftka, R., Shyy, W., Goel, T., Vaidhyanathan, R., and Tucker, P., "Surrogate-based Analysis and Optimization," *Progress in Aerospace Sciences*, Vol. 41, No. 1, 2005, pp. 1–28. 8, 47

[34] Chung, H. S. and Alonso, J. J., "Using Gradients to Construct Cokriging Approximation Models for High-Dimensional Design Optimization Problems," AIAA Paper, 2002-0317, 2002. 9, 13

[35] Laurenceau, J. and Sagaut, P., "Building Efficient Response Surfaces of Aerodynamic Functions with Kriging and Cokriging," *AIAA Journal*, Vol. 46, No. 2, 2008, pp. 498–507. 9, 13

[36] Mani, K. and Mavriplis, D. J., "Unsteady Discrete Adjoint Formulation for Two-Dimensional Flow Problems with Deforming Meshes," *AIAA Journal*, Vol. 46 No. 6, 2008, pp. 1351–1364. 9, 131

[37] Sherman, L. L., Taylor III, A. C., Green, L. L., and Newman, P. A., "First- and second-order aerodynamic sensitivity derivatives via automatic differentiation with incremental iterative methods," *Journal of Computational Physics*, Vol. 129, 1996, pp. 307 – 331. 9, 10

[38] Taylor III, A. C., Green, L. L., Newman, P. A., and Putko, M., "Some Advanced Concepts in Discrete Aerodynamic Sensitivity Analysis," *AIAA Journal*, Vol. 41 No. 7, 2003, pp. 1224–1229. 9

[39] Chalot, F., Dinh, Q., Herbin, E., Martin, L., Ravachol, M., and Roge, G., "Estimation of the impact of geometrical uncertainties on aerodynamic coefficients using CFD," AIAA Paper, 2068-2008, April, 2008. 9

[40] Rumpfkeil, M. P. and Mavriplis, D. J., "Efficient Hessian Calculations using Automatic Differentiation and the Adjoint Method," AIAA Paper, 2010-1268, January, 2010. 9

[41] Rumpfkeil, M. P. and Mavriplis, D. J., "Efficient Hessian Calculations using Automatic Differentiation and the Adjoint Method with Applications," *AIAA Journal*, Vol. 48, No. 10, 2010, pp. 2406–2417. 9, 10

[42] Mavripilis, D. J., "Aerodynamic Drag Prediction Using Unstructured Mesh Solvers, VKI Lecture Notes, CFD-Based Aircraft Drag Prediction and Reduction, Von Karman Institute for Fluid Dynamics, Rhode-Saint-Genese, Belgium," 2003. 10, 21

[43] Keane, A. and Nair, P., *Computational Approaches for Aerospace Design*, John Wiley & Sons, 2005. 12, 13, 34, 51, 98, 99, 100, 102, 103

[44] Gallatly, R. A., Berke, L., and Gibson, W., "The Use of Optimality Criteria in Automated Structural Design," 3^{rd} Conference on Matrix Methods in Structural Mechanics, WPAFB, Ohio, 1971. 12

[45] Schmit, L. A. and Farshi, B., "Some Approximation Concepts for Structural Synthesis," *AIAA Journal*, Vol. 12, 1974, pp. 692–699. 12

[46] Schmit, L. A., "Structural Synthesis – Its Genesis and Development," *AIAA Journal*, Vol. 19, No. 10, 1981, pp. 1249–1263. 13

[47] Yamazaki, W., Mouton, S., and Carrier, G., "Efficient Design Optimization by Physics-Based Direct Manipulation Free-Form Deformation," AIAA Paper, 2008-5953, 2008. 13, 19

[48] Wang, Q., Moin, P., and Iaccarino, G., "A rational interpolation scheme with super-polynomial rate of convergence," *SIAM Journal of Numerical Analysis*, Vol. 47, No. 6, 2010, pp. 4073–4097. 13, 29, 32, 52, 61, 63, 145

[49] Wang, Q., Moin, P., and Iaccarino, G., "A High-Order Multi-Variate Approximation Scheme for Arbitrary Data Sets," *Journal of Computational Physics*, Vol. 229, No. 18, 2010, pp. 6343–6361. 13, 29, 32, 52, 61, 63, 145

[50] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P., "Design and Analysis of Computer Experiments," *Statistical Science*, Vol. (4), 1989, pp. 409–423. 13

[51] Jeong, S., Murayama, M., and Yamamoto, K., "Efficient Optimization Design Method Using Kriging Model," *Journal of Aircraft*, Vol. 42, No. 2, 2005, pp. 413–420. 13

[52] Martin, J. D. and Simpson, T. W., "Use of Kriging Models to Approximate Deterministic Computer Models," *AIAA Journal*, Vol. 43, No.4, 2005, pp. 853–863,. 13

[53] Han, Z. H., Goertz, S., and Zimmermann, R., "Improving variable-fidelity surrogate modeling via gradient-enhanced kriging and a generalized hybrid bridge function," *Aerospace Science and Technology*, doi:10.1016/j.ast.2012.01.006, 2012. 13, 21, 63, 88, 96

[54] Rosenbaum, B. and Schulz, V., "Efficient response surface methods based on generic surrogate models," *SIAM Journal of Scientific Computing*, Vol. 35, No. 2, 2013, pp. B529–B550. 13, 19

[55] Yamazaki, W., "Uncertainty Quantification via Variable Fidelity Kriging Model," *Japan Society of Aeronautical Space Sciences*, Vol. 60, 2012, pp. 80–88. 13, 21, 58, 63, 88, 96

[56] Rumpfkeil, M. P., Yamazaki, W., and Mavriplis, D. J., "Uncertainty Analysis Utilizing Gradient and Hessian Information," Sixth International Conference on Computational Fluid Dynamics, ICCFD6, St. Petersburg, Russia, July 12-16, 2010. 13, 19

[57] Jones, D. R., "A taxonomy of global optimization methods based on response surfaces," *Journal of Global Optimizations*, Vol. 21, 2001, pp. 345–383. 13

[58] Forrester, A., Sobester, A., and Keane, A., *Engineering Design via Surrogate Modelling: A Practical Guide*, John Wiley & Sons, 2008. 16, 34

[59] Xiu, D. and Karniadakis, G. E., "The Wiener-Askey Polynomial Chaos for Stochastic Differential Equations," *SIAM Journal of Scientific Computing*, Vol. 24, No. 2, 2002, pp. 619–644. 22, 24

[60] Cameron, R. and Martin, W., "The orthogonal development of nonlinear functionals in series of Fourier-Hermite functionals," *Ann. Math.*, Vol. 48, 1947, pp. 385. 22

[61] Ghanem, R. and Spanos, P. D., *Stochastic Finite Elements: A Spectral Approach*, New York: Springer, 1991. 22

[62] Walters, R. W., "Towards Stochastic Fluid Mechanics via Polynomial Chaos," AIAA Paper, 2003-0413, 2003. 22

[63] Elred, M. S., Webster, C. G., and Constantine, P. G., "Evaluation of Non-Intrusive Approaches for Wiener-Askey Generalized Polynomial Chaos," AIAA Paper, 2008-1892, 2008. 22, 26

[64] Ghanem, R. and Spanos, P. D., *Stochastic Finite Elements: A Spectral Approach (2nd edition)*, New York: Springer, 1991. 23, 24, 25

[65] Jones, B. A., Doostan, A., and Born, G. H., "Nonlinear Propagation of Orbit Uncertainty Using Non-Intrusive Polynomial Chaos," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 36, No.2, 2013, pp. 415–425. 24

[66] Tatang, M. A., Pan, W., Prinn, R. G., and McRae, G., "An efficient method for parametric uncertainty analysis of numerical geophysical models," *J. of Geophysical Research*, Vol. 102, No.D18, doi:10.1029/97JD01654, 1997, pp. 21925–21932. 25

[67] Elred, M. S., "Recent Advances in Non-Intrusive Polynomial Chaos and Stochastic Collocation Methods for Uncertainty Analysis and Design," AIAA Paper, 2009-2274, 2009. 25

[68] Li, Y., Anitescu, M., Roderick, O., and Hickernell, F., "Orthogonal Bases for Polynomial Regression with Derivative Information in Uncertainty Quantification," *International Journal for Uncertainty Quantification*, Vol. 1, No.4, 2011, pp. 297–320. 27

[69] Fishman, G., *Monte-Carlo: Concepts, Algorithms, and Applications*, New York: Springer-Verlag, 1996. 34

[70] Ecuyer, D. L., *Monte Carlo and Quasi-Monte Carlo Methods*, Springer, Berlin, Heidelberg. 34, 36, 37

[71] Rumpfkeil, M. P., Yamazaki, W., and Mavriplis, D. J., "A Dynamic Sampling Method for Kriging and Cokriging Surrogate Models," AIAA Paper, 2011-883, 2011. 35

[72] Holtz, M., *Sparse Grid Quadrature in High Dimensions with Applications in Finance and Insurance*, Springer, New York, 2011. 40

[73] Jones, D. R., Schonlau, M., and Welch, W. J., "Efficient Global Optimization of Expensive Black-Box Functions," *Journal of Global Optimization*, Vol. 13, 1998, pp. 455–492. 41

[74] Jin, R., Chen, W., and Simpson, T. W., "Comparative Studies of Metamodeling Techniques under Multiple Modeling Criteria," *Structural and Multidisciplinary Optimization*, Vol. 23, 2001, pp. 1–13. 45, 46

[75] Efron, B., "Bootstrap Methods: Another Look at the jackknife," *Annals of Statistics*, Vol. 7, 1979, pp. 1–26. 48

[76] Efron, B., *An introduction to the bootstrap*, New York: Chapman & Hall, 1993. 48

[77] Chernick, M. R., *Bootstrap methods: A guide for practitioners and Researchers (2nd ed.)*, NJ: John Wiley & Sons, Inc, 2008. 48

[78] Efron, B. and Tibshirani, R., "Improvements on cross-validation: The .632+ bootstrap method," *Journal of the American Statistical Association*, Vol. 92 No.438, 1997, pp. 548–560. 48, 49, 50

[79] Lachenbruch, P. A. and Mickey, M. R., "Estimation of error rates in discriminant analysis," *Technometrics*, Vol. 10, 1968, pp. 1–11. 49, 67

[80] Efron, B., "Estimating the error rate of a prediction rule: Some improvements on cross-validation," *Journal of the American Statistical Association*, Vol. 78, 1983, pp. 316–331. 49, 50, 67

[81] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, Vol. 12, 2011, pp. 2825–2830. 53

[82] Dyn, N., Light, W., and Cheney, E., "Interpolation by piecewise-linear radial basis functions, I," *Journal of Approximation Theory*, Vol. 59, No. 2, 1989, pp. 202 – 223. 56

[83] Desai, A., Witteveen, J. A. S., and Sarkar, S., "Uncertainty Quantification of a Nonlinear Aeroelastic System Using Polynomial Chaos Expansion With Constant Phase Interpolation," *Journal of Vibration and Acoustics*, Vol. 135, No. 5. 56

[84] Buhmann, M., *Radial Basis Functions (1st edn)*, Cambridge University Press: Cambridge, 2005. 57, 145

[85] Cochocki, A. and Unbehauen, R., *Neural Networks for Optimization and Signal Processing*, John Wiley & Sons, Inc., New York, NY, USA, 1st ed., 1993. 57, 145

[86] Han, Z. H., Zimmermann, R., and Goertz, S., "Alternative Cokriging Method for Variable-Fidelity Surrogate Modeling," *AIAA Journal*, Vol. 50, No. 5, 2012, pp. 1205–1210. 63, 88, 96

[87] Han, Z. H., Zimmermann, R., and Goertz, S., "On Improving Efficiency and Accuracy of Variable-Fidelity Surrogate Modeling in Aero-data for Loads Context," CEAS 2009 European Air and Space Conference, 2009. 63, 88, 96

[88] Han, Z. H., Zimmermann, R., and Goertz, S., "A New Cokriging Method for Variable-Fidelity Surrogate Modeling of Aerodynamic Data," AIAA Paper, 2010-1225, 2010. 63, 88, 96

[89] Yamazaki, W. and Mavripilis, D. J., "Derivative-Enhanced Variable Fidelity Surrogate Modeling for Aerodynamic Functions," *AIAA Journal*, Vol. 51, No. 1, 2013, pp. 126–137. 63, 88, 96

[90] Yamazaki, W. and Mavriplis, D. J., "Derivative-Enhanced Variable Fidelity Surrogate Modeling for Aerodynamic Functions," AIAA Paper, 2011-1172, 2011. 63, 88, 96

[91] Sheshadri, P., Constantine, P., Gonnet, P., and Parks, G. T., "Sparse Robust Rational Interpolation for Parameter-dependent Aerospace Models," AIAA Paper, 2013-1680, 2013. 65, 80

[92] Wendland, H., *Scattered Data Approximation (1st edn)*, Cambridge University Press: Cambridge, 2005. 76

[93] Hawkins, D. M., "The Problem of Overfitting," *J. Chem. Inf. Comput. Sci.*, Vol. 44, 2004, pp. 1–12. 80

[94] Mani, K. and Mavriplis, D. J., "An Unsteady Discrete Adjoint Formulation for Two-Dimensional Flow Problems with Deforming Meshes," AIAA Paper, 2007-60, 2007. 89, 131

[95] Mani, K. and Mavriplis, D. J., "Discrete Adjoint Based Time-Step Adaptation and Error Reduction in Unsteady Flow Problems," AIAA Paper, 2007-3944, 2007. 89, 131

[96] Lockwood, B. A. and Anitescu, M., "Gradient-Enhanced Universal Kriging for Uncertainty Propagation in Nuclear Engineering," Preprint ANL/MCS-P1833-0111, 2011. 106, 108, 111

[97] Lockwood, B., Anitescu, M., and Mavriplis, D. J., " Mixed Aleatory/Epistemic Uncertainty Quantification for Hypersonic Flows via Gradient-Based Optimization and Surrogate Models," AIAA Paper, 2012-1254, 2012. 106, 108, 111

[98] Lockwood, B., Rumpfkeil, M. P., Yamazaki, W., and Mavriplis, D. J., "Uncertainty Quantification in Viscous Hypersonic Flows using Gradient Information," AIAA Paper, 2011-885, 2011. 106, 108, 111

[99] Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C., "A Limited Memory Algorithm for Bound Constrained Optimization," *SIAM Journal on Scientific Computing*, Vol. 16(5), 1995, pp. 1190–1208. 107

[100] Zhu, C., Byrd, R. H., Lu, P., and Nocedal, J., "L-BFGS-B: A Limited Memory FORTRAN Code for Solving Bound Constrained Optimization Problems," Tech. Rep. NAM-11, Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, Illinois, USA, 1994. 107

[101] Rumpfkeil, M. P., "Optimizations Under Uncertainty Using Gradients, Hessians, and Surrogate Models," *AIAA Journal*, Vol. 51, No. 2, 2013, pp. 444–451. 108, 111, 112, 113

[102] Putko, M. M., Newmann, P. A., Taylor III, A. C., and Green, L. L., "Approach for uncertainty propagation and robust design in CFD using sensitivity derivatives," AIAA Paper, 2001-2528, June, 2001. 109, 110

[103] Du, X. and Chen, W., "Methodology for Managing the Effect of Uncertainty in Simulation-Based Design," *AIAA Journal*, Vol. 38(8), 2000, pp. 1471–1478. 110

[104] Parkinson, A., Sorensen, C., and Pourhassan, N., "A general approach for robust optimal design," *Trans. ASME*, Vol. 115, 1993, pp. 74–80. 110

[105] Putko, M. M., Taylor III, A. C., Newmann, P. A., and Green, L. L., "Approach for Input Uncertainty Propagation and Robust Design in CFD Using Sensitivity Derivatives," *Journal of Fluids Engineering*, Vol. 124(1), 2002, pp. 60–69. 110

[106] Waechter, A. and Biegler, L. T., "On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming," *Mathematical Programming*, Vol. 106(1), 2006, pp. 25–57. 111

[107] Boopathy, K. and Rumpfkeil, M. P., "A Unified Framework for Training Point Selection and Error Estimation for Surrogate Models," *AIAA Journal*, Vol. In Revision., 2013. 111, 119

[108] Boopathy, K. and Rumpfkeil, M. P., "A Multivariate Interpolation and Regression Enhanced Kriging Surrogate Model," AIAA Paper, 2013-2964. 111, 119

[109] Hicks, R. and Henne, P., "Wing Design by Numerical Optimization," *Journal of Aircraft*, Vol. 15 No. 7, 1978, pp. 407 – 412. 131

[110] Batina, J. T., "Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes," *AIAA Journal*, Vol. 28, No. 8, 1990, pp. 1381 – 1388. 131

APPENDIX A

FINITE ELEMENT PROCEDURE FOR THREE-BAR TRUSS ANALYSIS

The finite element procedure adopted for obtaining the nodal displacements (u_x and u_y) as well as the elemental stresses (σ_1, σ_2, and σ_3) for the three-bar truss problem (see section 9.1) is discussed here. The structure is assumed to have three elements with two degrees of freedom at each node (*i.e.,* a two-dimensional truss analysis). The element connectivity information is given in Table A.1.

Table A.1: Connectivity of elements.

Element	Node		Element length	$\lambda_e = \cos(\phi_e)$	$\mu_e = \sin(\phi_e)$
	i	j	l_e		
1	1	4	$L_1 = H/\sin(\phi_1)$	$\cos(\phi_1)$	$\sin(\phi_1)$
2	2	4	$L_2 = H/\sin(\phi_2)$	$\cos(\phi_2)$	$\sin(\phi_2)$
3	3	4	$L_3 = H/\sin(\phi_3)$	$\cos(\phi_3)$	$\sin(\phi_3)$

The stiffness matrix of the e-th element in global coordinate system is given by,

$$[\boldsymbol{k}]_e = \frac{E_e A_e}{L_e} \begin{bmatrix} {\lambda_e}^2 & \lambda_e\mu_e & -{\lambda_e}^2 & -\lambda_e\mu_e \\ \lambda_e\mu_e & {\mu_e}^2 & -\lambda_e\mu_e & -{\mu_e}^2 \\ -{\lambda_e}^2 & -\lambda_e\mu_e & {\lambda_e}^2 & \lambda_e\mu_e \\ -\lambda_e\mu_e & -{\mu_e}^2 & \lambda_e\mu_e & {\mu_e}^2 \end{bmatrix}. \tag{1.1}$$

The individual stiffness matrices of each element: $[\boldsymbol{k}]_1$, $[\boldsymbol{k}]_2$, and $[\boldsymbol{k}]_3$ are obtained using

Eq. (1.1) and Table A.1. The elemental stiffness matrices are assembled to form the global stiffness matrix $\boldsymbol{K}$ of size 8×8 (not shown here). After applying the boundary conditions (i.e. x- and y-displacements at nodes 1, 2 and 3 are zero) and performing elimination, a simplified linear system is obtained: $\boldsymbol{KQ} = \boldsymbol{F}$, where $\boldsymbol{Q} = \left\{ \begin{array}{c} Q_{4x} \\ Q_{4y} \end{array} \right\}$, $\boldsymbol{F} = \left\{ \begin{array}{c} F_{4x} \\ F_{4y} \end{array} \right\}$ and $\boldsymbol{K} = \left[\begin{array}{cc} \frac{E_1 A_1}{L_1}{\lambda_1}^2 + \frac{E_2 A_2}{L_2}{\lambda_2}^2 + \frac{E_3 A_3}{L_3}{\lambda_3}^2 & \frac{E_1 A_1}{L_1}\lambda_1\mu_1 + \frac{E_2 A_2}{L_2}\lambda_2\mu_2 + \frac{E_3 A_3}{L_3}\lambda_3\mu_3 \\ \frac{E_1 A_1}{L_1}\lambda_1\mu_1 + \frac{E_2 A_2}{L_2}\lambda_2\mu_2 + \frac{E_3 A_3}{L_3}\lambda_3\mu_3 & \frac{E_1 A_1}{L_1}{\mu_1}^2 + \frac{E_2 A_2}{L_2}{\mu_2}^2 + \frac{E_3 A_3}{L_3}{\mu_3}^2 \end{array} \right]$.

The x- and y-displacements at node 4 are given by, $\boldsymbol{Q} = \boldsymbol{K}^{-1}\boldsymbol{F}$. Once the nodal displacements are found, the stresses in element $i - j$ can be calculated using:

$$\sigma_e = \frac{E_e}{L_e} \left\{ \begin{array}{cccc} -\lambda_e & -\mu_e & \lambda_e & \mu_e \end{array} \right\} \left\{ \begin{array}{c} Q_{ix} \\ Q_{iy} \\ Q_{jx} \\ Q_{jy} \end{array} \right\}. \tag{1.2}$$

Using Eq. (1.2) and Table A.1, expressions for the stresses acting on each element can be obtained:

$$\begin{aligned} \sigma_1 &= \frac{E_1}{L_1} \left\{ \begin{array}{cccc} -\lambda_1 & -\mu_1 & \lambda_1 & \mu_1 \end{array} \right\} \left\{ \begin{array}{c} 0 \\ 0 \\ Q_{4x} \\ Q_{4y} \end{array} \right\} = \frac{E_1}{L_1} \left(Q_{4x}\lambda_1 + Q_{4y}\mu_1 \right), \\ \sigma_2 &= \frac{E_2}{L_2} \left\{ \begin{array}{cccc} -\lambda_2 & -\mu_2 & \lambda_2 & \mu_2 \end{array} \right\} \left\{ \begin{array}{c} 0 \\ 0 \\ Q_{4x} \\ Q_{4y} \end{array} \right\} = \frac{E_2}{L_2} \left(Q_{4x}\lambda_2 + Q_{4y}\mu_2 \right), \\ \sigma_3 &= \frac{E_3}{L_3} \left\{ \begin{array}{cccc} -\lambda_3 & -\mu_3 & \lambda_3 & \mu_3 \end{array} \right\} \left\{ \begin{array}{c} 0 \\ 0 \\ Q_{4x} \\ Q_{4y} \end{array} \right\} = \frac{E_3}{L_3} \left(Q_{4x}\lambda_3 + Q_{4y}\mu_3 \right). \end{aligned} \tag{1.3}$$

The constraints (for the optimization problem) can be evaluated by substituting Eq (1.3) into Eq. (9.1). The gradients of the constraints and objective function with respect to the design variables are obtained via differentiation with Maple.