Metamodel-Based Probabilistic Design for Dynamic Systems with Degrading Components

by

Turuna Saraswati Seecharan

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2012

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The probabilistic design of dynamic systems with degrading components is difficult. Design of dynamic systems typically involves the optimization of a time-invariant performance measure, such as Energy, that is estimated using a dynamic response, such as angular speed. The mechanistic models developed to approximate this performance measure are too complicated to be used with simple design calculations and lead to lengthy simulations. When degradation of the components is assumed, in order to determine suitable service times, estimation of the failure probability over the product lifetime is required. Again, complex mechanistic models lead to lengthy lifetime simulations when the Monte Carlo method is used to evaluate probability.

Based on these problems, an efficient methodology is presented for probabilistic design of dynamic systems and to estimate the cumulative distribution function of the time to failure of a performance measure when degradation of the components is assumed. The four main steps include; 1) transforming the dynamic response into a set of static responses at discrete cycle-time steps and using Singular Value Decomposition to efficiently estimate a time-invariant performance measure that is based upon a dynamic response, 2) replacing the mechanistic model with an approximating function, known as a “metamodel” 3) searching for the best design parameters using fast integration methods such as the First Order Reliability Method and 4) building the cumulative distribution function using the summation of the incremental failure probabilities, that are estimated using the set-theory method, over the planned lifetime.

The first step of the methodology uses design of experiments or sampling techniques to select a sample of training sets of the design variables. These training sets are then input to the computer-based simulation of the mechanistic model to produce a matrix of corresponding responses at discrete cycle-times. Although metamodels can be built at each time-specific column of this matrix, this method is slow especially if the number of time steps is large. An efficient alternative uses Singular Value Decomposition to split the response matrix into two matrices containing only design-variable-specific and time-specific information. The second step of the methodology fits metamodels only for the significant columns of the matrix containing the design variable-specific information. Using the time-specific matrix, a metamodel is quickly developed at any cycle-time step or for any time-invariant performance measure such as energy consumed over the cycle-lifetime. In the third step, design variables are treated as random variables and the First

Order Reliability Method is used to search for the best design parameters. Finally, the components most likely to degrade are modelled using either a degradation path or a marginal distribution model and, using the First Order Reliability Method or a Monte Carlo Simulation to estimate probability, the cumulative failure probability is plotted. The speed and accuracy of the methodology using three metamodels, the Regression model, Kriging and the Radial Basis Function, is investigated.

This thesis shows that the metamodel offers a significantly faster and accurate alternative to using mechanistic models for both probabilistic design optimization and for estimating the cumulative distribution function. For design using the First-Order Reliability Method to estimate probability, the Regression Model is the fastest and the Radial Basis Function is the slowest. Kriging is shown to be accurate and faster than the Radial Basis Function but its computation time is still slower than the Regression Model. When estimating the cumulative distribution function, metamodels are more than 100 times faster than the mechanistic model and the error is less than ten percent when compared with the mechanistic model. Kriging and the Radial Basis Function are more accurate than the Regression Model and computation time is faster using the Monte Carlo Simulation to estimate probability than using the First-Order Reliability Method.

Acknowledgements

Without the following people, this thesis would not be possible and for their assistance I am grateful.

- My supervisor, Prof. Gordon Savage, for his guidance.
- My committee members Profs. Keith Hipel, Jonathan Kofman, Stefan Steiner and my external examiner Dr. Guangbin Yang for their comments and suggestions.
- My parents and brother for their support.

Table of Contents

List of Figures

List of Tables

Nomenclature

d_e	Degree of Polynomial
$g_i(\mathbf{v})$	The i^{th} Limit State Function
n_{LF}	Total number of LSFs
$tol_{\boldsymbol{v}}$	Vector of the tolerances of the design variables
t_l	l^{th} service time
t_{p}	Planned Lifetime
Δt	Service time increment.
$x_{\mathrm{i,j}}$	The i^{th} run of the j^{th} design variable
$\hat{z}(\mathbf{v})$	Metamodel approximation of the mechanistic model.
A_l	The accumulation of all system instantaneous failure regions for all discrete service times up to t_l.
$\overline{A}_l$	The accumulation of all system instantaneous safe regions for all discrete service times up to t_l.
F_i	$g_i(\mathbf{v}) < 0$ or $g_i(\mathbf{u}) < 0$
$\Pr(F_i)$	$\Pr(g_i(\mathbf{v}) < 0)$ or $\Pr(g_i(\mathbf{v}) < 0)$
$\Pr(F_{l,i})$	$\Pr(g_i(\mathbf{v}, t_l) < 0)$
C	Total number of dynamic response time increments
N	Total number sample sets in a MCS
M	Total number of training sets
$\mathbf{d}_{\boldsymbol{h}}$	The vector of responses in the h^{th} column of $\mathbf{D}$.
$\mathbf{r}(\mathbf{v})$	Vector of metamodel functions
$\boldsymbol{\tau}$	Vector of cycle-time increments
$\mathbf{v}$	Vector of design variables
$\boldsymbol{w}^{\mathrm{T}}$	Row vector of metamodel fitting parameters
$\mathbf{w}_h^T$	Row vector of metamodel fitting parameters for column h in $\mathbf{D}$
$\mathbf{z}$	Vector of training responses
$\Pr(\mathbf{B}_l)$	The incremental failure probability over time Δt.
$\Pr\left(\mathbf{B}_l^{\backslash}\right)_T$	The true incremental failure probability

$\mathbf{B}_l$	The emergence of the incremental failure region from a safe region from time t_l during time interval Δt.
$\mathbf{D}$	Matrix of design-variable-feature information.
Pr($\mathbf{F}_l$)	The probability of system failure at the l^{th} service time.
Pr($\mathbf{F}$)	Probability of system failure
$\mathbf{Q}$	Matrix of time-feature information
$\mathbf{Q}_{\boldsymbol{l}}$	Matrix of time-feature information at t_l.
$\mathbf{X}$	Matrix of training design variable sets
$\mathbf{X}_\mathrm{a}$	Augmented Matrix used in RM to estimate model parameters
$\mathbf{X}_\mathbf{b}$	Matrix used in Kriging to estimate model parameters
$\mathbf{Z}$	Matrix of training response from a dynamic system
$\mathbf{\Gamma}$	Correlation matrix, in the Kriging model, whose elements represent a so-called correlation between training sets.
β_R	Reliability Index
τ_L	Cycle Lifetime
Φ	Normal CDF Function
$\beta_0, \beta_1, \ldots ,\beta_\mathrm{i}$	Metamodel parameters
λ	Number of levels of each design variable used to generate the training design
ζ	Upper or Lower Design Specification
η	Total number of design variables.
σ_{KG}^2	Kriging metamodel variance
σ_{RM}^2	RM variance
$\Delta\tau$	The increment used to obtain discrete samples of the continuous dynamic response.
$\theta_1, \ldots,\theta_\eta$	Kriging correlation function parameters
$\mu_\mathbf{V}$	Vector of the means of the design variables
μ_{V_i}	Mean of the i^th design variable
ψ	Performance Measure

Terminology and Abbreviations

Term	**Definition**
Cdf	Cumulative Distribution Function of time to failure
CV	Cross-Validation
CV-RMSE	Cross-Validation Root-Mean-Square-Error
CV-MAE	Cross-Validation Mean Absolute Error
FORM	First-Order Reliability Method
LSF	Limit-State Function
LSS	Limit-State Surface
MCS	Monte Carlo Simulation
MLFP	Most-Likely Failure Point
RM	Regression Model
RBF	Radial Basis Function
SVD	Singular Value Decomposition
Design Variables	The variables whose distribution parameters are optimized to reduce system failure.
Dynamic System	A system whose response changes with time.
Mechanistic Model	A mathematical representation of a physical system.
Metamodel	A simple model that approximates a complex or implicit mechanistic model.
Training Design	A sample of M design variable sets and their corresponding responses used to estimate metamodel parameters.
Static System	A system whose response does not change with time.
Cycle Time	The lifetime of the dynamic response

Chapter 1

Introduction

1.1 Problem Statement

The design of physical systems involves finding the values of the design variables that in turn position the performance measures to best meet their specification limits. Design variables include, for example, sizes, weights and material composition of parts. In order to allow convenient and cheap experimentation on the physical system, a mathematical representation, called the “mechanistic model”, is developed.

For real physical systems, experimentation to improve the design is costly and time consuming, especially when the experimental space is large. In response to this, mathematical models, or mechanistic models have been developed and computer simulations run to provide the behaviour. For a static system we have

$$\text{Static System: } f(\mathbf{z}, \mathbf{v}, \mathbf{d}) = 0 \tag{1.1}$$

Where **z** contains the responses and **v** represents the design variables. The solution is obtained by a Newton-Raphson iteration process. When the mechanistic model is complex and simulations are time consuming, the design process becomes onerous and thus a faster, metamodel, approach has been investigated. The idea is to use a few properly selected training sets of the design variables, simulate the outcomes and then fit a simpler, explicit, model.

In dynamic systems, the mechanistic models are usually in the form of differential equations that are typically non-linear and complex (Cochin 1980); (Esfandiari and Lu 2010) and (Ogata 2004) and we write

$$\text{Dynamic System: } \dot{\mathbf{z}}(\tau) = f(\mathbf{z}, \mathbf{v}, \mathbf{d}, \tau), \qquad \mathbf{z}(\tau = 0) = \mathbf{z}_0 \tag{1.2}$$

Where τ is the cycle time usually in seconds and **d** are the drivers. A response is now provided by a numerical simulation over the life-time. Most dynamic systems, such as position or velocity servo systems, automotive stability control systems, hammer mechanisms, hydraulic actuators, and so forth, are repetitive in nature and exhibit common performance measures. Examples include over-shoot at peak time, settling error at settling time, energy consumed over the life-time or common performance indices such as the integral of the squared error. Usually, there are multiple, competing performance measures.

Any degradation through aging and deterioration affects the design variables over time and this changes the problem to

$$\text{Degrading System: } \dot{\mathbf{z}} = F(\mathbf{z}, \mathbf{v}(t), \mathbf{d}, \tau), \qquad \mathbf{z}(\tau = 0) = \mathbf{z}_0 \tag{1.3}$$

where *t* is the service time typically in years and the design variables, **v**, due to degradation, are now a function of the service time. Clearly, the cycle-time dynamic responses change over service time. Based on customer needs, a set of design specifications are normally provided that involves a range within which an acceptable response should lie. Since experimentation on the physical system is costly, a mathematical representation is developed that can be used for design calculations.

Quality is the “goodness of fit” or the “conformance to specifications” of a product (Savage and Carr 2001). The design of the systems traditionally involves deterministic optimization where the design variables are identified and the optimization routine searches for the combination of the design variables that result in the best response. However, due to variations in manufacturing or environmental conditions that inevitably exists, the design variables are never the same value leading to the violation of the constraints by the response (Ju and Lee 2008); (Tu, Choi and Park 1999); (Jensen and Catalan 2007). Robust design techniques attempt to ensure that the system responses are insensitive to both the input uncertainty and component variations without actually eliminating the causes. This is done by first assuming that the design variables follow a particular distribution types whose distribution parameters are assumed to exist.

In quality design applications, the standard deviation is normally expressed as a percentage tolerance of the mean (Park 1996). Using the provided design specifications and the mechanistic model, a "limit-state function" is developed and the failure probability (the probability of not meeting design specifications) is estimated at the current mean and tolerance of the design variables (Madsen, Krenk and Lind 1986); (Melchers 1987). Not all combinations of the design variables will produce a response that meets design specifications. The simplest method to determine an acceptable system is to estimate the failure probability. Limits are now placed on the failure probability to determine an acceptable performing system. Probabilistic design, therefore, involves searching for the mean and tolerance of the design variables that result in an acceptable failure probability.

Mechanistic models of dynamic systems typically involve state equations that cannot be easily used in probability or quality based design methods. Therefore, in order to estimate the failure probability, sampling methods are used. Sampling methods, such as the Monte Carlo Simulation (MCS), importance sampling and adaptive sampling, are typically used to estimate failure when uncertainty is considered. Sampling methods involve finding the response at various possible combinations of the design variables and determining the number of occurrences in which the response falls outside of specification limits. Even though computer power is increasing, the complexity of the mechanistic models also increases (Youn and Choi 2004); (Gayton, Bourinet and Lemaire 2003); (Schueremans and Van Gemert 2005) leading to computer simulations that are still time-consuming. For an accurate probability estimation in any sampling method, typically $10^{n+2} \sim 10^{n+3}$ samples are needed to accurately compute a probability of failure of 10^{-n} (Andrieu-Renaud, Sudret and Lemaire 2004).

An alternative to these sampling methods are the fast integration methods (Wu, Millwater and Cruse 1990) such as the First Order Reliability Method, the Second-Order Reliability Method or the Advanced Mean Value Method. Normally, when provided with the distributions of the design variables, the estimation of the probability of failure would normally involve integrating the joint probability distribution function. However, this can lead to complicated integrals for correlated design variables. Fast integration methods approximate the integrand by transforming the variables to uncorrelated **u**-space and using a Taylor series expansion, the Most Likely Failure Point is approximated, along with the reliability index, that enables an efficient calculation of probability (Madsen and Tvedt 1990); (Chiralaksanakul and Mahadevan 2005); (Bucher and

Bourgund 1990). In most cases, due to the complexity of the physical system, complex or implicit mechanistic models lead to complex limit-state functions that are difficult to use with FORM or SORM.

Most researchers have started using “metamodels” to approximate the mechanistic model to allow simple analysis and design. Metamodels have been popular in response prediction (Barton 1998); (Clarke, Griebsch and Simpson 2005); (Hussain, Barton and Joshi 2002), deterministic optimization (Barton and Meckesheimer 2006); (Srivastava, et al. 2004) and even some reliability analysis (Deng 2006) but not in probability-based design optimization. Reliability is defined as the probability that, when operating under stated conditions, a system will perform its intended function successfully for a specified interval of time (Son 2006). The reliability of a system is important since degradation of the component inevitably exists. Degradation typically arises from environment conditions or stresses under which the component operates. Degradation is defined as the deterioration in quality, level or standard of performance of a functional unit that is an entity of hardware, software, or both, capable of accomplishing a specified purpose. Performance reliability is defined as the probability that performance measures of a system are within specification limits for the lifetime, conditional on the system being in a functional topology (Savage and Carr 2001). Furthermore, degradation must be accounted for due to its impact on quality, reliability and cost through aging and deterioration.

The traditional design methods for system performance reliability improvement using component degradation data have been based upon a sampling-based approach that uses Monte-Carlo simulation (MCS) to predict system reliability. More specifically, the sampling approach takes samples of the component distributions at time zero and traces their paths using their particular degradation function to provide time variant system responses. Through tracking and comparing the time-variant responses with critical specification limits, a system performance reliability function is predicted. However, there are the usual concerns when applying the sampling approach to predict reliability measures. For example, a large number of simulations and a vast computer memory are required to achieve a reasonable accuracy.

Design for quality has focused on the uncertainty issues at time zero and it has invoked robust design techniques such as Taguchi’s method (Park 1996). Furthermore, for an acceptable system, customers would require that the quality at the start of the lifetime (time zero) is high for an acceptable period of time.

For systems that exhibit aging, deterioration and degradation the problem is compounded and methods to find the distributions are not straightforward. Traditionally, lifetime distributions have been invoked; however, this approach is limited since there is usually insufficient data to estimate the required distribution. One way to overcome the disadvantage of lifetime distribution models is to model the degradation of the characteristics of the actual components (e.g. dimensions and material properties).

1.2 Motivation and Goals

Based upon the problems outlined in the previous section, this thesis is motivated by several factors

- The increasing complexity of mechanistic models that result in an increase in the CPU time of the computer-based simulation. Implicit or too complex models are too difficult to use in simple design calculations.
- Using the uncertainty inevitably present in design variables to search for a design that meets design specifications.
- The lack of research that uses metamodels to plot the failure of a system as it degrades.

The goal of this thesis is to develop an efficient and accurate methodology that combines Singular Value Decomposition with metamodels in order to search for the best design of dynamic systems when uncertainty is considered in its components. Also, when degradation is assumed in the components, the set-theory method will be used to estimate the incremental failure using two contiguous service time increments.

1.3 Proposed Methodology

1.3.1 Approach

The methodology involves four major steps:

1) Selecting the training design using some design of experiments technique or sampling method. For a dynamic system, the sampled dynamic responses are recorded as a set of static responses at discrete cycle time increments.
2) To this matrix of static responses at various training design variable sets, SVD is applied to get two matrices containing design-variable-specific and time-specific information. Metamodels are built for only the dominant columns of the design-variable-specific matrix.
3) Based upon the design specifications, the limit-state function is estimated, using the metamodel that is then used with FORM to estimate the failure probability. The best design parameters (mean and tolerance of the design variables) are then searched for using parameter design, tolerance design or integrated design methods.
4) Finally, the degradation is modelled using a particular type of degradation model and the incremental failure probability is estimated using the set-theory method.

In order to fit the metamodel, a sample of design variable sets and the corresponding responses must be obtained. Factorial designs or sampling methods select the sample of design variable combinations, known as “training sets”. These training sets are then input to a computer-based model of the mechanistic model to obtain the corresponding “training responses”. Now, in the case of a dynamic response, an estimate of a time-invariant performance measure is required, for design, that is based upon a time-varying response. The dynamic cycle lifetime of the response is broken into a set of discrete time increments/steps and the response at each of these steps is recorded into a matrix whose rows correspond to the response at each training set and the columns represent the response at the discrete cycle-time increments. This matrix consists of two main “feature spaces” which are the “time feature-space” and the “design-variable feature-space”. To this matrix, SVD is applied.

After SVD, three matrices are obtained; the first contains only the design-variable feature-space information of the original matrix, the second contains the so-called “singular values” and the third contains the time feature-space information. From principal component analysis the singular values quantify the importance of the rows of the first matrix (or the columns of the second matrix). Therefore, using only dominant singular values, a metamodel at any discrete time can be efficiently built and, ultimately, for the time-invariant performance measure such as the Integral Squared Error or Energy. This thesis investigates the speed and accuracy of three metamodels.

In the third step, using the metamodel estimate of the time-invariant response and the design specifications, a limit-state function is built. The fast integration method, FORM, is used to compute probability. In robust design applications, there are three types of design methods; parameter design searches for the means of the design variables that result in a reduced system failure, tolerance design moves the tolerances of the design variables while keeping the means constant and integrated design moves both the means and tolerances of the design variables.

In the final step, the degrading component is modelled a degradation path model. The intended lifetime is then broken into a series of discrete service time increments. At each service-time increments, the instantaneous failure probability is estimated and, using the set-theory method proposed by (Savage and Son 2011), the incremental failure probability is estimated. Finally, the cumulative distribution failure (*Cdf*) function is plotted to show how the system failure increases over time. A list of steps, shown below, illustrates the methodology required to plot the *Cdf* function of a dynamic degrading system.

1. Performance measures and their specification limits are selected. At initial time, t_0, the important component characteristics (through sensitivity analysis) are designated as design variables and M training sets are selected. Degradation rates are also determined through experimental analysis of a similar system to determine how the component degrades.

2. Discretize the service time by an increment Δt and let $\mathbf{t} = t_0, t_1,..t_l..t_L$ where $t_l = t_{l-1} + \Delta\tau$. For each service time, discretize the cycle time, of the response used to estimate the performance measure, by an increment $\Delta\tau$ so that $\boldsymbol{\tau} = \tau_0..\tau_c..\tau_{Lt}$ where $\tau_c = \tau_{c-1} + \Delta\tau$.

3. At service time t_l, the responses for the M experiments are simulated using the mechanistic model and evaluated at cycle time τ_c. For any t_l a matrix $[\mathbf{Z}_l]_{M\times C}$ is produced (Chapter 3).

4. Apply SVD to separate $[\mathbf{Z}_l]_{M\times C}$ into design variable space $[\mathbf{D}_l]_{M\times M}$ and time space $\left[\mathbf{Q}_l^T\right]_{M\times C}$ (Chapter 3).

5. Develop a metamodel for each significant column of $\mathbf{D}_l$ and store into a row vector $\hat{\mathbf{d}}^T(\mathbf{v})$ that is subsequently used to calculate an approximate dynamic response over $\boldsymbol{\tau}$ (Chapter 3).

6. Assign distributions (through experimental analysis or prior knowledge) to the design variables and degradation rates. Transform the limit-state functions that relate time-invariant performance measures and specification limits to **u**-space using a transformation method such as the Rosenblatt transformation (Chapter 4).

7. At each discrete service time, a sequence of limit-state functions is developed. Using the set-theory method, the incremental failure at service time t_l emerging from the safe region at t_{l-1} is obtained (Chapter 7).

1.3.2 Novelties and Significance

The novelties of this research are:

- Using SVD to quickly find a metamodel approximation of a time-invariant performance measure whose estimation depends upon a dynamic response.

- Combining the metamodel of the time-invariant response with FORM to search for the best design parameters (the mean and/or tolerance of each design variable) that meets specifications at discrete cycle times and for the performance measure.

- Combining set-theory and metamodels to estimate the *Cdf* and incremental failure probabilities, of a dynamic system, when degradation is considered in the components.

The speed and accuracy of three metamodels will be compared and conclusions will be based upon results obtained.

Example: For ease of presentation of the steps involved in building the *Cdf*, consider an explicit model of the over-damped decay of the shaft speed of a electro-mechanical servo

$$\omega(t) = e^{-\left(\frac{k}{R}\right)t}$$

Where k is a deterministic lumping of several parameters and R is the rotor winding resistance. Let R degrade according to a degradation path model then an expression for the speed in service time t_l and cycle time τ becomes

$$\omega(t_l, \tau) = e^{-\left(\frac{k}{(R_0+K_R t_l)}\right)\tau}$$

Let R_0 and K_R be uncertain, then for any service time t_l and samples of R_0 and K_R, the angle can be captured at any cycle time τ_k. In Figure 1-1 the uncertainty over cycle-time is shown plus the change in this uncertainty through aging of the resistance for initial time and some future service-time. The three curves show three samples of the dynamic response. The figure on the left shows the three sampled dynamic responses at the start of the service time, t_0, and the figure on the right shows the three samples at a later service time, t_l. Notice how the samples move which is due to the degradation of the resistance.

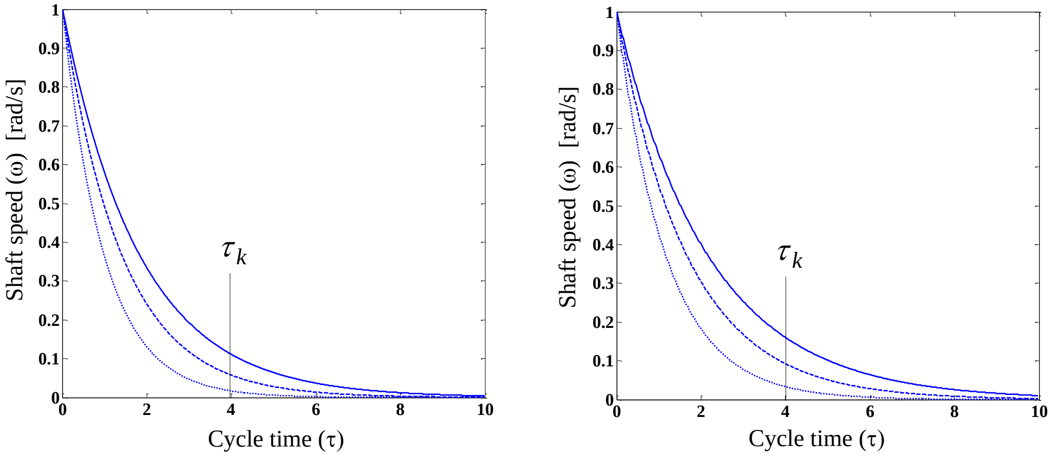


Figure 1-1: An uncertain dynamic response and the effects of degradation: (a) t = t_0 and (b) t = t_l

Suppose for service time t_l and through a few selected sets of R_0 and K_R with corresponding experiments, we are able to write a simple deterministic metamodel for the shaft speed at τ_k, say

$$\hat{z}(t_l, \tau_k) = \beta_0 + \beta_1 R_0 + \beta_2 K_R$$

where β_0, β_1 and β_2 are deterministic fitting parameters. We note that time has been subsumed and we are left with a function of the design variables. Then we treat R_0 and K_R as random variables to reintroduce the uncertainty in the angle. A choice of a specification limit, ζ, for the shaft speed leads to limit-state functions g_l and with Normal distribution parameters, for example, we have

$$g_l = \pm\left(\zeta - [\beta_0 + \beta_1(\mu_1 + \sigma_1 u_1) + \beta_2(\mu_2 + \sigma_2 u_2)]\right)$$

Given the distribution parameters, a plot of g_l = 0 for the axis variables (u_1, u_2) provides the limit-state surface from which the failure region and the associated joint density function provide the probability of failure. Then the associated joint density function of R_0 and K_R provides the probability of failure. As service time advances the LSS moves and probability of failure changes. For complex systems where the model is implicit, the metamodel approach, using design of experiments, lets us write an explicit function of the response - similar to that in the example.

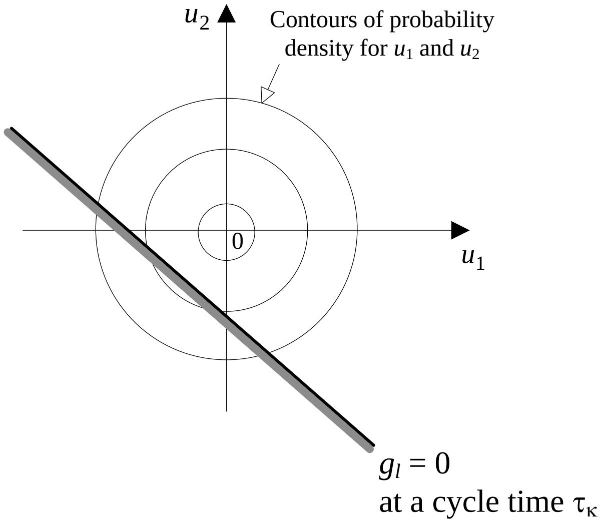
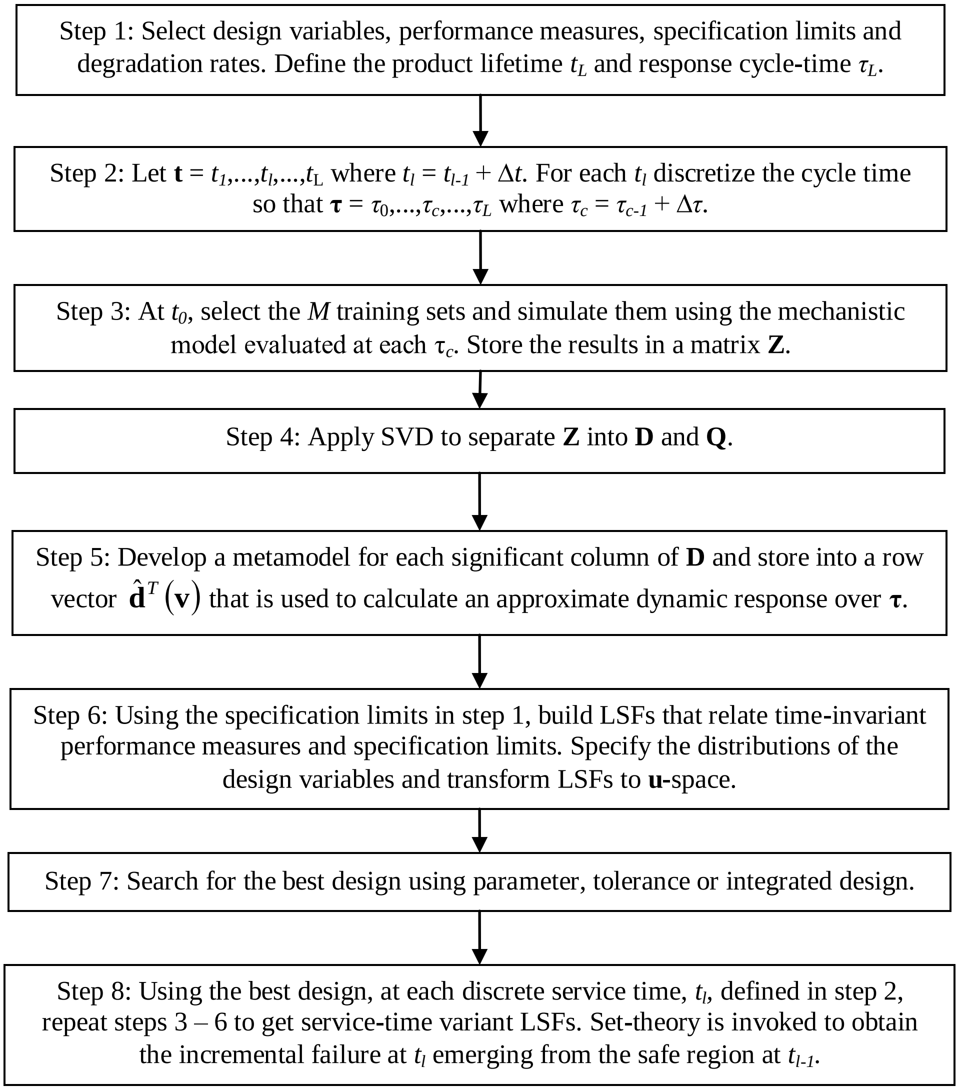

u_2
Contours of probability density for u_1 and u_2
0
u_1
$g_l = 0$
at a cycle time τ_κ

Figure 1-2: Limit-state surface with failure side shaded

1.4 Outline

Chapter 2 presents an overview of the previous work done with metamodels, reliability-based analysis and optimization. Chapter 3 provides a discussion of the Singular Value Decomposition method and how it is used to develop a metamodel to represent the time-invariant performance measure. The mathematical background of three metamodels, the Regression Model, Kriging and the Radial Basis Function that are used is also presented. This chapter also discusses how the accuracy of the metamodels is checked. Chapter 4 discusses how the First Order Reliability Method and the Monte Carlo Simulation are used to estimate the probability of failure and the three techniques involved to optimize the parameters of the design variables. Chapters 5 and 6 presents examples to show probability-based design optimization of static and dynamics systems using metamodels. Chapter 7 presents an overview of the set-theory method that is used to estimate the Cumulative Distribution Function when degradation of the design variables is assumed and Chapter 8 presents a case study to illustrate how the Cumulative Distribution Function of dynamic degrading systems is plotted. Finally, Chapter 9 presents a summary of the thesis and possible future directions. Figure 1-3 presents a flowchart of the main steps in the methodology.



Step 1: Select design variables, performance measures, specification limits and degradation rates. Define the product lifetime t_L and response cycle-time τ_L.

Step 2: Let $\mathbf{t} = t_1,...,t_l,...,t_L$ where $t_l = t_{l-1} + \Delta t$. For each t_l discretize the cycle time so that $\boldsymbol{\tau} = \tau_0,...,\tau_c,...,\tau_L$ where $\tau_c = \tau_{c-1} + \Delta\tau$.

Step 3: At t_0, select the M training sets and simulate them using the mechanistic model evaluated at each τ_c. Store the results in a matrix **Z**.

Step 4: Apply SVD to separate **Z** into **D** and **Q**.

Step 5: Develop a metamodel for each significant column of **D** and store into a row vector $\hat{\mathbf{d}}^T(\mathbf{v})$ that is used to calculate an approximate dynamic response over $\boldsymbol{\tau}$.

Step 6: Using the specification limits in step 1, build LSFs that relate time-invariant performance measures and specification limits. Specify the distributions of the design variables and transform LSFs to **u**-space.

Step 7: Search for the best design using parameter, tolerance or integrated design.

Step 8: Using the best design, at each discrete service time, t_l, defined in step 2, repeat steps 3 – 6 to get service-time variant LSFs. Set-theory is invoked to obtain the incremental failure at t_l emerging from the safe region at t_{l-1}.

Figure 1-3: The sequence of steps in the proposed methodology.

Chapter 2

Literature Review

Design optimization typically involves determining the most desirable design under various conditions with deterministic optimization being the most popular in engineering design (Ju and Lee 2008). Deterministic optimization searches for the best setting of the design variables that result in some function of interest being optimized. Various techniques for deterministic optimization of static systems involve gradient-based methods to search for local optima or genetic algorithms for global optimization. When uncertainty is considered, designers typically use Taguchi methods or reliability-based design optimization (Chiralaksanakul and Mahadevan 2005);. For the deterministic optimization of dynamic systems, particle swarm optimization, used by (Eberhart and Shi 2001), is an evolutionary computation technique similar to a genetic algorithm where the system is initialized with a population of candidate solutions.

Some authors have explored various methods for the optimization of dynamic systems when uncertainty is considered. Chen and Wu 2004 presented an interval optimization method for the dynamic response of structures with interval parameters. In their paper, in order to account for the error or uncertainties present in some structural parameters caused by manufacture, installation or measurement, the errors from the aforementioned uncertainties were predicted and denoted by intervals. For parameter design, Chang 2006, developed an optimization approach for dynamic multi-response systems based upon backpropagation neural networks, BPN, and desirability functions to the parameter design of the dynamic multi-response. The BPN is the metamodel and is used to predict all possible multi-responses of the system by inputting all combinations of the parameters. An alternative parameter measurement was then developed for dynamic multi-response to measure and integrate the multi-responses into a single desired value by introducing desirability functions. The best parameter setting was then obtained by maximizing the desired value.

In order to simplify the mechanistic model, (Han, Ruduyi and Korvink 2005) focussed on the application of model order reduction techniques so as to reduce the transient analysis time for the optimization process. They presented open-source software, known as “mor4ansys”, to perform model order reduction directly from ANSYS models (engineering simulation software) to generate a reduced model of a second-order linear system. They were able to integrate this method into an optimization process to optimize MEMS system rapidly. As illustrated through a structural dynamics problem expressed in the second-order ODEs, the use of reduced-order models within the optimization cycles produces almost the same optimization results and reduces the total computational costs by about an order of magnitude.

There has also been an attempt to utilize metamodels with robust optimization tools for design. A paper by Kim et. al, 2007 presents a metamodel-based design optimization for dynamic response optimization which avoids design sensitivity analysis and overcome the numerical noise. The authors utilized their method for the Design for Six Sigma (DFSS) optimization of the dynamic response of a paper feeding mechanism. Their design objective was to minimize the slip amounts between paper and mechanisms and satisfy the 6-sigma constraint for the nip forces of rollers. Through the use of the gradient information of metamodels, the authors have found that DFSS and robust optimization is easily implemented.

The main problem of most dynamic systems is that the mechanistic model can become very complex. In such cases, simplification is required in either the model or the optimization method. This thesis proposes to simplify the mechanistic model using a metamodel and using this approximate function, fast integration methods can be easily applied to quickly search for a design that reduces failure probability. The following sections will discuss previous research done with metamodels and reliability methods.

2.1 Metamodels

Metamodels, or approximating functions, are most often used in analyses where the behaviour of a physical system is being investigated. Metamodels are typically used for response prediction and curve fitting (Barton, 1998). However, more recently, metamodels are gaining attention where they replace mechanistic models to allow for more efficient and simpler design.

Most research pertaining to the design of a physical system involves building a mechanistic model of the physical system and then using computer software to perform simulations (Barton 1998); (Kleijnen and Sargent 2000). Conceptually, if the inputs to a computer simulation are supplied in vector **X**, and the outputs from the analysis in vector **z**, then the mathematical model evaluates z = $f(\mathbf{v})$ where $f(\mathbf{v})$ is a complex engineering analysis function such as finite element analysis or Lagrange equations. The metamodel enables the real function $f(\mathbf{v})$ to be approximated by a simpler function $\hat{f}(\mathbf{v})$ where $f(\mathbf{v}) = \hat{f}(\mathbf{v}) + \varepsilon$ and ε is the error term (Clarke, Griebsch and Simpson 2005).

There exist a number of popular metamodels such as response surface models based on regression methods, neural networks or Kriging models. Among these, the most popular are regression based response surface models. The simplicity of this method is a result of the simplicity in estimating the model parameters compared with the other three metamodels. Although simple to use, the ability of these models to estimate highly nonlinear responses is quite poor. Neural networks and Kriging models are the alternative option which offer a better fit of highly nonlinear responses (Clarke, Griebsch and Simpson 2005). The major downfall with these methods is that estimating model parameters requires more effort than the Regression Model.

In order to fit the metamodel, an experimental design (termed hereafter as the "training design") must be selected. For this, various design of experiments techniques are considered. Statistical design of experiments refers to the process of planning the experiment so that appropriate data that can be analyzed by statistical methods will be collected, resulting in valid and objective conclusions. Many experiments involve the study of the effects of two or more factors/variables. Factorial designs, which are the most efficient for this type of experiment, involve investigating all possible combinations of the levels of the factors in each complete trial or run of the experiment (Montgomery 2005). For example, if there are *a* levels of factor A and *b* levels of factor B, the experiment has *ab* treatment combinations. Factorial designs have several advantages; they are more efficient than one-factor-at-a-time experiments and they are necessary when interactions may be present to avoid misleading conclusions. Finally, factorial designs allow the effects of a factor to be estimated at several levels of the other factors, yielding conclusions that are valid over a range of experimental conditions. Other popular techniques include the central-composite design, Box-Behnken design or D-optimal designs (Montgomery 2005).

Sometimes, three levels are used instead of two so that curvature in the system can be more accurately captured. A three-level factorial design consists of *k* factors each at three levels. This is denoted as a 3^k design. The three levels are referred to as low, intermediate and high. The major disadvantage of factorial designs is that a full design can get big quickly, especially 3^k designs. For example, 3^2 design = 9 runs, 3^3 design = 27 runs, 3^4 design = 81 runs and so on. In many instances, the experimenter may not have the resources available to run a full factorial design. Therefore, fractional factorial designs are used.

Since the 3^k factorial design can get very large quickly, a fractional replicate of the full design is usually performed. Fractional 3^k factorial designs are not normally performed. Instead a Box-Behnken, Central Composite or D-optimal designs can be used (Montgomery 2005). Other methods that may be used are sampling methods such as the Latin-Hypercube Sampling method or space-filling designs.

The second step of the metamodelling process involves selecting a metamodel and estimating the model parameters. Some popular metamodels include response surface models, Radial Basis Functions and Kriging (Hussain, Barton and Joshi 2002). All of these techniques are capable of function approximation, but they vary in their accuracy, robustness and computational efficiency. Much research has compared the performance of the different metamodel on approximating deterministic computer models (Hussain, Barton and Joshi 2002); (Simpson, Peplinski, et al. 2001); (Jin, Chen and Simpson 2001).

Hussain, Barton and Joshi, 2002, used the Radial Basis Function on some known test functions. The performance of the RBF as a metamodel was tested and compared to the popular Regression-Based Response Surface Models using the same set of input points selected through the factorial and the Latin Hypercube design. Seven test functions were used and the performance of each metamodel using both training designs was compared. The authors found that the RBF metamodel provided a better fit than the RM for all seven cases studied.

Jin, Chen and Simpson, 2001, provided a detailed comparison of four popular Metamodelling techniques, RM, multivariate adaptive regression splines (MARS), RBFs and Kriging, based on multiple performance criteria using fourteen test problems representing different classes of problem. The fourteen test problems were classified based on the following representative features of engineering design problems: Problem Scale (two scales: no. of variables ≥10, no. of variables = 2,3), Nonlinearity of the performance behaviour and “noisy” versus “smooth” behaviour. The performance of each metamodel was measured using the following aspects

- Accuracy: Capability of predicting the system response over the design space of interest.
- Robustness: Capability of achieving good accuracy for different problems. Indicates whether a modelling technique is highly problem dependent.
- Efficiency: Computational effort required for constructing the metamodel and for predicting the response for a set of new points by metamodels.
- Transparency: Capability of providing the information concerning contributions of different variables and interactions among variables.
- Conceptual Simplicity: Ease of implementation. Simple methods should require less user input and be easily adapted to each problem.

Based on the above aspects, the authors found that in terms of accuracy and robustness, when large sample sizes are used, MARS, Kriging and RBF perform equally well. When noise in the data is considered, the RM performs the best and RBF works well; however, Kriging was found to be quite sensitive to the noise because it interpolates the sample data. When efficiency is considered, Kriging can be very time-consuming especially for large-scale problems with large sample sizes and RM takes the least amount of time for model building. In terms of accuracy and robustness, for small and scarce sample sets, RBF performs the best. The RM has good transparency. This feature makes it very helpful to reduce the scale of a problem by removing insignificant factors; neither Kriging nor RBF have such transparency. Finally, the RM and RBF were found to be the easiest to implement while estimating the parameters for Kriging was found to be difficult.

There has been one attempt to apply metamodels for response prediction of a dynamic response. Lee et. al, 2007 attempted to predict the motion of a tracked vehicle travelling on soft soil by using the Kriging metamodel. Their paper has found that Kriging has been better able to predict the dynamic response than the Regression Model. Kriging was found to accurately approximate the nonlinear response of the vehicle.

Along with fitting the metamodel, validating the metamodel is important to show how well it fits the data. Also important is the method used to validate the metamodel. Since simulations are assumed to be time consuming, a method must be chosen such that additional simulation runs are not required. (Meckesheimer, et al. 2002), investigated validation strategies for assessing the metamodel fidelity of deterministic computer simulation codes without the use of additional expensive computer simulations. In their work, the authors investigated the leave-k-out cross-validation strategy for metamodel assessment. This method was applied to two test problems that were each fit using Kriging, a low-order polynomial and the Radial Basis Function using different methods for selecting design points.

In terms of practicality, the leave-k-out cross-validation strategy provides a reasonable indicator of metamodel fidelity without the use of additional computationally expensive analyses. The method involves dividing the training design into a set of N_f groups, known in cross-validation applications as “folds”. One fold is withheld while the others are used to fit the metamodel. The response at each design variable set of the withheld fold is estimated using the newly fit metamodel. This process is repeated until all folds are used in fitting the metamodel.

Since the metamodels are fit N_f times during each cross-validation cycle, the numerical efficiency of the leave-k-out cross-validation strategy depends on the type of metamodel. Where accuracy and precision of estimating an error measure is concerned, their results show that as more points are omitted during the metamodel fit, the $\mathrm{AVE}_{\mathrm{RMSE}_{\mathrm{CV}}}$ [1] estimate increases when using LOP and RBF metamodels. Based on the observations from the experimental study conducted to assess the leave-k-out cross-validation strategy, a value of k = 1 is recommended for providing a prediction error estimate that was within ± 25% of the true prediction error. Choosing k as a function of the fitting design (that is, k = 0.1N or $k = \sqrt{N}$) is recommended for estimating the prediction error for Kriging metamodels (Meckesheimer, et al. 2002); (Kleijnen and Sargent, 2000) and (Martin and Simpson, 2003).

Metamodels have been previously used in reliability analysis. Deng, 2006 observed the feasibility of using three Radial Basis Function networks in reliability analysis methods. According to the author, the Regression Model becomes impractical for problems involving a large number of nonlinear random variables or in cases where mixed or statistically dependent random variables are involved.

2.2 Reliability-Based Design Optimization

Traditional deterministic design optimization has been successfully applied, in engineering design to reduce the cost and improve quality. Previously, iterative optimization methods have often been used in conjunction with engineering simulation models to search for designs with desired properties. Such methods have been time consuming due to expensive run times of the simulation model caused by underlying complex mechanistic models. Metamodels have been used many times in response prediction applications. Due to their popularity and feasibility for these applications, authors have started using metamodels for optimization and reliability analysis (Barton and Meckesheimer, 2006).

Due to uncertainties present in engineering simulation or manufacturing process, authors have used reliability-based design optimization models for robust and cost-effective designs (Tu, Choi and Park 1999); (Allen, et al. 2004). Reliability-based design optimization, denoted as RBDO, of an engineering system deals with optimizing a prescribed performance function while ensuring that the system reliability is within an acceptable limit (Chiralaksanakul and Mahadevan 2005).

[1] Average Root Mean Squared Cross-Validation

In the RBDO model for robust system parameter design, the mean values of the design variables and the cost are optimized subject to prescribed probability constraints by solving a mathematical nonlinear programming problem. Deng, 2006 has shown how to employ multi-layer perceptron (MLP) techniques to approximate implicit performance functions and derivates for the first- and second-order reliability methods.

Grandhi and Wang, 1998 used reliability methods for structural optimization in a multidisciplinary environment. Probabilistic structural analysis is a computationally intensive procedure and it requires multiple, sometimes hundreds of deterministic analyses. Thus, there is strong motivation to develop efficient techniques for reducing computer time. Allen et. Al., 2004 studied the feasibility, potential and limitations of using electromechanical simulation for the reliability-analysis and design optimization of Micro-Electrical Mechanical Systems (MEMS). In their paper, they stated that the stochastic nature of engineering systems have been compensated for through the use of safety factors. They also state that such approaches were found to lead to either a conservative design or over-compensating for uncertainties. The authors, therefore, used FORM, to deal with uncertainty, combined with electromechanical modeling to develop a reliability-based design optimization framework for the design of electrostatically actuated MEMS devices.

In this thesis, structural reliability concepts are used to search for the best design. Structural reliability analysis deals with the statistical nature of many basic variables in structural safety and design (Deng 2006). Therefore, when presented with a physical system, it is assumed that the design variables are modeled using a mean and standard deviation.

The Monte-Carlo simulation (MCS), the first-order reliability (FORM) and the second-order reliability methods (SORM) are the three methods that have been widely used to estimate the failure probability of structural systems (Deng 2006). The MCS requires the calculations of hundreds of thousands of performance function values. In the first-order reliability method, an approximation to the probability of failure is obtained by linearizing the limit-state surface (the boundary of the failure domain) at the “most likely failure point”. This is the point on the limit-state surface that is nearest to the origin in a standard normal space obtained by a suitable transformation of the random variables. Due to the rotational symmetry and exponential decay of the probability density in the standard normal space, the design point has the highest density among all points in the failure domain. It follows that the neighbourhood of this point makes the dominant contribution to the failure probability integral. This property is the basis for FORM which in effect constructs an approximation to the failure probability integral by using the tangent plane at the design point as the integration boundary (Der Kiureghain and Dakessian 1998).

The FORM and SORM generally demand the values and partial derivatives of the performance function with respect to the design random variables. Such calculations can be performed efficiently when the performance function g(**v**) can be expressed as an explicit form or simply analytical form in terms of the basic variables **v**. When the performance functions are implicit, such calculations require additional effort and will be time-consuming.

Kaymaz, 2005 investigated the use of the Kriging method for structural reliability problems by comparing it with the commonly used RM. Some advantage and disadvantages of Kriging is reported. This paper used a two-stage approach to find the final design point. In the first stage, the design point has been found using either FORM or MCS. Using this design point, a new point is obtained using a method proposed by Bucher and Bourgund, 1990. This two-stage approach has been used in three numerical examples to compare the classical RM with the Kriging method. The first two examples have an explicit limit state function with two random variables whilst the third involves an implicit LSF. Among these three examples, Kriging was able to produce better results than the RM. The author also observed the effect of the choice of the Kriging parameters, theta and the value of the reliability index. It has been found that the choice of theta does affect the value of the calculated reliability index. The choice of correlation function in the Kriging model is also important. For problems having a nonlinear LSF, the Gaussian correlation function is a much better choice than either the linear or exponential functions.

2.3 Degradation Modelling

Degradation of the components of systems is inevitable due to wear or aging. In order to ensure the system is acceptable over the product lifetime, the times and frequencies of inspection, maintenance and replacement are required (Savage and Son, 2009). However, due to uncertainty in the degradation rates of the components, determining service times is a complicated process.

Traditionally, the uncertainty in aging, deterioration and degradation is modelled using lifetime distributions (Savage and Son 2011). Instead of using lifetime distribution models, the degradation of the characteristic of the actual components is modelled using any one of three models; (a) random variable models, (b) marginal distribution models and (c) cumulative damage models (Savage and Son 2011).

Modelling degradation data has been used to infer 1) failure distribution, 2) system reliability from inferred component reliabilities and 3) the system reliability from component degradation data using analytical system response. Previously, the path-tracing approach, a sample-based method that uses Monte Carlo Simulation, has been used to model degradation. Sampling approaches take a sample of the variable assumed to degrade, say R_0, from its *pdf* at time zero and traces it path over time using a particular degradation function denoted as K. A conceptually simpler and more efficient approach has been proposed to solve the time-history problem called the set-theory method (Savage and Son 2006); (Savage and Son 2011).

The set-theory method has several advantages over the path-tracing approach. First, no explicit lifetime is required. Second, the conceptual simplicity of the method comes from including the random degradation rate in the joint *pdf* of the design variables, **v**. Thirdly, the probabilities are found, contiguously, at successive time increments and finally, for explicit limit-state functions, the method is especially computationally efficient since vector arithmetic may be used to evaluate the signs of a vector of limit-state functions. The set-theory method will be discussed in more detail in Chapter 7.

In general, there are three types of reliability degradation models: degradation path models, marginal distribution models and cumulative damage models (Huang and Dietrich 2005); (Savage and Son 2011). Degradation modelling based upon degradation path models assumes that each component degrades in the same way under fixed environmental conditions. Therefore, each degradation path has the same functional form. Random coefficients are introduces to describe variations in the path model and the variations are due to either manufacturing process or environmental conditions (Lawless 1982). Statistical distribution parameters of random coefficients are numerically estimated using observed degradation data. The simplest form of a degradation path model of, for example, resistance R is

$$R = R_0 \pm Kt \tag{2.1}$$

Where R_0 is the initial resistance and K is the degradation rate.

The second type of model, the “marginal distribution model”, characterizes degradation data using a change of distribution parameters versus time. The two-step statistical analysis, that includes (1) estimating the distribution parameters at each observed time and (2) fitting time-dependent distribution parameter functions, is carried out to model the degradation data. Degradation data are, therefore, modelled as time-variant distribution parameters that are functions of initial distribution parameters and time. For example, since the distribution parameters are now functions of the service time t

$$R = R_0\left[\mathbf{p}(t)\right] \tag{2.2}$$

Where $\mathbf{p}(t)$ denotes the service-time variant parameters.

The third type of model, the cumulative damage (CD) model, assumes that the degradation is caused by shocks and that damage accumulated additively (Bogdanoff and Kozin 1985); (Finkelstein and Cha 2010). These models are used when temporal uncertainty associated with the deterioration cannot be ignored. Examples that use this type of model include bridge deck degradation and storm water pipe deterioration.

Chapter 3

Metamodels

3.1 Introduction

Metamodels has been commonly used in function approximation where the analyst replaces a more complex mathematical model with the metamodel in order to estimate the response at some untried inputs. In the literature, many types of metamodels exist with the most common being the Regression Model (RM). Although simple to use, this metamodel usually is not as accurate in response prediction as some of the more complex metamodels such as Kriging or neural networks. Building a metamodel normally involves three main steps; selecting the training design, estimating metamodel parameters and then checking the fit of the metamodel.

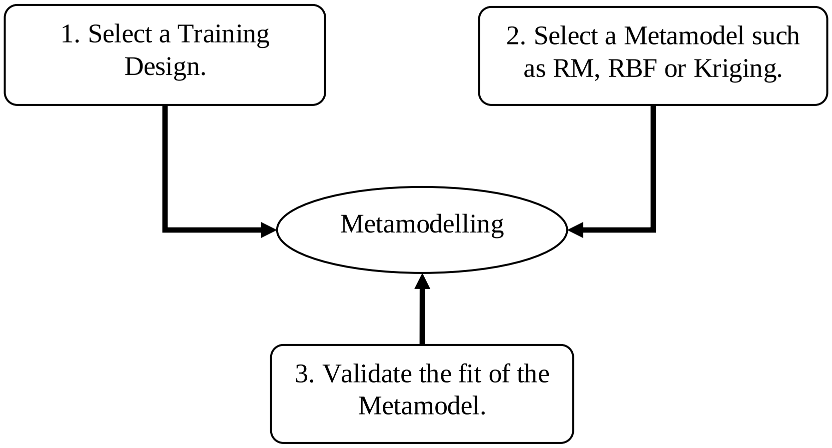

1. Select a Training Design.
2. Select a Metamodel such as RM, RBF or Kriging.
Metamodelling
3. Validate the fit of the Metamodel.

Figure 3-1: The three main steps involved in the metamodelling process.

In this thesis, three popular metamodels are investigated; the RM, Kriging and the Radial Basis Function (RBF). This chapter will present the general form of these metamodels and show how the metamodel parameters are estimated. This chapter will also show how, using Singular Value Decomposition, SVD, time-specific metamodels are derived and then used with the $\mathbf{Q}^{\mathrm{T}}$ matrix in SVD to develop a time-invariant metamodel. The metamodel is put into matrix form by collecting and grouping the constant and variable terms. This is done to allow for efficient differentiating that is required for the application of FORM.

3.2 Training Design

We assume we have a mechanistic model that relates responses to design variables for which we have their initial or nominal values. The number of prospective design variables is usually large but may be reduced by a sensitivity-based importance analysis so only η variables that provide the greatest perturbations are retained. Rejected variables are considered deterministic. Deterministic variables are set to their nominal values. In order to fit the metamodel, a sample set of design variables and responses, called the "training design" is selected. The training design is chosen through the use of design of experiments techniques, sampling methods or space-filling designs and assists with estimating metamodel parameters. The most common design of experiments technique is the factorial experiment where the controllable factors are varied together instead of one at a time (Montgomery 2005). Other methods that can be used to develop the training design are sampling methods such as Latin Hypercube Sampling, D-optimal designs, Box-Behnken Designs or Central Composite Designs (Montgomery 2005).

Using either a full or fractional factorial design, a sample of design variables is obtained which we call the "training sets". In order to set up the training design, the number of levels, λ, is assigned to each variable. Following a full factorial design formulation, the total number of training sets $M = \lambda^{\eta}$ where η represents the total number of design variables. Other design of experiments, sampling methods or space-filling designs can be used to select the training sets. The choice of method depends upon the number of experiments that can be performed or the data available. The M training sets are recorded into a matrix $\mathbf{X}$ as shown in (3.1) where x_{ij} corresponds to the i^{th} training set of the j^{th} design variable ($i = 1,\ldots,M$ and $j = 1,\ldots,\eta$).

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_M^T \end{bmatrix} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,\eta} \\ x_{2,1} & \cdots & x_{2,\eta} \\ \vdots & \ddots & \vdots \\ x_{M,1} & \cdots & x_{M,\eta} \end{bmatrix}_{M\times\eta} \tag{3.1}$$

3.3 General Metamodel

In the mechanistic modelling process, the modeller abstracts properties of various components and their interactions and formulates these properties into a set of equations. In most cases, these models are highly non-linear, or even implicit, thus making them difficult to use in design calculations or just too time-consuming. In an alternative way, using design of experiments and fitting functions, a model of the mechanistic model, called a metamodel is used to approximate the normally complex or implicit mechanistic models (Barton, 1998); (Barton amd Meckesheimer, 2006). The most common metamodels available are the Regression Models, Radial Basis Functions and the more recent Kriging models. The details are presented later. Now let us develop a matrix-vector form of the metamodels that are advantageous for our purposes.

Static systems

Let us consider the static model in (1.1) and from M experiments place the corresponding responses in a column **z**.

$$\mathbf{z}(\mathbf{x}) = \begin{bmatrix} \mathbf{z}(\mathbf{x}_1^T) \\ \mathbf{z}(\mathbf{x}_2^T) \\ \vdots \\ \mathbf{z}(\mathbf{x}_M^T) \end{bmatrix}_{M\times 1} \tag{3.2}$$

For our purposes, let a metamodel have the universal form

$$\hat{z}(\mathbf{v}) = \mathbf{w}^T \mathbf{r}(\mathbf{v}) \tag{3.3}$$

The vector $\mathbf{w}$ contains the so-called fitting parameters that are functions of the data (i.e. $\mathbf{X}, \mathbf{z}$). The vector $\mathbf{r}(\mathbf{v})$ depends on the particular metamodel, and the vector $\mathbf{v}$ represents the design variables. Both vectors in (3.3) have length L. Note eq. (3.3) provides an explicit, approximate, response for any set of design variables.

Dynamic systems

Next, let us expand the above to a dynamic problem as outlined in (1.2). To build on static system ideas, it is convenient to capture the response at only selected time increments $\Delta\tau$. For a dynamic lifetime τ_L , the continuous response is broken into a set of static responses at individual time steps and recorded into a matrix, **Z.** Let the number of discrete times be

$$C = \frac{\tau_L}{\Delta\tau} + 1 \tag{3.4}$$

where τ_L represent the cycle lifetime of the dynamic response and $\Delta\tau$ is the time increment size. The vector of all discrete cycle times is

$$\boldsymbol{\tau} = \begin{bmatrix} \tau_1 & \cdots & \tau_k & \cdots & \tau_C \end{bmatrix}^T$$

Computer simulations provide the cycle-time responses for all of the M training sets. The response magnitudes at $\boldsymbol{\tau}$ for each experiment are recorded in the matrix **Z**

$$\mathbf{Z} = \begin{bmatrix} \mathbf{z}^T(\mathbf{x}_1, \boldsymbol{\tau}) \\ \mathbf{z}^T(\mathbf{x}_2, \boldsymbol{\tau}) \\ \vdots \\ \mathbf{z}^T(\mathbf{x}_M, \boldsymbol{\tau}) \end{bmatrix} = \begin{bmatrix} z(\mathbf{x}_1, \tau_1) & z(\mathbf{x}_1, \tau_2) & \ldots & z(\mathbf{x}_1, \tau_C) \\ z(\mathbf{x}_2, \tau_1) & z(\mathbf{x}_2, \tau_2) & \ldots & z(\mathbf{x}_2, \tau_C) \\ \vdots & \vdots & \ddots & \vdots \\ z(\mathbf{x}_M, \tau_1) & z(\mathbf{x}_M, \tau_2) & \ldots & z(\mathbf{x}_M, \tau_C) \end{bmatrix}_{M \times C} \tag{3.5}$$

The rows of **Z** represent the dynamic response at various design variable combinations and the columns represent the discrete time step. There are a total of M design variable combinations and C discrete time steps.

When the system is dynamic, one major question that must be answered is how to determine the best value for $\Delta\tau$. Consider the case where some performance measure such as integral squared error or energy must be estimated using a dynamic response. First, an initial value for $\Delta\tau$ is assumed, the dynamic response is recorded at each discrete time increment and the performance measure is estimated. The time increment, $\Delta\tau$, is then reduced and the performance measure is estimated again. When an asymptotic value for the performance measure is reached then an acceptable value for $\Delta\tau$ has been obtained.

In order to estimate a performance measure based upon the dynamic response such as the integral squared error or energy, a metamodel can be built for each column in (3.5). However, a more efficient approach exists using Singular Value Decomposition.

3.3.1 Singular Value Decomposition

Singular Value Decomposition, denoted as SVD, can be used in cases where the dynamic response of a system is required to estimate some time-invariant performance measure such as estimating the energy consumed using the dynamic angular velocity. In order to do this, the feature-space extraction and principal component analysis (PCA) characteristics of SVD is used. A paper by Hassanpour, Mesbah and Boashash, 2004, shows how SVD was used to extract EEG feature signals of newborn babies to detect seizure activity. In another paper by Wall, Rechtsteiner and Rocha, 2007, SVD was used to analyze gene expression data for visualization of gene expression data, representation of the data using a smaller number of variables and detection of patterns in noisy gene expression data. These two characteristics can be combined to build a 'time-dependent metamodel' of the nonlinear dynamic response of a vehicle to replace the high computational effort of using SIMULINK in optimization (Wehrwein and Mourelatos 2008).

In the paper by Wehrwein and Mourelatos, 2008, the dynamic response was recorded into a matrix and SVD was used to separate the two features spaces. After which, the significant singular values were used to reduce the matrix containing design variable information. A metamodel was then built for each column of the matrix containing the design variable information. Ultimately, the authors found their 'time-dependent metamodel' to be efficient and accurate. The continuous dynamic response is broken into a set of static responses at discrete time steps. This information is recorded into an $M \times C$ matrix, **Z,** shown in (3.5). In this matrix, **Z**, $z(\boldsymbol{x}_i, \tau_j)$ is the response of the i^{th} design variable set at the j^{th} time step.

Elements of the i^{th} row of **Z** form the C-dimensional vector $\boldsymbol{z}(\boldsymbol{x_i}, \boldsymbol{\tau})$ which is referred to as the dynamic response of the i^{th} design variable set. Elements of the j^{th} column form the M-dimensional vector $\boldsymbol{z}(\boldsymbol{X}, \tau_j)$ which is referred to as the response profile of the j^{th} time step at the various design variable sets.

The equation for SVD of **Z** is

$$\mathbf{Z}_{M\times C} = \mathbf{U}_{M\times C}\mathbf{S}_{C\times C}\mathbf{Q}_{C\times C}^{T} \tag{3.6}$$

where $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}$ and $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}$; therefore **U** and **Q** are orthogonal. Columns of **U** are called the left eigenvectors, $\{u_k\}$, and form an orthonormal basis in the M-dimensional space of design variable combinations. Rows of $\mathbf{Q}^{\mathrm{T}}$ contain the elements of the right eigenvectors,$\{q_i\}$, and form an orthonormal basis in the *C*-dimensional space of time instances (Wehrwein and Mourelatos 2008); (Hassanpour, Mesbah and Boashash 2004).

Calculating the SVD consists of finding the eigenvalues and eigenvectors of $\mathbf{ZZ^T}$ and $\mathbf{Z^TZ}$. The eigenvectors of $\mathbf{Z^TZ}$ make up the columns of **Q**, the eigenvectors of $\mathbf{ZZ^T}$ make up the columns of **U**. The elements of **S** are only nonzero on the diagonal and are called the singular values. The singular values of **S** are square roots of eigenvalues from $\mathbf{ZZ^T}$ or $\mathbf{Z^TZ}$. The singular values are ordered from high-to-low with the highest singular value in the upper left index of the **S** matrix. By setting the small (non-dominant) singular values to zero, we can obtain matrix approximations whose rank equals the number of remaining singular values.

Since the columns of **U** represent the information of **Z** in the space of design variable combinations, a matrix **P** can be formulated that contains the columns of **U** corresponding to the dominant singular values of **S**. If **P** is multiplied by the dominant singular values of **S**, a matrix containing the dominant design variable information of **Z** is obtained. Therefore

$$\mathbf{D}_{M\times s} = \mathbf{P}_{M\times s}\mathbf{S}_{s\times s} \tag{3.7}$$

Where *s* is the number of dominant singular values of **S**.

Now, another question that must be answered, similar to the previous step where *C* must be determined, is what are the dominant singular values of **S**? This question is answered in a similar way as the first. Using the first column of **S,** and estimate of the matrix of training responses, $\hat{\mathbf{Z}}$, is computed and the performance measure is estimated. These estimates are compared with the “true” values found using the original matrix **Z**. The comparison is done by simply finding the error between $\hat{\boldsymbol{Z}}$ and **Z**. If the error is not small enough, another column of **S** is added until the error is negligible. The total number of columns of **S**, denoted as *s*, that results in an acceptable error is retained and *s* metamodels are built where *s* << C.

Herein, metamodels are built only for the columns of **D** and for h^{th} column (i.e. $\mathbf{d}_h$) the metamodel is denoted as $\hat{\mathbf{d}}_h(\mathbf{v})$. After fitting a metamodel for each significant column, the *s* metamodels are assembled into a vector

$$\hat{\mathbf{d}}^T(\mathbf{v}) = \left[\hat{d}_1(\mathbf{v}) \quad \hat{d}_2(\mathbf{v}) \quad \ldots \quad \hat{d}_s(\mathbf{v})\right] \tag{3.8}$$

When an estimate of the dynamic response at **v** is required, then (3.8) pre-multiplies the time-feature matrix to give

$$\hat{\mathbf{z}}^T(\mathbf{v},\boldsymbol{\tau}) = \left[\hat{\mathbf{d}}(\mathbf{v})\right]^T_{1\times s}\left[\mathbf{Q}^T\right]_{s\times C} \tag{3.9}$$

Every metamodel comprises constants, the “fitting parameters”, and functions of the design variables **v**. These constants and functions may be grouped into two vectors; one containing only constants and the other only vectors and can be written in a universal form as

$$\hat{d}_h(\mathbf{v}) = \mathbf{w}_h^T\mathbf{r}_h(\mathbf{v}) \qquad h = 1,2..s \tag{3.10}$$

For each of the s columns in **D**, the vector $\mathbf{w}_h$ contains the fitting parameters that are functions of the training sets and responses (i.e. **X**, $\mathbf{d}_h$). The vector $\mathbf{r}_h(\mathbf{v})$ depends upon the particular metamodel (regression, Kriging or RBF) and the vector **v** represents the design variables. Using this general metamodel form, (3.9) becomes

$$\hat{\mathbf{d}}^T(\mathbf{v}) = \left[\mathbf{w}_1^T\mathbf{r}_1 \quad \mathbf{w}_2^T\mathbf{r}_2 \quad \dots \quad \mathbf{w}_s^T\mathbf{r}_s\right]_{1\times s} \tag{3.11}$$

Again, grouping and separating constants and variables, (3.11) becomes the column vector

$$\hat{\mathbf{d}}(\mathbf{v}) = \begin{bmatrix} \mathbf{w}_1^T & 0 & \dots & 0 \\ 0 & \mathbf{w}_2^T & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathbf{w}_s^T \end{bmatrix}_{s\times(s\times L)} \begin{bmatrix} \mathbf{r}_1(\mathbf{v}) \\ \mathbf{r}_2(\mathbf{v}) \\ \vdots \\ \mathbf{r}_s(\mathbf{v}) \end{bmatrix}_{(s\times L)\times 1} \tag{3.12}$$

Now, using (3.9) and (3.11), the dynamic response of **v** is written as a column vector

$$\left[\hat{\mathbf{z}}(\mathbf{v},\boldsymbol{\tau})\right]_{C\times 1} = \left[\mathbf{Q}\right]_{C\times s} \begin{bmatrix} \mathbf{w}_1^T & 0 & \dots & 0 \\ 0 & \mathbf{w}_2^T & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathbf{w}_s^T \end{bmatrix}_{s\times(s\times L)} \begin{bmatrix} \mathbf{r}_1(\mathbf{v}) \\ \mathbf{r}_2(\mathbf{v}) \\ \vdots \\ \mathbf{r}_s(\mathbf{v}) \end{bmatrix}_{(s\times L)\times 1} \tag{3.13}$$

Stacking $\mathbf{r}_1(\mathbf{v}), \mathbf{r}_2(\mathbf{v})\cdots$to form$\,\bar{\mathbf{r}}(\mathbf{v})$, a simpler notation for the dynamic response at some unknown **v,** is

$$\hat{\mathbf{z}}(\mathbf{v},\boldsymbol{\tau}) = \mathbf{Q}\,\mathbf{W}\bar{\mathbf{r}}(\mathbf{v}) \tag{3.14}$$

In some cases, like the Regression Model, **r**(**v**) never changes for each column of **D**. Therefore, if $\mathbf{r}_1(\mathbf{v}) = \mathbf{r}_2(\mathbf{v}) = ...= \mathbf{r}_h(\mathbf{v}) =... = \mathbf{r}_s(\mathbf{v})$, then (3.13) becomes

$$[\hat{\mathbf{z}}(\mathbf{v},\boldsymbol{\tau})]_{C\times 1} = [\mathbf{Q}]_{C\times s}\begin{bmatrix} \mathbf{w}_1^T & 0 & \dots & 0 \\ 0 & \mathbf{w}_2^T & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathbf{w}_s^T \end{bmatrix}\begin{bmatrix} \mathbf{I}_L \\ \mathbf{I}_L \\ \vdots \\ \mathbf{I}_L \end{bmatrix}_{s\times ::} \quad [\mathbf{r}(\mathbf{v})]_{L\times 1} \tag{3.15}$$

Or succinctly

$$\hat{\mathbf{z}}(\mathbf{v},\boldsymbol{\tau}) = \mathbf{Q}\left(\mathbf{W}\bar{\mathbf{I}}\right)\mathbf{r}(\mathbf{v}) = \mathbf{Tr}\left(\mathbf{v}\right) \tag{3.16}$$

The form of (3.16) is extremely informative since it shows how a single column function of the design variables is propagated over the cycle time. Indeed, it is shown later that functions of the dynamic response, such as energy, need only the single column of functions. For the response at a selected cycle time only the corresponding row of **T** is required and the column of functions.

Since rows of **Q** correspond to the cycle-time information of **Z**, the response at some important cycle time (e.g. peak or settling time) can be found by picking out the corresponding row in **Q** and pre-multiplying the matrices **W** and $\bar{\mathbf{r}}(\mathbf{v})$. If the response at cycle-time step τ_c is required, the corresponding row number of **Q** is

$$c = 1 + \frac{\tau_L}{\tau_c} \tag{3.17}$$

Where *c* represents the row number of **Q** corresponding to the cycle-time of interest. For example, if $\tau_L = 0.1$ and the response at cycle time $\tau_c = 0.004$, then $c = 26$ which means that the 26th row of **Q** is selected.

$$\hat{\mathbf{z}}(\mathbf{v},\tau_c) = \mathbf{q}(\tau_c)\mathbf{W}\bar{\mathbf{r}}(\mathbf{v}) \tag{3.18}$$

Where $\mathbf{q}(\tau_c)$ represents the row in **Q** that corresponding to the cycle-time of interest.

In gradient-based FORM optimization, the first derivative is required. From (3.14)

$$\frac{\partial \hat{\mathbf{z}}(\mathbf{v},\mathbf{t})}{\partial \mathbf{v}^T} = \mathbf{Q}\,\mathbf{W}\frac{\partial \bar{\mathbf{r}}(\mathbf{v})}{\partial \mathbf{v}^T} \tag{3.19}$$

If, however, $\mathbf{r}(\mathbf{v})$ is the same, then differentiating (3.16)

$$\frac{\partial \hat{\mathbf{z}}(\mathbf{v},\mathbf{t})}{\partial \mathbf{v}^T} = \mathbf{Q}(\mathbf{W}\mathbf{I})\frac{\partial \mathbf{r}(\mathbf{v})}{\partial \mathbf{v}^T} \tag{3.20}$$

Dynamic Degrading Systems

The metamodel work involved in static and dynamic systems can be combined to solve the degradation problem in (1.3). This is discussed in detail in Chapter 7. Now, the theory behind each metamodel will be discussed.

3.4 Regression Models

Regression models (RM), were originally developed to analyze the results of physical experiments to create empirically based models of the observed response values. Given a response, *z*, and a vector of design variables, **v** influencing *z*, the relationship between *z* and **v** is

$$\hat{z} = f(\mathbf{v}, \boldsymbol{\beta}) + \varepsilon \tag{3.21}$$

ε is the random error and is assumed $\varepsilon \sim N\left(0, \sigma_{RM}^2\right)$. The error, ε_i , at each observation is assumed to be independent and identically distributed. The most widely used Regression Models are low-order polynomials. In this work, the accuracy and speed a second-order polynomial is investigated. The general form of these models is

$$f(\mathbf{v}, \boldsymbol{\beta}) = \beta_0 + \sum_{i=1}^{\eta} \beta_i v_i + \sum_{i=1}^{\eta} \sum_{j=1}^{\eta} \beta_{ij} v_i v_j + \varepsilon \tag{3.22}$$

The model parameter, ***β***, represents the expected change in the response *z* per unit change in v_j when all remaining independent variables v_i (*i≠j*) are held constant (Montgomery 2005). Equation (3.22) can be converted to the general metamodel form by grouping together the fitting parameters, **β**, to form **w** and the corresponding functions of **v** make up r(**v**) as shown

$$\mathbf{r}(\mathbf{v}) = \left[1 \; v_1 \ldots v_\eta \quad v_1^2 \ldots v_\eta^2 \quad v_1 v_2 \ldots v_i v_j \left(\forall i < j\right)\right]_{L\times 1}^T \tag{3.23}$$

The length *L* of $\mathbf{w}^{\mathrm{T}}$ and **r**(**v**) depends upon the degree of the polynomial and number of design variables η. For a second order Regression Model, the length, *L*, is found from

$$L = \prod_{j=0}^{1} \frac{\eta + (2 - j)}{2 - j} \tag{3.24}$$

The *L* fitting parameters in **w** are estimated using the least square method (Montgomery 2005). First, the data matrix **X** shown in (3.1), is augmented by both a unit vector to its left and an appropriate sub-matrix of functions of **X** to its right. Therefore,

$$\mathbf{X}_a = \left[\mathbf{1} \quad \mathbf{X} \quad f(\mathbf{X})\right]_{(M \times L)} \tag{3.25}$$

The sub-matrix of functions of **X**, *f*(**X**), reflects the higher-order and interaction terms of the vector (3.23). By the least squares method, the fitting parameters for the *h*th column of **D** are

$$\hat{\mathbf{w}}_h = \left[\mathbf{X}_a^T \mathbf{X}_a\right]^{-1} \mathbf{X}_a^T \mathbf{d}_h \tag{3.26}$$

Since $\mathbf{w}^{\mathrm{T}}$ is a row vector of constants, to find the derivative of the RM, only derivatives of the vector **r**(**v**) is required. Therefore,

$$\frac{\partial \mathbf{r}(\mathbf{v})}{\partial \mathbf{v}^T} = \begin{bmatrix} \mathbf{0}_{1\times\eta} \\ \mathbf{I}_{\eta\times\eta} \\ 2\mathbf{Iv}_{\eta\times\eta} \\ F(\mathbf{v})_{(L-2\eta-1)\times\eta} \\ \end{bmatrix} \tag{3.27}$$

Note that for the RM, **r**(**v**) and, hence, its derivative never changes. Therefore, the only task is to estimate $\mathbf{w}_h$.

Example 1: Converting the standard RM to a general metamodel.

As an example to show how the Regression Model can be converted to the general metamodel form, consider a 2nd order RM with two design variables from (3.24), *L* = 6

$$\hat{d}(\mathbf{v}) = \beta_0 + \beta_1 v_1 + \beta_2 v_2 + \beta_3 v_1^2 + \beta_4 v_2^2 + \beta_5 v_1 v_2 \tag{3.28}$$

Grouping and separating the constants, **β**, and the variables, **v**, two vectors from (3.28) are obtained

$$\mathbf{w}^T = [\beta_0 \quad \beta_1 \quad \beta_2 \quad \beta_3 \quad \beta_4 \quad \beta_5]_{1\times 6} \tag{3.29}$$

$$\mathbf{r}(\mathbf{v}) = \left[1 \quad v_1 \quad v_2 \quad v_1^2 \quad v_2^2 \quad v_1 v_2\right]_{6\times 1}^T \tag{3.30}$$

Differentiating the Regression Model is not difficult because **r**(**v**) does not contain complex functions of the design variables. Therefore, the matrix of derivatives of **r**(**v**) with respect to each variable is

$$\frac{\partial \mathbf{r}(\mathbf{v})}{\partial \mathbf{v}} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 2v_1 & 0 \\ 0 & 2v_2 \\ v_2 & v_1 \end{bmatrix} = \begin{bmatrix} [\mathbf{0}]_{1\times 2} \\ [\mathbf{I}]_{2\times 2} \\ 2[\mathbf{I}(\mathbf{v})]_{2\times 2} \\ [F(\mathbf{v})]_{1\times 2} \end{bmatrix} \tag{3.31}$$

Notice the dimensions of **0**, **I**, **I**(**v**) and *F*(**v**) in (3.31) coincide with the general form in (3.27).

3.5 Kriging Model

Kriging predicts the response of unobserved points (i.e. those whose response has not been obtained by the simulation) based on all of the training responses (i.e. the response that has already been obtained). It is a method of spatial prediction that is based on minimizing the mean error of the weighting sum of the sampling values. The general form of the Kriging metamodel is

$$\hat{d}(\mathbf{v}) = f(\mathbf{v}) + \varepsilon(\mathbf{v}) \tag{3.32}$$

There are two main types of Kriging model and each is identified by the form of f(**v**). In Universal Kriging, f(**v**) is a known function (usually a linear Regression Model) that “globally” approximates the design space. In Ordinary Kriging, f(**v**) takes a constant value, β (Simpson, Peplinski, et al. 2001). Herein a universal Kriging metamodel is used.

The second part, ε(**v**), is a realization of a stochastic process with mean 0, variance σ^2 and nonzero covariance (Rijpkema, Etman and Schoofs 2001). It creates “localized” deviations so that the Kriging model interpolates the *M* sampled data points (Martin and Simpson, 2003). The covariance is calculated from the product of the variance and a correlation function as shown

$$\mathrm{cov}[\varepsilon(\mathbf{x}_j), \varepsilon(\mathbf{x}_k)] = \sigma^2 \gamma(\mathbf{x}_j, \mathbf{x}_k) \tag{3.33}$$

The correlation function, $\gamma(\mathbf{x}_j,\mathbf{x}_k)$, affects the smoothness of the model, the impact or weight of nearby points and the differentiability of the surface by quantifying the correlation between two observations (Martin and Simpson, 2003). Kriging requires iterative calculations to estimate the model parameters; therefore, a correlation function with the least number of fitting parameters is most desirable. The most commonly used is the Gaussian correlation function

$$\gamma(\mathbf{x}_j, \mathbf{x}_k) = \exp\left(\sum_{i=1}^{\eta}\left(-\theta_i\left(x_{i,j} - x_{i,k}\right)^2\right)\right) \tag{3.34}$$

where $\boldsymbol{\theta}$ is a vector of the fitting parameter.

The general form of the Kriging model is

$$\hat{d}_h(\mathbf{v}) = \beta_{0,h} + \tilde{\boldsymbol{\beta}}_h^T \mathbf{v} + \tilde{\mathbf{r}}_h^T(\mathbf{v})\boldsymbol{\Gamma}_h^{-1}(\mathbf{d}_h - \mathbf{X}_b\boldsymbol{\beta}_h) \tag{3.35}$$

where $\tilde{\boldsymbol{\beta}}^T = [\beta_1 \quad \dots \quad \beta_\eta]$, $\mathbf{v} = [v_1 \quad \dots \quad v_\eta]$, $\mathbf{X}_b = [\mathbf{1} \quad \mathbf{X}]_{M\times(\eta+1)}$, $\boldsymbol{\beta} = [\beta_0 \quad \tilde{\boldsymbol{\beta}}^T]_{1\times(\eta+1)}$, $\mathbf{d}_h$ is the vector of responses in column h of $\mathbf{D}$ and

$$\boldsymbol{\Gamma} = \begin{bmatrix} \gamma(\mathbf{x}_1,\mathbf{x}_1) & \gamma(\mathbf{x}_1,\mathbf{x}_2) & \dots & \gamma(\mathbf{x}_1,\mathbf{x}_M) \\ \gamma(\mathbf{x}_2,\mathbf{x}_1) & \gamma(\mathbf{x}_2,\mathbf{x}_2) & \dots & \gamma(\mathbf{x}_2,\mathbf{x}_M) \\ \vdots & \vdots & \ddots & \vdots \\ \gamma(\mathbf{x}_M,\mathbf{x}_1) & \gamma(\mathbf{x}_M,\mathbf{x}_2) & \dots & \gamma(\mathbf{x}_M,\mathbf{x}_M) \end{bmatrix}_{M\times M} \tag{3.36}$$

The derivation of the Kriging model is shown in Appendix A. Now, if the Kriging model is converted to the general form, **w** and **r**(**v**) would look like

$$\mathbf{w}_h^T = \left[\boldsymbol{\beta}_h \quad \boldsymbol{\Gamma}_h^{-1}(\mathbf{d}_h - \mathbf{X}_b\boldsymbol{\beta}_h)\right]_{1\times(M+(\eta+1))} \tag{3.37}$$

$$\mathbf{r}_h(\mathbf{v}) = \left[1 \quad \mathbf{v} \quad \tilde{\mathbf{r}}_h(\mathbf{v})\right]^T_{(M+(\eta+1))\times 1} \tag{3.38}$$

Each element in $\tilde{\mathbf{r}}(\mathbf{v})$ is computed using the expression

$$\tilde{r}_{j,h} = \exp\left(-\sum_{i=1}^{\eta}\theta_{i,h}\left(v_i - x_{ij}\right)^2\right) = \quad j = 1\dots M \tag{3.39}$$

which can be further simplified

$$\tilde{r}_{j,h}(\mathbf{v}) = \exp(-\rho_{j,h}(\mathbf{v})) \tag{3.40}$$

where

$$\rho_{j,h}(\mathbf{v}) = [\mathbf{v} - \mathbf{x}_j]^T[\boldsymbol{\theta}_h][\mathbf{v} - \mathbf{x}_j] \tag{3.41}$$

In some cases, $\boldsymbol{\theta}$ can be the same for each of the s columns of **D**. If this can happen, then (3.37) does not change for each column of **D** nor does $\boldsymbol{\Gamma}^{-1}$. Therefore vectors (3.37) and (3.38) become

$$\mathbf{w}_h^T = \left[\boldsymbol{\beta}_h \quad \boldsymbol{\Gamma}^{-1}(\mathbf{d}_h - \mathbf{X}_b\boldsymbol{\beta}_h)\right]_{1\times(M+(\eta+1))} \tag{3.42}$$

$$\mathbf{r}(\mathbf{v}) = \left[1 \quad \mathbf{v} \quad \tilde{\mathbf{r}}(\mathbf{v})\right]^T_{(M+(\eta+1))\times 1} \tag{3.43}$$

This simplifies Kriging further since **r**(**v**) is generated only once and $\boldsymbol{w}_h^T$ depends only upon $\mathbf{d}_h$.

Example 2: Converting the Kriging Model to a general metamodel.

Consider a Kriging model consisting of two design variables, v_1 and v_2. In its common form, this model would look like

$$\hat{d}_h(\mathbf{v}) = \beta_{0,h} + \beta_{1,h}v_1 + \beta_{2,h}v_2 + \tilde{\mathbf{r}}_h^T(\mathbf{v})\boldsymbol{\Gamma}_h^{-1}(\mathbf{d}_h - \mathbf{X}_b\boldsymbol{\beta}_h) \tag{3.44}$$

Where

$$\tilde{\mathbf{r}}_h^T(\mathbf{v}) = \left[\exp(-\rho_{1,h}(\mathbf{v})) \quad \cdots \quad \exp(-\rho_{M,h}(\mathbf{v}))\right]_{1\times M} \tag{3.45}$$

Grouping and separating constants and variables

$$\hat{d}_h(\mathbf{v}) = \beta_{0,h} + \tilde{\boldsymbol{\beta}}_h^T\mathbf{v} + \tilde{\mathbf{r}}_h^T(\mathbf{v})\boldsymbol{\Gamma}_h^{-1}(\mathbf{d}_h - \mathbf{X}_b\boldsymbol{\beta}_h) \tag{3.46}$$

Where $\tilde{\boldsymbol{\beta}}_h^T = \left[\beta_{1,h} \quad \beta_{2,h}\right]$, $f(\mathbf{v}) = \left[v_1 \quad v_2\right]^T$, $\boldsymbol{\beta}_h = \left[\beta_{0,h} \quad \beta_{1,h} \quad \beta_{2,h}\right]^T$ and $\mathbf{X}_b = \begin{bmatrix} 1 & x_{1,1} & x_{1,2} \\ 1 & x_{2,1} & x_{2,2} \\ \vdots & \vdots & \vdots \\ 1 & x_{M,1} & x_{M,2} \end{bmatrix}_{M\times 3}$.

From these groupings, $\tilde{\boldsymbol{\beta}}^T$, $\boldsymbol{\beta}$ and $\boldsymbol{\Gamma}^{-1}(\mathbf{d} - \mathbf{X}_b\boldsymbol{\beta})$ are all vectors whose elements contain constants. Therefore, further grouping all these terms into the matrix $\mathbf{w}^{\mathrm{T}}$, the two design variable universal Kriging model looks like

$$\mathbf{w}_h^T = \left[\boldsymbol{\beta}_h^T \quad \left(\boldsymbol{\Gamma}_h^{-1}(\mathbf{d}_h - \mathbf{X}_b\boldsymbol{\beta}_h)\right)\right]_{1\times(M+3)} \tag{3.47}$$

$$\left[\mathbf{r}_h(\mathbf{v})\right] = \left[1 \quad \mathbf{v}^T \quad \tilde{\mathbf{r}}_h^T(\mathbf{v})\right]^T_{(M+3)\times 1} \tag{3.48}$$

3.5.1 Estimating Kriging Model Parameters

In most instances, the parameters **θ** and **β**, are estimated, for each column of **D**, using Maximum likelihood estimation (MLE) (Martin and Simpson 2003). From MLE, the log of the Gaussian likelihood function is

$$L(\boldsymbol{\beta}_h, \sigma_{KG}^2(\mathbf{d}_h), \boldsymbol{\theta}_h) = -\frac{M}{2}\ln(2\pi\sigma_{KG}^2(\mathbf{d}_h)) - \frac{1}{2}\ln|\boldsymbol{\Gamma}_h| - \frac{1}{2\sigma_{KG}^2(\mathbf{d}_h)}(\mathbf{d}_h - \mathbf{X}_b\boldsymbol{\beta}_h)^T\boldsymbol{\Gamma}_h(\mathbf{d}_h - \mathbf{X}_b\boldsymbol{\beta}_h) \tag{3.49}$$

MLE assumes that the residuals have a known probability distribution shape, which in most cases is assumed to be the Gaussian probability distribution. By taking the derivative of the log-likelihood equation with respect to σ^2 and **β** and solving for zero, the closed-form solutions are

$$\hat{\sigma}_{KG}^2(\mathbf{d}_h) = \frac{1}{M}(\mathbf{d}_h - \mathbf{X}_b\boldsymbol{\beta}_h)\boldsymbol{\Gamma}^{-1}(\mathbf{d}_h - \mathbf{X}_b\boldsymbol{\beta}_h) \qquad (3.50)$$

$$\hat{\boldsymbol{\beta}}_h = \left(\mathbf{X}_b^T\boldsymbol{\Gamma}_h^{-1}\mathbf{X}_b\right)^{-1}\mathbf{X}_b^T\boldsymbol{\Gamma}_h^{-1}\mathbf{d}_h \qquad (3.51)$$

An explicit form does not exist for **θ** for most SCFs; therefore, numerical optimization is used (Simpson, et al. 2001). Therefore, after substituting equations (3.50) and (3.51) into (3.49), the best estimate of correlation parameter, **θ**, is found by maximizing

$$L\left(\hat{\boldsymbol{\theta}}_h\right) = -\frac{1}{2}\left(M\ln\left(\hat{\sigma}_{KG}^2(\mathbf{d}_h)\right) + \ln\left|\boldsymbol{\Gamma}_h(\boldsymbol{\theta})\right|\right) \qquad (3.52)$$

To maximize (3.52), a simulated annealing algorithm is normally used (Kleijnen and Van Beers 2003).

Although the MLE is the common method used to estimate Kriging model parameters, it assumes that the data follows a Gaussian distribution. Cross-validation can also be used to estimate model parameters where the data does not follow a Gaussian distribution. Cross-validation of a Kriging model is determined by holding all of the model parameters, **β**,$\hat{\sigma}_{KG}^2$ and **θ**, constant while creating N_f Kriging models using each subset of the remaining M – (M/N_f) points and calculating the error at each omitted location in turn (Martin and Simpson 2003). The best parameters are those that minimize the cross-validation mean squared error. Martin and Simpson 2003 presents a method that uses Cross-Validation to estimate Kriging model parameters.

3.5.2 Kriging Derivatives

Recall the first derivative of the general metamodel is

$$\frac{\partial \hat{d}_h(\mathbf{v})}{\partial v_i} = \mathbf{w}_h^T \frac{\partial \mathbf{r}_h(\mathbf{v})}{\partial v_i} \qquad (3.53)$$

The right side derivative in (3.53) looks like

$$\frac{\partial \mathbf{r}_h(\mathbf{v})}{\partial v_i} = \begin{bmatrix} 0 & \dfrac{\partial \mathbf{v}^T}{\partial v_i} & \left(\dfrac{\partial \tilde{\mathbf{r}}_h(\mathbf{v})}{\partial v_i}\right)^T \end{bmatrix}^T \qquad (3.54)$$

From the chain rule of differentiation

$$\frac{\partial \tilde{r}_{j,h}(\mathbf{v})}{\partial v_i} = \frac{\partial \tilde{r}_{j,h}(\mathbf{v})}{\partial \rho_{j,h}(\mathbf{v})} \times \frac{\partial \rho_{j,h}(\mathbf{v})}{\partial v_i} \tag{3.55}$$

After some simplification the derivative in (3.55) becomes

$$\frac{\partial \tilde{r}_{j,h}(\mathbf{v})}{\partial v_i} = -2\theta_{i,h}\left(v_i - x_{ij}\right)\tilde{r}_{j,h}(\mathbf{v}) \tag{3.56}$$

Therefore,

$$\left(\frac{\partial \tilde{\mathbf{r}}_h(\mathbf{v})}{\partial v_i}\right) = \begin{bmatrix} -2\theta_{i,h}\left(v_i - x_{1,j}\right)r_{1,h}(\mathbf{v}) \\ \vdots \\ -2\theta_{i,h}\left(v_i - x_{M,j}\right)r_{M,h}(\mathbf{v}) \end{bmatrix}_{M\times 1} \tag{3.57}$$

Example 3: Differentiating the Kriging Metamodel.

Returning to example 2, the first derivative with respect to v_1 looks like

$$\frac{\partial \mathbf{r}_h(\mathbf{v})}{\partial v_1} = \begin{bmatrix} 0 & 1 & 0 & \left(\frac{\partial \tilde{\mathbf{r}}_h(\mathbf{v})}{\partial v_1}\right)^T \end{bmatrix}^T \tag{3.58}$$

Now, following equation (3.56)

$$\frac{\partial \tilde{r}_{j,h}(\mathbf{v})}{\partial \rho_{j,h}(\mathbf{v})} = -\exp\left(-\rho_{j,h}(\mathbf{v})\right) \tag{3.59}$$

$$\frac{\partial \rho_{j,h}(\mathbf{v})}{\partial v_1} = 2\theta_{1,h}\left(v_1 - x_{j,1}\right) \tag{3.60}$$

and after simplification

$$\frac{\partial \tilde{r}_{j,h}(\mathbf{v})}{\partial v_1} = -2\theta_{1,h}\left(v_1 - x_{1,j}\right)\exp\left(-\rho_{j,h}(\mathbf{v})\right) \tag{3.61}$$

But $\tilde{r}_{j,h}(\mathbf{v}) = \exp\left(-\rho_{j,h}(\mathbf{v})\right)$, therefore,

$$\left(\frac{\partial \tilde{\mathbf{r}}_{j,h}(\mathbf{v})}{\partial v_1}\right)^T = \begin{bmatrix} -2\theta_{1,h}\left(v_1 - x_{1,1}\right)\tilde{r}_{1,h}(\mathbf{v}) & \cdots & -2\theta_{1,h}\left(v_1 - x_{M,1}\right)\tilde{r}_{M,h}(\mathbf{v}) \end{bmatrix} \tag{3.62}$$

We repeat the above for v_2 and, finally, for two variables

$$\frac{\partial \mathbf{r}_h(\mathbf{v})}{\partial \mathbf{v}} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ -2\theta_{1,h}(v_1 - x_{1,1})\tilde{r}_{1,h}(\mathbf{v}) & -2\theta_{2,h}(v_2 - x_{1,1})\tilde{r}_{1,h}(\mathbf{v}) \\ \vdots & \vdots \\ -2\theta_{1,h}(v_1 - x_{M,1})\tilde{r}_{M,h}(\mathbf{v}) & -2\theta_{2,h}(v_2 - x_{M,2})\tilde{r}_{M,h}(\mathbf{v}) \end{bmatrix}_{(M+3)\times 2} \tag{3.63}$$

3.6 Radial Basis Functions

Another popular metamodel, the Radial Basis Function (RBF), is a type of layered feed-forward neural network (NN) capable of approximating any continuous function (Andina and Pham 2007). The RBF network consists of two layers. The input level distributes input vectors to each of the receptive field units in the second layer (hidden layer) without any multiplicative factors. The hidden layer has *M* receptive field units (or hidden units) each of which represents a nonlinear transfer function called a basis function (Karray and De Silver 2004).

The hidden units play a role in simultaneously receiving the input vector and nonlinearly transforming the input vector into an *M*-dimensional vector. The outputs from the *M*-hidden units are then linearly combined with weights to produce the network output at the output layer. Thus, the typical RBF model is described by specifying the number of “basis functions” (hidden units), basis function parameters, and the weights of the basis function outputs to produce the network output.

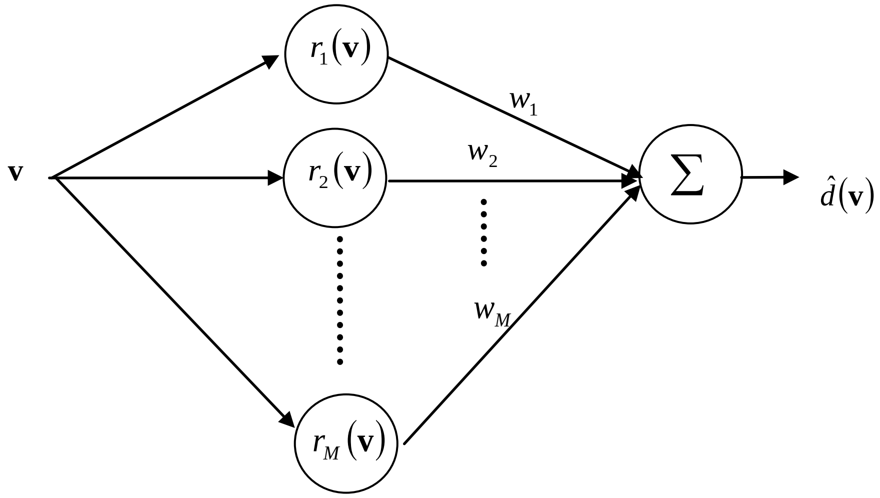


Figure 3-2: A Radial Basis Function Neural Network (Deng 2006).

Given an input vector **v**, the output of the RBF network is given by

$$\hat{d}_h(\mathbf{v}) = \mathbf{w}_h^T \mathbf{r}(\mathbf{v}) \tag{3.64}$$

where r(**v**) represents the RBF and *M* is the number of training sets. The most commonly used Radial Basis Functions are multiquadratics, inverse multiquadratics and Gaussians (Deng 2006).

Table 3-1: The most common RBF functions

RBF Name	**RBF**
Multi-quadratic	$\sqrt{\rho_j^2(\mathbf{v}) + b_j^2}$
Inverse multi-quadratic	$\frac{1}{\sqrt{\rho_j^2(\mathbf{v}) + b_j^2}}$
Gaussians	$\exp\left(-\frac{\rho(\mathbf{v})_j^2}{b_j^2}\right)$
Thin Plate Spline	$\rho(\mathbf{v})_j^2 \ln\left(b_j^{\rho_j(\mathbf{v})}\right)$
Cubic	$\left(\rho_j(\mathbf{v}) + b_j\right)^3$

In this work, the feasibility of the simplest function, the multiquadratic RBF, is investigated. From Table 3-1, for a multi-quadratic RBF, each element of **r**(**v**) in (3.64) looks like

$$r_{j,h}(\mathbf{v}) = \sqrt{\rho_j^2(\mathbf{v}) + b_{j,h}^2} \qquad j = 1, \ldots, M \tag{3.65}$$

Where ρ_j is the Euclidean distance calculated using (3.65) and b_j is the metamodel parameter to be estimated.

$$\rho_j^2(\mathbf{v}) = (\mathbf{v} - \mathbf{x}_j)^T (\mathbf{v} - \mathbf{x}_j) = \sum_{i=1}^{\eta} (v_i - x_{ij})^2 \quad j = 1 \ldots M \tag{3.66}$$

Efficient training algorithms have been developed to minimize the sum of squared error by adaptively updating the fitting parameters of the RBF network. These parameters are the centre of each basis function (centroids of the hidden layer), the receptive field widths and the output layer weights. Training of the RBF metamodel involves choosing the centre and width of each RBF and calculating the weights of the output layer. This can be done using the *k*-mean clustering approach. However, most analysis involving the RBF metamodel uses all the training sets as the centres to produce a total of *M* basis functions.

Since all training sets are taken to be the centres of the network and the width of the j^{th} basis function is determined using cross-validation. To estimate the weight matrix, **w**, the inverse or pseudo-inverse method is normally used. The matrix **R**(**x**) is defined in (3.67) where each element is calculated using equation (3.68).

$$\mathbf{R}(\mathbf{x}) = \begin{bmatrix} r(\mathbf{x}_1, \mathbf{x}_1) & r(\mathbf{x}_1, \mathbf{x}_2) & \ldots & r(\mathbf{x}_1, \mathbf{x}_M) \\ r(\mathbf{x}_2, \mathbf{x}_1) & r(\mathbf{x}_2, \mathbf{x}_2) & \ldots & r(\mathbf{x}_2, \mathbf{x}_M) \\ \vdots & \vdots & \ddots & \vdots \\ r(\mathbf{x}_M, \mathbf{x}_1) & r(\mathbf{x}_M, \mathbf{x}_2) & \ldots & r(\mathbf{x}_M, \mathbf{x}_M) \end{bmatrix}_{M \times M} \tag{3.67}$$

$$r(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\left(\sum_{i=1}^{\eta} (x_{i,j} - x_{i,k})^2 + b^2 \right)} \tag{3.68}$$

If $\mathbf{R}^{-1}$ exists, then the weight matrix is estimated from

$$\hat{\mathbf{w}}_h = \mathbf{R}_h^{-1} \mathbf{d}_h \tag{3.69}$$

If, however, **R** is ill-conditioned or even non-square, then the pseudo-inverse method is used to estimate the weight matrix (Karray and De Silver 2004)

$$\hat{\mathbf{w}}_h = \left(\mathbf{R}_h^T\mathbf{R}_h\right)^{-1}\mathbf{R}_h^T\mathbf{d}_h \tag{3.70}$$

3.6.1 Radial Basis Function Derivatives

Like the Kriging model, differentiation of $r_j(\mathbf{v})$ for the RBF involves the chain rule of differentiation as follows

$$\frac{\partial r_{j,h}(\mathbf{v})}{\partial v_i} = \frac{\partial r_{j,h}(\mathbf{v})}{\partial \rho_{j,h}(\mathbf{v})} \times \left(\frac{\partial \rho_{j,h}(\mathbf{v})}{\partial v_i}\right) \tag{3.71}$$

Using eqs. (3.65) and (3.66), the two terms on the right hand side of (3.71) are

$$\frac{\partial r_{j,h}(\mathbf{v})}{\partial \rho_{j,h}} = \frac{\rho_{j,h}}{\sqrt{\left(\rho_{j,h}^2 + b_{j,h}^2\right)}} \tag{3.72}$$

$$\frac{\partial \rho_{j,h}}{\partial v_i} = \frac{\left(v_i - x_{i,j}\right)}{\rho_{j,h}} \tag{3.73}$$

After substituting equations (3.72) and (3.73) into (3.71), the first derivative of $r_j(\mathbf{v})$ with respect to v_i is

$$\frac{\partial r_{j,h}(\mathbf{v})}{\partial v_i} = \frac{\left(v_i - x_{ij}\right)}{\sqrt{\rho_{j,h}^2 + b_{j,h}^2}} \quad j = 1 \ldots M \tag{3.74}$$

3.7 Metamodel Error Analysis

Now, after estimating metamodel parameters, the next step is to determine the fit of the metamodel. Checking the fit of the metamodel determines if the metamodel has been able to capture the relationship between the design variable and the response with an acceptable amount of accuracy. A poor fit will result in inaccurate estimates of the response at untried design sets and will ultimately result in an inaccurate calculation of the best design. There are two main ways to determine if a metamodel is a good fit 1) estimating the fitting error and 2) estimating the predictive error. Estimating the fitting error determines how well the metamodel fits the training design. Estimating the predictive error, on the other hand, determines how well the metamodel would predict the response at some new training design set sample.

When building a response surface model, the commonly used method to check the fit of the model is to calculate the coefficient of determination (R^2).

$$R^2 = 1 - \frac{SS_{Err}}{SS_{Total}} \tag{3.75}$$

In (3.75), for $\mathbf{d}_h$, the Error Sum of Squares (SS_{Err}) and the Total Sum of Squares (SS_{Total}) are estimated using (3.76) and (3.77) respectively.

$$SS_{Err}^{(h)} = \sum_{i=1}^{M} \left(d_{h,i} - \hat{d}_{h,i} \right) \quad h = 1, \ldots, s \tag{3.76}$$

$$SS_{Total}^{(h)} = \sum_{i=1}^{M} \left(d_{h,i} - \bar{d}_h \right)^2 \quad h = 1, \ldots, s \tag{3.77}$$

Since one of the features of Kriging is to exactly predict the training design (Kleijnen and Van Beers 2003); (Sakata, Ashida and Zako 2003), $SS_{Err} = 0$ thereby always producing an R^2 value equal to 1 thus providing an inaccurate estimate of the performance of the method. One option is to fit the model with the existing training design and then run the simulation again to obtain responses at some additional sample design variable sets. The accuracy of the metamodel is then checked by calculating the error of the metamodel prediction at the additional sample sets. Although effective in determining the accuracy of the metamodel in prediction, this method would require additional simulations which can be potentially expensive.

Using this idea of checking the fit at new design variable sets instead of the training sets, authors use the cross-validation method to determine the fit of a particular metamodel. In this method, the training design is divided into N_f groups normally called "folds". For a particular fold, there is a total of (M/N_f) design variable sets that are used as test sets and the remaining ($M - M/N_f$) training sets are used to fit the metamodel. This process is repeated until all folds are used to both train and validate the metamodel (Meckesheimer, et al. 2002); (Kleijnen and Sargent 2000). In the end, there are response estimates for all of the original M training sets.

As an illustration, suppose we have an experimental design made up of a total of 15 design variable sets. Suppose the 15 sets are grouped into 5 folds with each fold containing a total of 3 design variable sets. In cross-validation, one fold is withheld and the other 4 folds are used to fit the metamodel. Using the newly fit metamodel, the response at each of the design variable sets of the withheld fold is estimated. This procedure is repeated until all folds are used to fit the metamodel. Most common is the leave-one-out cross-validation method where one training set is withheld and the remaining sets are used to fit the metamodel. Eventually, there are estimates of the response at all the training sets. Using these response estimates, different methods can be used to assess the metamodel (Ahmed and Qin 2009); (Wang and Shan 2007). Among these are the Root Mean Squared Error (RMSE) and the Mean Absolute Error (MAE). The formulae for CV-RMSE and CV-MAE are shown below.

$$\text{CV - RMSE}^{(\text{h})} = \sqrt{\frac{\sum_{i=1}^{M}\left(d_{h,i} - \hat{d}_{h,i}\right)^2}{M}} \tag{3.78}$$

$$\text{CV - MAE}^{(\text{h})} = \frac{\sum_{i=1}^{M}\left|d_{h,i} - \hat{d}_{h,i}\right|}{M} \tag{3.79}$$

Since CV-RMSE and CV-MAE are error estimates, a smaller CV-RMSE or CV-MAE value is ideal.

3.8 Summary

In this chapter, two main steps in the proposed methodology for metamodel-based probabilistic design optimization of dynamic systems have been described in detail. These include (1) describing how SVD is used to reduce the number of metamodels to be built when provided with a time-varying response that is used to estimate the performance measure and (2) presenting the theory behind the three popular metamodels that are used throughout this thesis. A method for checking the predictive error of each metamodel, the CV-RMSE or CV-MAE, has also been described. Since the goal of this work is to present an accurate methodology, the sources of possible error must be known.

When approximating the response matrix $\mathbf{Z}$ with $\widehat{\boldsymbol{Z}}$, the three possible sources of error are shown in eq. (3.80)

$$\hat{\varepsilon}_{\hat{\mathbf{Z}}} = \hat{\varepsilon}_{\Delta\tau} + \hat{\varepsilon}_{SVD} + \hat{\varepsilon}_{\text{Metamodel}} \tag{3.80}$$

The first term on the right hand side of (3.80) refers to the step size, $\Delta\tau$, being too large. The step size is chosen by selecting a value for $\Delta\tau$ and then estimating the performance measure. The step size is then reduced and the performance measure is re-calculated. This process is repeated until an asymptotic value for the performance measure is reached.

The second source of error is from estimating the matrix **D** in SVD, $\widehat{\mathbf{D}}$, with a reduced number of columns in **S**. The adequate number of columns is checked in a similar way as checking the best step size. Here, a value of *s* is chosen and the performance measure estimated. If the error is negligible, stop, however, add another column to $\widehat{\boldsymbol{D}}$.

The third source of error is from the fit of the metamodel. The error of the metamodel is quantified using cross-validation methods. The only way to reduce this error is to choose a better metamodel or, perhaps, a different method to select the training design.

Chapter 4

Probability Evaluations and Design Optimization

One of the main goals is to optimize a dynamic system when design variables are uncertain. When uncertainty is assumed in the design variables, they are typically described by a type of distribution whose parameters are defined. Design optimization involving uncertain design variables tries to find the distribution parameters that minimize system failure or total cost. In order to estimate system failure, the conceptual components are assumed to follow a series or parallel arrangement. Estimating the probability of failure is important in design of systems with uncertain parameters. Therefore, two main points must be addressed. First, how to relate the response to the specification limits to determine when ‘success’ or ‘failure’ occurs and second, how to calculate the probability of failure (Madsen, Krenk and Lind 1986) (Melchers 1987). The first is answered using a ‘limit state function’. The probability of failure can be estimated using either a Monte Carlo Simulation (MCS) or some fast integration method such as the First-Order Reliability Method (FORM).

This chapter will show how limit-state functions and surfaces are built to relate the metamodel for the performance measure of interest with the design specification. The methods, MCS and FORM, will be presented to show how the probability of failure is estimated. Then, three design optimization methods will be presented to show how to find the best mean that minimizes system failure (parameter design), tolerance that minimizes cost (tolerance) or mean and tolerance that minimizes cost (integrated design).

4.1 Limit State Function

By definition, when provided with an upper or lower specification limit, ζ , the limit-state function (LSF) is written as

$$g(\mathbf{v}) = \pm(\zeta - \hat{z}(\mathbf{v})) \tag{4.1}$$

where $\hat{z}(\mathbf{v})$ is the metamodel. For an upper specification limit, $g(\mathbf{v}) = (\zeta - \hat{z}(\mathbf{v}))$ and for a lower specification limit $g(\mathbf{v}) = \hat{z}(\mathbf{v}) - \zeta$. By definition,

$$\begin{aligned} g(\mathbf{v}) &> 0 \quad \mathbf{v} \in \text{Safe Region} \\ g(\mathbf{v}) &= 0 \quad \mathbf{v} \in \text{Limit - State Surface} \\ g(\mathbf{v}) &< 0 \quad \mathbf{v} \in \text{Failure Region} \end{aligned}$$

The probability of conforming to specifications (success or S) is defined as $\Pr\{g(\mathbf{V}) > 0\}$, the probability of failure is $\Pr\{g(\mathbf{V}) < 0\}$. An example of the limit-state surface (LSS), in **v**-space, is shown in Figure 4-1.

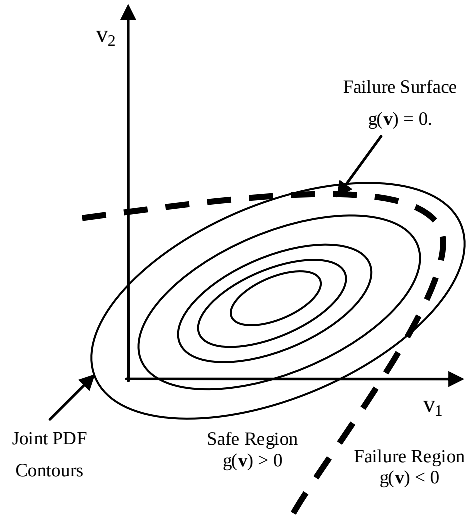


Figure 4-1: Failure surface in **v**-space.

If the joint *pdf* of **V** is $f_\mathbf{V}(\mathbf{v})$, then the probability of failure is evaluated using the integral (Madsen, Krenk and Lind 1986)

$$\Pr(F) = \Pr(g(\mathbf{V}) < 0) = \int_{g(\mathbf{v})<0} f_\mathbf{v}(\mathbf{v}) d\mathbf{v} \tag{4.2}$$

The direct evaluation of (4.2) is difficult for three reasons; first, since several random variables, **V,** are involved, the probability integration is multi-dimensional. Second, the integrand $f_{\mathbf{V}}(\mathbf{V})$ is the joint *pdf* of **V** and is generally a nonlinear multidimensional function and third, the limit-state surface, $g(\mathbf{v}) = 0$, is also multi-dimensional and usually a nonlinear function. Due to these three reasons, there is seldom an analytical solution to the probability integration to find the solution due to the high dimensionality in most engineering applications. There are two main methods to estimate the failure probability, these are sampling methods or fast-integration methods. These will be discussed later.

4.2 System Failure

In system reliability applications, when the system failure is to be estimated, the assumption is that the components follow either a series or parallel arrangement. This does not mean that the components are physically arranged in a series or parallel way but that the system failure is calculated based upon the well-known characteristics of these arrangements (Rao, 1992); (Savage and Son, 2011). If the system assumes a series arrangement, then system failure occurs if any one of the components fails. Using a set-theory interpretation, the system failure is

$$\mathbf{F} = F_1 \cup F_2 \cup \ldots \cup F_{n_{LF}} \tag{4.3}$$

In terms of probability, eq. (4.3) becomes

$$\Pr(\mathbf{F}) = \Pr\left(F_1 \cup F_2 \cup \ldots \cup F_{n_{LF}}\right) = \Pr\left(\bigcup_{i=1}^{n_{LF}} F_i\right) \tag{4.4}$$

In parallel systems, system failure occurs if all components fail. Therefore, from the set-theory interpretation

$$\Pr(\mathbf{F}) = \Pr\left(\bigcap_{i=1}^{n_{LF}} F_i\right) \tag{4.5}$$

4.3 Monte Carlo Simulation

Traditionally, sampling methods have been used to estimate probability. Among these, the most popular is the Monte Carlo Method that is based upon the theory of large numbers (Papadrakakis and Lagaros 2002). For a Monte Carlo evaluation, N sample sets of **v** are generated and substituted into each LSF. Then, the output at each sample set is observed to determine if failure occurs. For any sample set, an indicator, e_k, for the k^{th} sample set, $\mathbf{v}^{(k)}$

$$e_k = \begin{cases} 1 & if\ g_k\left(\mathbf{v}^{(k)}\right) < 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

For a series system we expand the idea in (4.6) and along with (4.4)

$$e_k = \begin{cases} 1 & \text{if } g_1\left(\mathbf{v}^{(k)}\right) < 0 \text{ or } g_2\left(\mathbf{v}^{(k)}\right) < 0 \text{ or} \ldots \text{or } g_{n_{LF}}\left(\mathbf{v}^{(k)}\right) < 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.7}$$

For a parallel system, the idea in (4.6) and the intent in (4.5)

$$e_k = \begin{cases} 1 & \text{if } g_1\left(\mathbf{v}^{(k)}\right) < 0 \text{ and } g_2\left(\mathbf{v}^{(k)}\right) < 0 \text{ and} \ldots \mathrm{g}_{\mathrm{n_{LF}}}\left(\mathbf{v}^{(k)}\right) < 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

Therefore, for a total of N sample sets, the number of failing sets is

$$N_f = \sum_{k=1}^{N} e_k$$

The system failure is

$$\Pr(\mathbf{F}) = \frac{N_f}{N} \tag{4.9}$$

4.4 Transformation of Variables from v-space to u-space

Approximation methods such as the first-order reliability method (FORM) or the second-order reliability method (SORM) simplify the evaluation of the integral in equation (4.2) by simplifying $f_{\mathbf{v}}(\mathbf{v})$ so that its contours become more regular and symmetric. The following section discusses transforming the variables to **u**-space where the contours of $f_{\mathbf{u}}(\mathbf{u})$ are symmetric.

In general, the design variables may have arbitrary distributions. Since probability is simply approximated in a standardized normal space a one-to-one transformation can be used to convert the non-normal and correlated variables into standard **u**-space using a transformation method such as the Rosenblatt transformation (Madsen, Krenk and Lind, 1986). Symbolically the transformation is

$$\mathbf{T} : \mathbf{U} \Leftarrow \mathbf{V} \tag{4.10}$$

Where **U** = U$_1$, U$_2$, ..., U$_\eta$ are uncorrelated and standardized normally distributed variables. The inverse transformation may also be obtained which is written as

$$\mathbf{T}^{-1} : \mathbf{U} \Rightarrow \mathbf{V} \tag{4.11}$$

In general for *j* random variables there are *j* equations of the form

$$\Gamma_T(\mathbf{u}, \mathbf{v}) = 0 \tag{4.12}$$

where a particular mapping is obtained by solving the equations given one of the sets of variables.

For our purposes, let the η-design variables be joint Normal and then the parameters in matrix form are

$$\mu_{\mathbf{V}} = \begin{bmatrix} \mu_{V_1} \\ \mu_{V_2} \\ \vdots \\ \mu_{V_\eta} \end{bmatrix} \text{ and } \mathbf{C}_{\mathbf{V}} = \begin{bmatrix} \sigma_{V_1}^2 & \text{cov}(V_1, V_2) & \dots & \text{cov}(V_1, V_\eta) \\ \text{cov}(V_2, V_1) & \sigma_{V_2}^2 & \dots & \text{cov}(V_2, V_\eta) \\ \vdots & \vdots & \ddots & \vdots \\ \text{cov}(V_\eta, V_1) & \text{cov}(V_\eta, V_2) & \dots & \sigma_{V_\eta}^2 \end{bmatrix}$$

In standard Normal space, by definition, we must have

$$\mu_{\mathbf{U}} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \text{and} \quad \mathbf{C}_{\mathbf{U}} = \begin{bmatrix} 1 & 0 & . & . \\ 0 & 1 & . & . \\ . & . & . & . \\ . & . & . & 1 \end{bmatrix}$$

The transformation is a generalization of the one-dimensional form $U = \frac{1}{\sigma}(V - \mu)$ and we write the linear form

$$\mathbf{U} = \mathbf{A}(\mathbf{V} - E[\mathbf{V}]) \tag{4.13}$$

where **A** is *η x η*. Also note that $\mathbf{A}^{-1}$ gives the reverse transformation.

$$\mathbf{V} = \mathbf{A}^{-1}\mathbf{U} + E[\mathbf{V}] \tag{4.14}$$

Let us now find **A**. From (4.14), we subtract the expected value, *E*[**U**], from each side and write

$$\Delta\mathbf{U} = \mathbf{A}(\Delta\mathbf{V})$$

Next, we multiply each side by the transpose to get

$$\Delta\mathbf{U}\Delta\mathbf{U}^{\mathbf{T}} = \mathbf{A}(\Delta\mathbf{V})(\Delta\mathbf{V}^{\mathbf{T}})\mathbf{A}^{T}$$

Where $\Delta\mathbf{U} = \mathbf{U} - E[\mathbf{U}]$ and $\Delta\mathbf{V} = \mathbf{V} - E[\mathbf{V}]$. The application of the expected value operator gives the covariance relation

$$\mathbf{C}_{\mathbf{U}} = \mathbf{A}\mathbf{C}_{\mathbf{V}}\mathbf{A}^{\mathbf{T}}$$

and from this we conclude that

$$\mathbf{A}\mathbf{C}_{\mathbf{V}}\mathbf{A}^{\mathbf{T}} = \mathbf{C}_{\mathbf{U}} = \mathbf{I}$$

We now pre-multiply by $\mathbf{A}^{-1}$ and then post-multiply by $(\mathbf{A}^{\mathbf{T}})^{-1}$ in order to isolate $\mathbf{C}_{\mathbf{V}}$. We have

$$\mathbf{C}_{\mathbf{V}} = (\mathbf{A}^{\mathbf{T}}\mathbf{A})^{-1}$$

or

$$\mathbf{A}\mathbf{A}^{\mathbf{T}} = \mathbf{C}_{\mathbf{V}}^{-1}$$

and by definition

$$\mathbf{A} = \text{Cholesky}(\mathbf{C}_{\mathbf{V}}^{-1})$$

or

$$\mathbf{A}^{-1} = \text{Cholesky}(\mathbf{C}_{\mathbf{V}})$$

The Cholesky decomposition can be found in any linear algebra text and essentially finds the matrix **A** that has a lower triangular form with positive diagonals. After transformation, the performance function becomes *g*(**u**) and eq. (4.2) becomes

$$\Pr(F_i) = \Pr\{g_i(\mathbf{U}) < 0\} = \int_{g_i(\mathbf{u})<0} \phi_{\mathbf{u}}(\mathbf{u})d\mathbf{u} \tag{4.15}$$

Where $\phi_{\mathbf{u}}(\mathbf{u})$ is the joint *pdf* of **U**.

Example 1: Mapping from U-space to V-space.

Suppose we have the two-variable covariance matrix

$$\mathbf{C}_{\mathbf{v}} = \begin{bmatrix} \sigma_2^2 & \rho\sigma_2\sigma_3 \\ \rho\sigma_2\sigma_3 & \sigma_3^2 \end{bmatrix}$$

Then,

$$\mathbf{A}^{-1} = Cholesky(\mathbf{C}_{\mathbf{V}}) = \begin{bmatrix} \sigma_2 & 0 \\ \sigma_3\rho & \sigma_3\sqrt{1-\rho^2} \end{bmatrix}$$

and we write the mapping explicitly as

$$\begin{bmatrix} v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} \sigma_2 & 0 \\ \sigma_3\rho & \sigma_3\sqrt{1-\rho^2} \end{bmatrix} \begin{bmatrix} u_2 \\ u_3 \end{bmatrix} + \begin{bmatrix} \mu_2 \\ \mu_3 \end{bmatrix}$$

Suppose the limit-state function is

$$g(\mathbf{v}) = \zeta - \frac{v_3}{v_2 + v_3}$$

Then, in **u**-space

$$g(\mathbf{u},\mathbf{p}) = \zeta - \frac{(\mu_3 + \rho\sigma_3 u_2 + \sigma_3\sqrt{1-\rho^2}u_3)}{(\mu_2 + \sigma_2 u_2) + (\mu_3 + \rho\sigma_3 u_2 + \sigma_3\sqrt{1-\rho^2}u_3)}$$

where the space variables **u** are [u_2, u_3] and the design parameters **p** are $[\mu_2, \mu_3, \sigma_2, \sigma_3, \rho]$. Note the special case when the correlation coefficient ρ is zero.

4.5 First-Order Reliability Method (FORM)

Since the standard normal random variables are independent, the joint *pdf* is the product of the individual *pdfs*, and is given by

$$\phi_{\mathbf{u}}(\mathbf{u}) = \prod_{i=1}^{\eta} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}u_i^2\right) \tag{4.16}$$

The probability of failure is then evaluated as

$$\Pr(F)=\int_{g(u_1\ldots u_\eta)<0}\cdots\int\prod_{i=1}^{\eta}\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}u_i^2\right)du_1\ldots du_\eta \tag{4.17}$$

The first-order reliability method (FORM) makes the assumption that the failure surface can be fitted exactly with a tangent hyper-plane through **u*** which is the closest point to the origin on the LSS. Since the probability density function is rotationally symmetric, the linear-approximation of the failure surface can be rotated to any convenient position without changing the probability content on either side of the surface. The most convenient position is such that it is perpendicular to any single axis, since then, the probability calculation becomes one-dimensional.

Example 2

Consider the case for two variables shown in Figure 4-2. In order to estimate the failure probability, the failure surface is first approximated by a “failure-line” tangent at the Most Likely Failure Point, MLFP, denoted as **u***, and a distance β_R from the origin. Next, this line is rotated so that it is perpendicular to u_1 on the left of the origin (on the negative u_1 axis and parallel to u_2). Notice the position of the new, but equivalent, non-conformance region. Figure 4-2(a) shows a limit state surface (curve) fitted by a tangent hyperplane (solid line) at **u***. Figure 4-2(b) shows the tangent hyperplane rotated to a point perpendicular to u_1 and parallel to u_2. For the bivariate, standard normal density function $\phi(u_1,u_2)$

$$\Pr(F)=\int_{-\infty}^{-\beta_R}\frac{e^{-\frac{u_1^2}{2}}}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\frac{e^{-\frac{u_2^2}{2}}}{\sqrt{2\pi}}du_2du_1 \tag{4.18}$$

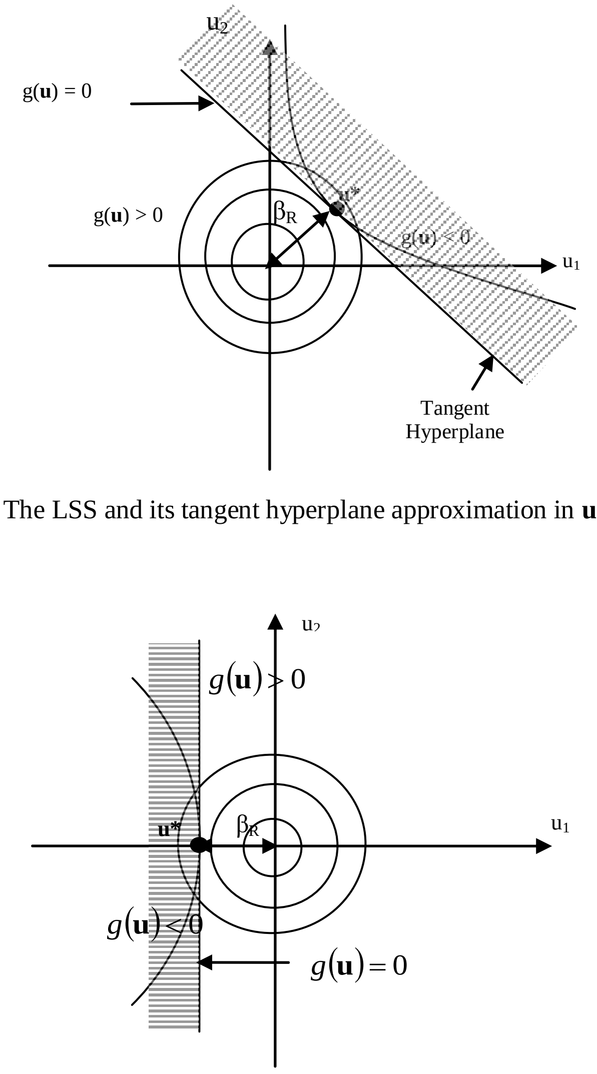
In (4.18), the u_2 term is unity, therefore, the probability of failure is now

$$\Pr(F)=\int_{-\infty}^{-\beta_R}\frac{e^{-\frac{u_1^2}{2}}}{\sqrt{2\pi}}du_1 \tag{4.19}$$

By definition of the normal CDF, the probability of failure is then

$$\Pr(F)=\Pr(g(\mathbf{u})<0)=\Phi(-\beta_R) \tag{4.20}$$

The reliability index, β_R, is just a distance and thus it is always positive.



(a): The LSS and its tangent hyperplane approximation in **u**-space

(b): The LSS rotated to a point perpendicular to the u_1 axis.

Figure 4-2: The limit-state surface in **u**-space.

However, once **u*** has been found, it is not obvious which side of the limit-state surface represents the conformance region and which side represents the failure region. In order to mathematically identify the conformance and non-conformance regions, the sign of the limit-state function g(**u**) at the origin (**u** = 0) is tested. A positive value indicates that the highest probability density point (the origin) is in the conformance region and so Pr(F) = $\Phi(-\beta_R)$.

Conversely, a negative value of g(**u** = 0) indicates that the highest probability density point is in the non-conformance region and so now Pr(F) = Φ(β$_R$). For the ith limit-state function, we let a$_i$ = sign(g$_i$(**u** = 0)) = ±1. We then write

$$\Pr(F_i) = \Phi(-a_i \beta_{R,i}) \tag{4.21}$$

4.5.1 Multiple Limit-State Functions

For a series system, the expression in (4.4) is re-written as

$$\Pr\left(\bigcup_{i=1}^{n_{LF}} F_i\right) = \sum_{i=1}^{n_{LF}} (\Pr(F_i)) - \sum\sum_{i<j}^{n_{LF}} (\Pr(F_i \cap F_j)) + \sum\sum\sum_{i<j<k}^{n_{LF}} (\Pr(F_i \cap F_j \cap F_k)) - \cdots \tag{4.22}$$

One way to simplify (4.22) is to consider at most, pair-wise intersection events. The intersection probability of all pairs of correlated non-conformance events is calculated using the angle between the vectors to MLFPs. To show how intersection probabilities are calculated, consider Figure (4-3) that shows the intersection probability of two arbitrary hyper-plane limit-state surfaces. The two planes, identified as AA and BB, are located at distances β_1 and β_2 from the origin with an angle θ separating the vectors to each MLFP.

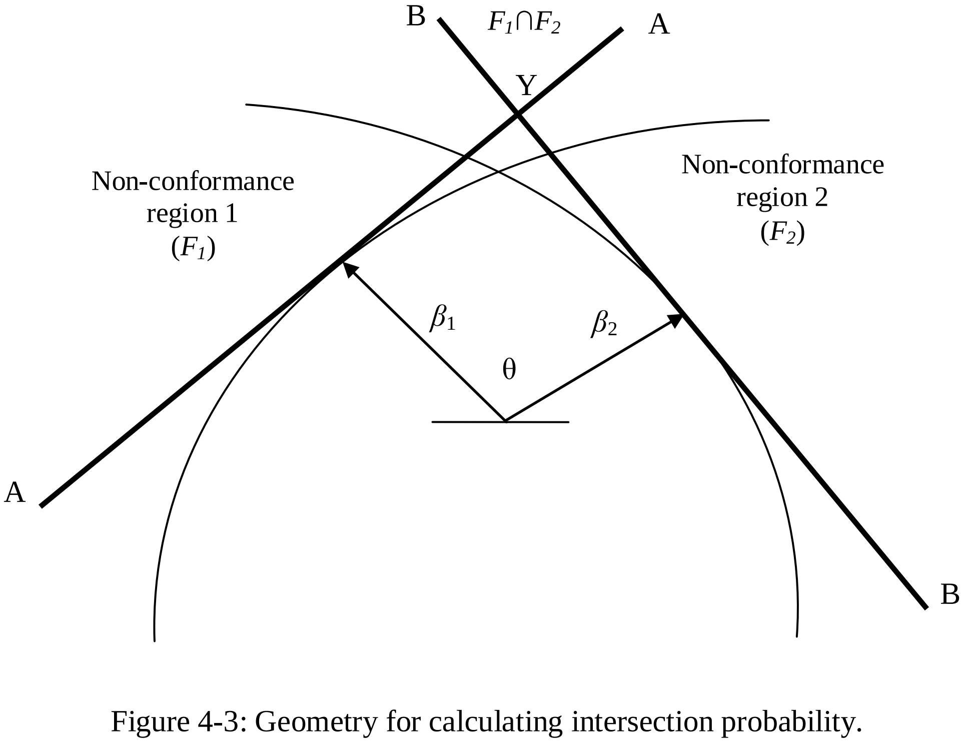

B
$F_1 \cap F_2$
A
Y
Non-conformance region 1 (F_1)
Non-conformance region 2 (F_2)
β_1
β_2
θ
A
B

Figure 4-3: Geometry for calculating intersection probability.

An indication of correlation between planes is ρ = cos(θ) and then the probability in the region AYB is obtained via the bi-variate cumulative function

$$\Pr(F_1 \cap F_2) = \Phi(\beta_1, \beta_2, \rho) = \int_{\beta_1}^{\infty}\int_{\beta_2}^{\infty} \frac{1}{2\pi\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2}\cdot\frac{u_1^2 - 2\rho u_1 u_2 + u_2^2}{1-\rho^2}\right) du_1 du_2 \qquad (4.23)$$

The numerical evaluation of this integral is provided by (Drezner and Wesolowsky 1990) and the MATLAB® files that are used to compute this integral are shown in the appendix (Seshadri 2002).

Consider the case when θ = 0 and ρ = 1 then the failure regions are fully correlated such as g_1 and g_2 in Figure 4-4. For such a case, the system failure is corresponds to the larger failure region where, for the example shown in Figure 4-4, is $g_1(\mathbf{u})$. Consider the case when θ = 180 and ρ = -1 then the failure regions are fully anti-correlated such as g_1 and g_3 in Figure 4-4

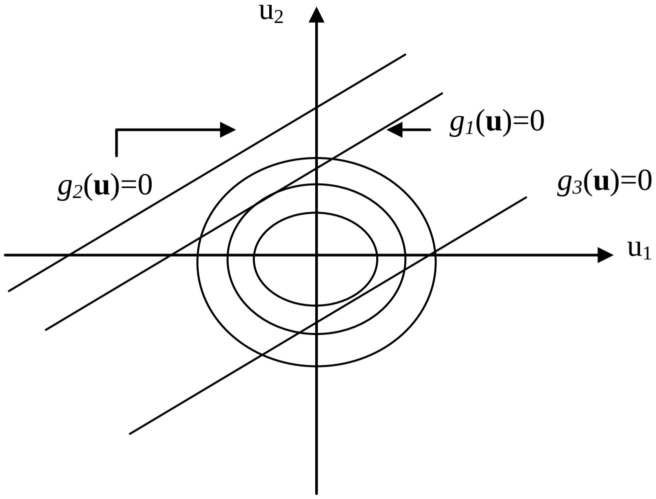


Figure 4-4: An example of fully correlated limit-state surfaces.

4.5.2 An algorithm for Most-Likely Failure Point (MLFP)

The position of the MLFP, **u***, may be found via the constrained optimization problem

$$\begin{aligned}&\underset{\mathbf{u}}{\text{Minimize}}\sqrt{\left(\mathbf{u}^T\mathbf{u}\right)}\\&\text{Subject to}\, g(\mathbf{u})=0\end{aligned} \tag{4.24}$$

A suitable, gradient-based algorithm exists to find **u*** (Madsen, Krenk and Lind 1986). Let *k* be a counter and for *k* = 0, the initial conditions are $\mathbf{u}_0 = 0$ in **u**-space. The steps in the algorithm are

1. Convert $\mathbf{u}_k$ to $\mathbf{v}_k$ using the inverse probability transformation such as the Rosenblatt Transformation.
2. Calculate $g(\mathbf{v}_k)$ and calculate the gradient in **u**-space, evaluated in **v**-space, $\nabla_{\mathbf{u}} g(\mathbf{v}_k)$.

$$\nabla_{\mathbf{u}} g(\mathbf{v})=\frac{\partial g}{\partial \mathbf{u}^T}=\frac{\partial g(\mathbf{v})}{\partial \mathbf{v}^T}\cdot\frac{\partial \mathbf{v}}{\partial \mathbf{u}^T}$$

3. Calculate a new **u**-space location.

$$\mathbf{u}_{k+1}^T=\frac{\mathbf{u}_k^T\nabla_{\mathbf{u}}^T g(\mathbf{v}_k)-g(\mathbf{v}_k)}{\nabla_{\mathbf{u}} g(\mathbf{v}_k)\nabla_{\mathbf{u}}^T g(\mathbf{v}_k)}\nabla_{\mathbf{u}} g(\mathbf{v}_k)$$

4. Calculate the distance from the origin to $\mathbf{u}_{k+1}$ and denote it as δ_{k+1}.
5. Stop when $|\delta_{k+1} - \delta_k| \leq \varepsilon$.

The gradient, $\nabla_{\mathbf{u}} g(\mathbf{v}_i)$, is calculated using the chain rule

$$\nabla_{\mathbf{u}} g(\mathbf{v})=\frac{\partial g}{\partial \mathbf{u}^T}=\frac{\partial g(\mathbf{v})}{\partial \mathbf{v}^T}\frac{\partial \mathbf{v}}{\partial \mathbf{u}^T} \tag{4.25}$$

Instead of using this algorithm, another constraint is to find a plane perpendicular (via the null space vectors) to the outward normal gradient vector; then, at the MLFP the vector **u*** is perpendicular to each of the null space vectors. More specifically, if we have η design variables, there will be η*-1* null space vectors – each of length η – denoted as the matrix $null(\nabla_{\mathbf{u}} g_i(\mathbf{u}))$. An additional feature is that at **u*** we have the orthogonal conditions as η*-1* scalar products

$$\mathbf{u}_i \cdot null(\nabla_{\mathbf{u}} g_i(\mathbf{v}))_j = 0 \quad j = 1,2,\ldots,\eta-1 \tag{4.26}$$

4.6 Parameter Design

In parameter design, we find the means of the design variables, with fixed tolerances, to meet some objective. Suppose we have initial design that we wish to improve. There are three steps involved. The first step is to find the starting MLFPs using (4.24).

The second step is to search for a feasible design such that the sign of each LSF at the origin is positive and the corresponding failure region is small. This step ensures that the probability of failure is small. A weight for each limit-state function is written as

$$W_i = \exp\left(-\alpha_i \beta_{R_i}\right) \tag{4.27}$$

Where $\alpha_i = sign\left(g_i\left(\mathbf{u}=0\right)\right)$. The weight W_i approaches zero for a large β_{R_i} and positive α_i but becomes large for negative products ($\alpha_i \beta_{R_i}$). Further, the exponential changes rapidly for small changes about a small β$_i$. The vector of weights of all limit-state functions is defined as

$$\gamma^T = \left[W_1 \ldots W_\eta\right] \tag{4.28}$$

A useful objective function is obtained by invoking the sum-of-squares type of formulation

$$Q = \frac{1}{2}\gamma^T \gamma = \frac{1}{2}\left[\sum_{i=1}^{n_{LF}} \exp\left(-2a_i \beta_{R_i}\right)\right] \tag{4.29}$$

If all a$_i$ = 0, then the success region contains the origin in **u-**space and the objective function has a lower bound of zero for very large β$_R$. Finally, using (4.29) and the constraint in (4.26), a feasible design is found using

$$\begin{aligned} &\min_{\mu_{\mathbf{V}}} \sum_{i=1}^{n_{LF}} \exp\left(-2\alpha_i \beta_{R_i}\right) \\ &s.t. \quad g_i\left(\mathbf{u}\right) = 0 \\ &\qquad \mathbf{u}_i \cdot null\left(\nabla_{\mathbf{u}} g_i\left(\mathbf{u}\right)\right)_j = 0 \\ &\qquad \mu_{\mathbf{V}}^{\min} \leq \mu_{\mathbf{V}} \leq \mu_{\mathbf{V}}^{\max} \\ &\qquad tol_{\mathbf{V}} \text{ is constant} \end{aligned} \tag{4.30}$$

A third possible step is to minimize the system probability of failure.

$$\begin{aligned}
&\min_{\mu_{\mathbf{V}}} \Pr(\mathbf{F}) \\
&s.t. \quad g_i(\mathbf{u}) = 0 \\
&\qquad \mathbf{u}_i \cdot null(\nabla g(\mathbf{u}_i))_j = 0 \\
&\qquad \mu_{\mathbf{V}}^{\min} \le \mu_{\mathbf{V}} \le \mu_{\mathbf{V}}^{\max} \\
&\qquad tol_{\mathbf{V}} \text{ is constant}
\end{aligned} \tag{4.31}$$

4.7 Tolerance Design

Tolerance design suggests that the mean values of the design variables are fixed and just the tolerances are adjusted. However, in order to minimize the objective functions used in parameter design, the tolerances would go to zero. Since this is not physically or economically feasible, some cost objective is needed to impede this tendency. The cost is typically the production cost (C_p) and it always increases for tighter tolerances.

Typically, the smaller the tolerance the higher the cost since more labour and energy is required to produce a “higher quality” product. This is due to three main factors:

- Cost of machines – Machines capable of high accuracy are manufactured to more exacting standards and are usually more expensive.
- Cost of set-up – Tighter tolerances require more careful and longer set-ups.
- Cost of tooling – Costs of tools and special fixtures rise for tighter tolerances.

There is a variety of cost versus tolerance mathematical models. In general, the two constants denoted as a_c and b_c are cost parameters set by a particular manufacturing process where a_c represents the fixed costs and the cost of means and b_c represents the cost of producing a single component dimension to a specified tolerance (Seshadri and Savage, 2002). The common cost-tolerance models are shown in Table 4-1 (Seshadri and Savage, 2002).

Table 4-1: Cost-tolerance models.

Model Name	**Cost Model (C_p)**
Reciprocal	$a_c + b_c/tol$
Reciprocal Squared	$a_c + b_c/tol^2$
Reciprocal Power	$a_c + b_c/tol^k$
Exponential	$b_c\exp(-m.tol)$
Exponential/Reciprocal Power	be^{-mtol}/tol^k
Piecewise Linear	$a_i - b_i tol_i$

The manufacturing cost, denoted as C_p, is only part of the cost picture. If a product, comprising of many random design variables, is manufactured, some products will meet the specifications assigned to the responses and some will not. The products outside of the limit specifications will be assigned to the “scrap box” or if possible “reworked”. The cost of this “loss of quality” of the unacceptable products is denoted as C_{LQ} and depends upon 1) how many systems are in the non-conforming category and 2) the unit cost, denoted as C_S needed to scrap or rework the product

$$C_{LQ} = \Pr(\mathbf{F}) \times C_s \tag{4.32}$$

In general, the total cost C_T combines the production costs and loss of quality costs and we have

$$C_T = C_P + C_{LQ} \tag{4.33}$$

The best tolerance is allocated as

$$\begin{aligned}
&\text{Minimize } C_T(\mu, tol) \\
&\text{subject to} \\
&\qquad g_i(\mathbf{u}) = 0 \\
&\qquad \mathbf{u}_i \cdot null(\nabla g(\mathbf{u}_i))_j = 0 \\
&\qquad tol_i^{\min} \le tol_i \le tol_i^{\max} \\
&\qquad \mu_{\mathbf{V}} \text{ is constant}
\end{aligned} \tag{4.34}$$

4.8 Integrated Design

So far, we have seen that Parameter design entails moving the means of the design variables while keeping the tolerances constants. Tolerance design holds the means constant while the tolerances are moved. A third option, integrated design, argues that both the means and tolerance values of the design variables should be treated together simultaneously to reach the optimal design (Seshadri and Savage 2002). In general, the integrated robust design problem is posed as a minimum total cost problem in the following form

$$\begin{aligned} & Min \quad C_T = C_P + C_{LQ} \\ & \text{subject to} \\ & \qquad g_i(\mathbf{u}) = 0 \\ & \quad \mathbf{u}_i \cdot null(\nabla g(\mathbf{u}_i))_j = 0 \\ & \quad \mu_i^{\min} \le \mu_i \le \mu_i^{\max} \\ & \quad tol_i^{\min} \le tol_i \le tol_i^{\max} \end{aligned} \tag{4.35}$$

4.9 Summary

A discussion of five main topics; Limit-State Functions, System Reliability, MCS, FORM and Probability-Based Optimization Methods has been presented in this chapter. The limit-state function has been introduced that relates the design specification with the metamodel. The limit-state function is particularly important for probability evaluation since it provides a conceptually simple way of determining when failure or success takes place.

System Reliability concepts has been used to deal with systems with many components. Here, it is assumed that components follow either a series or parallel arrangement. For a series arrangement, system failure occurs if any component fails and for a parallel arrangement, system failure occurs only when all components fail. This chapters has shown how the probability of system failure is estimated using either MCS or FORM.

The MCS generates a large sample of sets that are substituted into the limit-state functions and the number of times the limit-state function goes to zero is counted. FORM evaluates probability by converting the variables to standard normalized variables and the distance from the origin to the MLFP on the LSS is estimated.

Three probabilistic design optimization methods have also been presented; parameter design, tolerance design and integrated design. Parameter design attempts to find the means of the design variables that reduce system failure. If the optimum tolerance is required, a cost function is involved. Therefore, tolerance design attempts to find the tolerance of the design variables that minimizes the total cost. Since tolerance design is not commonly used since design would involve also moving the means of the design variables, integrated design is just an extension to tolerance design that searches for both the mean and tolerance that minimizes the total cost.

Chapter 5

Examples: Probabilistic Design of Static Systems

This chapter will illustrate the metamodel-based design methodology as applied to static systems. The static system has the form of (1.1) and the general metamodel for a static system has been discussed in Chapter 3. The methodology will be used for both parameter and integrated design and the results obtained from the three metamodels will be compared. Metamodel accuracy will be compared using cross-validation methods and comparing the metamodel-based FORM estimate of failure probability with both the metamodel-based and mechanistic model-based MCS methods.

5.1 Example 1 – Simple Servo

Consider a simple servo system where the rotational steady-state shaft speed is a function of the voltage E, the motor's torque constant K, and the electrical power amplification G. The design variables are *K* and *E*, v_1 and v_2 respectively, with *G* deterministic. The steady-state shaft speed, *z* has the function

$$z(\mathbf{v}) = G\frac{v_2}{v_1} \tag{5.1}$$

where G = 2. The variables follow a normal distribution with parameters shown in Table 5-1.

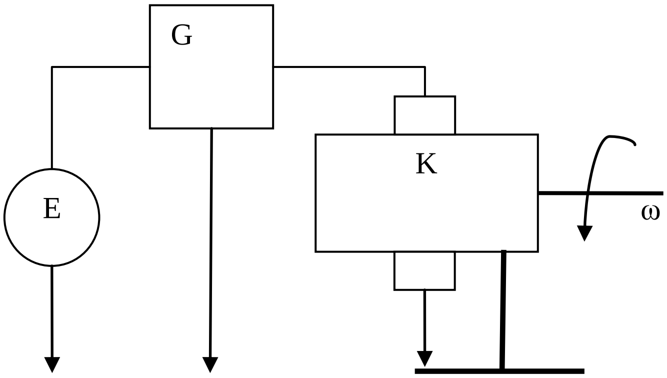


Figure 5-1: Schematic of a simple electro-mechanical servo motor.

The distribution parameters of v_1 and v_2 are shown in Table 5-1.

Table 5-1: The distribution and corresponding parameters of each design variable.

Design Variable	**Distribution**	**mean**	**Units**	**Standard Deviation**
v_1 – Torque Constant (K)	Normal	$0.0079 \leq \mu_1 \leq 0.0081$	Nm/A	$\sigma_1 = \left(\frac{2\mu_1}{300}\right)$
v_2 – Applied Voltage (E)	Normal	$0.9 \leq \mu_2 \leq 1.1$	V	$\sigma_2 = \left(\frac{1.5\mu_2}{300}\right)$

In order to fit the metamodels, the training design range is chosen to be wider than the range of the means of the design variables. This is to compensate for the event that the final design is one of the upper or lower limit boundaries. To develop the training design, the three levels of each design variable are shown in Table 5-2.

Table 5-2: The low, nominal and high values of each design variable.

Variable	v_1	v_2
Low	0.0078	0.8
Nominal	0.0080	1.0
High	0.0082	1.2

Since there are 3 levels (λ = 3) for 2 design variables (η = 2), the total number of training sets, M, is $M = \lambda^{\eta} = 9$. The nine unique training sets are recorded into a matrix **X**

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \mathbf{x}_3^T \\ \vdots \\ \mathbf{x}_9^T \end{bmatrix}_{9\times 2} = \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \\ x_{3,1} & x_{3,2} \\ \vdots & \vdots \\ x_{9,1} & x_{9,2} \end{bmatrix}_{9\times 2}$$

The corresponding training responses are found by substituting **X** into (5.1) to get

$$\mathbf{z}(\mathbf{X}) = \begin{bmatrix} z_1\left(\mathbf{x}_1^T\right) \\ z_2\left(\mathbf{x}_2^T\right) \\ z_3\left(\mathbf{x}_3^T\right) \\ \vdots \\ z_9\left(\mathbf{x}_9^T\right) \end{bmatrix}_{9\times 1}$$

For a static system, the general metamodel looks like

$$\hat{z}(\mathbf{v}) = \mathbf{w}^T \mathbf{r}(\mathbf{v}) \tag{5.2}$$

Regression Model

A two-variable Regression Model has already been discussed in Chapter 3 where

$$\mathbf{w}^T = \begin{bmatrix} \beta_0 & \beta_1 & \beta_2 & \beta_1^2 & \beta_2^2 & \beta_1\beta_2 \end{bmatrix}_{1\times 6}$$

$$\mathbf{r}(\mathbf{v}) = \begin{bmatrix} 1 & v_1 & v_2 & v_1^2 & v_2^2 & v_1 v_2 \end{bmatrix}_{6\times 1}^T$$

In order to estimate the parameters in $\mathbf{w}^T$, a matrix $\mathbf{X}_{\text{a}}$ must first be developed from the training design **X** that is substituted into (3.26). Following (3.25), $\mathbf{X}_{\text{a}}$ looks like

$$\mathbf{X}_a = \begin{bmatrix} 1 & x_{1,1} & x_{2,1} & x_{1,1}^2 & x_{2,1}^2 & x_{1,1}x_{2,1} \\ 1 & x_{1,2} & x_{2,2} & x_{1,1}^2 & x_{2,2}^2 & x_{1,2}x_{2,2} \\ 1 & x_{1,3} & x_{2,3} & x_{1,3}^2 & x_{2,3}^2 & x_{1,3}x_{2,3} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{1,9} & x_{2,9} & x_{1,9}^2 & x_{2,9}^2 & x_{1,9}x_{2,9} \end{bmatrix}_{9\times 6}$$

Kriging Model and RBF

The Kriging and RBF 2-variable models have been presented in Chapter 3 and follow the same format for this example.

The accuracy of each metamodel is checked using the techniques CV-RMSE and CV-MAE. These values are shown in Table 5-3.

Table 5-3: The CV-RMSE and CV-MAE of each metamodel for the simple servo example.

Model	CV-RMSE	CV-MAE
RM	4.014×10^{-3}	9.813×10^{-3}
Kriging	4.028×10^{-4}	3.714×10^{-4}
RBF	5.102×10^{-5}	1.376×10^{-4}

Smaller values of CV-RMSE and CV-MAE are better since these are error calculations. From Table 5-3, all three metamodels provide excellent fits of the training design as seen by the very low CV-RMSE and CV-MAE values. Among the three metamodels, the RBF is best. All three metamodels will be used with FORM to search for the final design.

Our goal is to find the means of the design variables that result in a reduced failure probability. For this servo, the upper and lower design specifications are 255 rad/s and 245 rad/s respectively. Therefore, the two LSFs become

$$g_1(\mathbf{v}) = 255 - z(\mathbf{v}) \tag{5.3}$$

$$g_2(\mathbf{v}) = z(\mathbf{v}) - 245 \tag{5.4}$$

Since the variables are assumed to be Normal distributed and uncorrelated, following example 1 in Chapter 4, the transformation to **u**-space looks like

$$\Gamma(\mathbf{u}, \mathbf{v}, \mu_i, tol_i) = \begin{bmatrix} v_1 - \mu_1\left(1 + \dfrac{tol_1 u_1}{300}\right) \\ v_2 - \mu_2\left(1 + \dfrac{tol_2 u_2}{300}\right) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

The first step is to search for a feasible design using (4.30) and then using this feasible design as a starting point, we search for the minimum system failure calculated using (4.31). Since the limit-state surfaces do not intersect, the optimization problem is formulated as shown in (5.5).

$$\begin{aligned}
&\min \quad \Pr(\mathbf{F}) = \Pr(g_1(\mathbf{V})<0) + \Pr(g_2(\mathbf{V})<0) \\
&\text{Subject to:} \\
&\qquad \mathbf{u}_1 \cdot null(\nabla g_1(\mathbf{u}))_1 = 0 \\
&\qquad \mathbf{u}_2 \cdot null(\nabla g_2(\mathbf{u}))_1 = 0 \\
&\qquad \begin{bmatrix} 0.0078 \\ 0.9 \end{bmatrix} \le \mu_{\mathbf{V}} \le \begin{bmatrix} 0.0081 \\ 1.1 \end{bmatrix}
\end{aligned} \tag{5.5}$$

The constraints ensure that the MLFP is found and the means fall within the specified boundaries.

Since the mechanistic model is very simple, the best design found using each metamodel is compared with the design found using the mechanistic model. In Table 5-4, the MLFP at each LSS is shown along with the corresponding reliability index. The design found using each metamodel is very close to that obtained when the mechanistic model is used. Following the metamodel validation results shown in Table 5-3 where the RBF provided the best fit, the MLFP estimated using the RBF is closest to the MLFP found using the mechanistic model. This result is also seen when comparing the reliability indices.

Table 5-4: The MLFP, and the corresponding reliability index, at the best design.

Model	**Best Design**	**MLFP and β_R**	
Mechanistic	$\mu_{\mathbf{V}} = \begin{bmatrix} 0.008000 \\ 0.9997 \end{bmatrix}$	$\mathbf{u}_1^* = \begin{bmatrix} -1.944 \\ 1.429 \end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix} 1.896 \\ -1.450 \end{bmatrix}$
		$\beta_{R_1} = 2.413$	$\beta_{R_2} = 2.387$
RM	$\mu_{\mathbf{V}} = \begin{bmatrix} 0.008001 \\ 0.9999 \end{bmatrix}$	$\mathbf{u}_1^* = \begin{bmatrix} -1.929 \\ 1.418 \end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix} 1.907 \\ -1.460 \end{bmatrix}$
		$\beta_{R_1} = 2.394$	$\beta_{R_2} = 2.402$
Kriging	$\mu_{\mathbf{V}} = \begin{bmatrix} 0.008001 \\ 0.9999 \end{bmatrix}$	$\mathbf{u}_1^* = \begin{bmatrix} -1.929 \\ 1.419 \end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix} 1.910 \\ -1.462 \end{bmatrix}$
		$\beta_{R_1} = 2.395$	$\beta_{R_2} = 2.405$
RBF	$\mu_{\mathbf{V}} = \begin{bmatrix} 0.008006 \\ 1.001 \end{bmatrix}$	$\mathbf{u}_1^* = \begin{bmatrix} -1.929 \\ 1.417 \end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix} 1.912 \\ -1.464 \end{bmatrix}$
		$\beta_{R_1} = 2.393$	$\beta_{R_2} = 2.408$

Table 5-5 compares the failure probabilities calculated using FORM and a MCS where $\Pr(F_1) = \Pr(g_1(\mathbf{V}) < 0)$ and $\Pr(g_2(\mathbf{V}) < 0)$. For MCS, 500,000 sample sets have been used to estimate probability. For the RM and Kriging, the FORM estimate of failure probability is very close to that found using the MCS. For the RBF, however, the error is larger. An interesting point, however, is that the when MCS is used with the RBF to estimate probability, the failure probability is closer to the mechanistic model results than either RM or Kriging. This result coincides with Table 5-3 that shows the RBF has the lowest predictive error.

Finally, the limit-state surfaces at the best design are plotted and are shown in Figure 5-2. The position of the LSSs is one of the special cases discussed in Chapter 4 where θ = 180 and ρ = -1, therefore the intersection is zero and the system failure is just the sum of the two failure regions. From Figure 5-2, it is clear that the design is also "balanced" since the distance of the LSSs from the origin is approximately equal. This point is also noticed in Table 5-4 where the reliability indices corresponding to each LSS is almost equal.

Table 5-5: The best design found using the model identified in the first column with FORM.

Model	**Best Design**	**FORM**			**MCS**		
		$\Pr(F_1)$	$\Pr(F_2)$	**CPU Time**	$\Pr(F_1)$	$\Pr(F_2)$	**CPU Time**
Mechanistic	$\mu_\mathbf{V} = \begin{bmatrix} 0.008000 \\ 0.9997 \end{bmatrix}$	0.007912	0.008490	< 2s	0.007778	0.008606	< 1s
RM	$\mu_\mathbf{V} = \begin{bmatrix} 0.008001 \\ 0.9999 \end{bmatrix}$	0.008323	0.008159	< 2s	0.008505	0.008030	< 1s
Kriging	$\mu_\mathbf{V} = \begin{bmatrix} 0.008001 \\ 0.9999 \end{bmatrix}$	0.008285	0.008133	~ 3s	0.008404	0.008118	~ 4.5s
RBF	$\mu_\mathbf{V} = \begin{bmatrix} 0.008006 \\ 1.001 \end{bmatrix}$	0.008400	0.008000	~ 8s	0.007830	0.008320	~ 4s

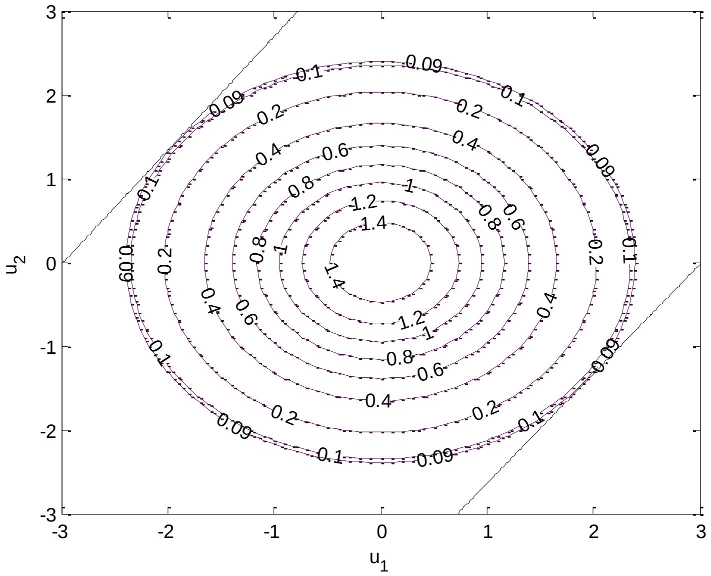


Figure 5-2: The two limit-state surface (in **U**-space) and the joint Normal PDF contours.

Summary

This example has found the means of the design variables that minimizes the system failure. The results found using the metamodel approximation of the LSF has been compared to those found using the original LSF. All metamodels have been able to estimate the location of the MLFP very accurately. The accuracy is checked by comparing the results from the metamodels with those obtained using the mechanistic model. When FORM is used to search for the best design, the results obtained from the RM and Kriging are better than those from the RBF. However, when the MCS is used to check the failure probability at the best design, the RBF has been found to be more accurate. In terms of speed, using FORM with the RM has been found to be the fastest but slowest with the RBF. For MCS, the RM is still the fastest but now the RBF is just about as fast as Kriging.

5.2 Example 2 - Thin Film Layer

The electrical impedance of a film is given explicitly as a function of three variables v_1, v_2 and v_3 through the equation (Bagchi and Templeton 1994)

$$z(\mathbf{v}) = \frac{87}{\sqrt{\varepsilon + \sqrt{2}}} \ln\left(\frac{5.98 v_1}{0.8 v_2 + v_3}\right) \tag{5.6}$$

The dielectric constant of the material, ε, has the value ε = 3.094539. Previous work done with this example assumes that the variables are normally and independently distributed with means and tolerances as shown in Table 5-6.

Table 5-6: The distribution parameters of each variable.

Design Variable	Distribution	Parameters	Standard Deviation
V_1 – Insulator Thickness	Normal	$\mu_1 = 26.6$	$\sigma_1 = 0.3333$
V_2 – Conductor Line Width	Normal	$\mu_2 = 17.5$	$\sigma_2 = 0.2222$
V_3 – Line Height	Normal	$\mu_3 = 6$	$\sigma_3 = 0.1111$

Using the relation $\mu_i \pm 6\sigma_i$, three levels of each design variable are obtained, as shown in Table 5-7, and a full 3^3 factorial design is used to generate the training sets. In the event that the final design may fall on the boundary constraints of the mean of each design variable, a training design range is chosen to be wider than the 3σ design limits to ensure that the metamodel fits outside of the design range.

Table 5-7: The low, nominal and high values of each design variable.

Variable	x_1	x_2	x_3
Low	24.6	16.17	5.33
Nominal	26.6	17.5	6
High	28.6	18.83	6.67

A sample of the training sets is shown in (5.7).

$$\mathbf{X} = \begin{bmatrix} x_{1,1} & x_{1,2} & x_{1,3} \\ x_{2,1} & x_{2,2} & x_{2,3} \\ \vdots & \vdots & \vdots \\ x_{26,1} & x_{26,2} & x_{26,3} \\ x_{27,1} & x_{27,2} & x_{27,3} \end{bmatrix}_{27\times 3} = \begin{bmatrix} 24.6 & 16.17 & 5.33 \\ 24.6 & 16.17 & 6 \\ \vdots & \vdots & \vdots \\ 28.6 & 18.83 & 6 \\ 28.6 & 18.83 & 6.67 \end{bmatrix}_{27\times 3} \tag{5.7}$$

Similar to example 1, the fitting parameters of each metamodel are estimated and CV-RMSE and CV-MAE is estimate for each metamodel. The results are shown in Table 5-8. All three metamodels again provide good fits of the training design. In this case, however, the best fit is obtained using the RM.

Table 5-8: The CV-RMSE and CV-MAE estimates of each metamodel.

Model	**CV-RMSE**	**CV-MAE**
RM	1.053×10^{-4}	7.272×10^{-4}
Kriging	2.67×10^{-4}	2.39×10^{-3}
RBF	1.849×10^{-3}	1.678×10^{-2}

For the electrical impedance of the thin-film layer introduced in the previous chapter, the upper and lower specifications are 86Ω and 84Ω respectively. Based on these specifications, the limit-state functions are

$$g_1(\mathbf{v}) = 86 - \hat{z}(\mathbf{v}) \tag{5.8}$$

$$g_2(\mathbf{v}) = \hat{z}(\mathbf{v}) - 84 \tag{5.9}$$

Variables are transformed according to the transformation in the previous example. Therefore, **V** is a function of the mean ($\mu_{\mathbf{V}}$) and standard deviation ($\sigma_{\mathbf{V}}$).

This problem is similar to (5.5)

$$\begin{aligned} &\min \quad \Pr(g_1(\mathbf{V}) < 0) + \Pr(g_2(\mathbf{V}) < 0) \\ &\text{Subject to:} \\ &\qquad \mathbf{u}_1 \cdot null(\nabla g_1(\mathbf{u})) = 0 \\ &\qquad \mathbf{u}_2 \cdot null(\nabla g_2(\mathbf{u})) = 0 \\ &\mu_{\mathbf{V}} - 3\sigma_{\mathbf{V}} \le \mu_{\mathbf{V}} \le \mu_{\mathbf{V}} + 3\sigma_{\mathbf{V}} \end{aligned} \tag{5.10}$$

The best designs are shown in Table 5-9 along with the corresponding MLFPs and the reliability indices.

The best design found using the mechanistic model is assumed to be the most accurate. From table 5-9, the closest design to the mechanistic model has been found using the RM followed by Kriging and then the RBF. A comparison of the MLFP and reliability indices also shows the closest estimates have been found using the RM.

Table 5-9: The MLFP and reliability index at the best design found using each metamodel.

Model	**Best Design**	**MLFP**		β_{R_1}	β_{R_2}
Mechanistic	$\mu_{\mathbf{v}} = \begin{bmatrix} 27.60 \\ 18.00 \\ 6.329 \end{bmatrix}$	$\mathbf{u}_1^* = \begin{bmatrix} 1.184 \\ -0.8643 \\ -0.5211 \end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix} -1.190 \\ 0.8299 \\ 0.5229 \end{bmatrix}$	1.556	1.542
RM	$\mu_{\mathbf{v}} = \begin{bmatrix} 27.60 \\ 18.01 \\ 6.328 \end{bmatrix}$	$\mathbf{u}_1^* = \begin{bmatrix} 1.185 \\ -0.8665 \\ -0.5219 \end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix} -1.191 \\ 0.8295 \\ 0.5226 \end{bmatrix}$	1.558	1.543
Kriging	$\mu_{\mathbf{v}} = \begin{bmatrix} 27.60 \\ 18.00 \\ 6.330 \end{bmatrix}$	$\mathbf{u}_1^* = \begin{bmatrix} 1.183 \\ -0.8519 \\ -0.5244 \end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix} -1.188 \\ 0.8319 \\ 0.5198 \end{bmatrix}$	1.553	1.544
RBF	$\mu_{\mathbf{v}} = \begin{bmatrix} 27.6 \\ 18.02 \\ 6.323 \end{bmatrix}$	$\mathbf{u}_1^* = \begin{bmatrix} 1.180 \\ -0.8670 \\ -0.5328 \end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix} -1.193 \\ 0.8275 \\ 0.5092 \end{bmatrix}$	1.558	1.538

Table 5-10 shows the probability of failure estimates using FORM and MCS and the corresponding CPU times. For MCS, 100,000 runs have been used. Similar to the previous example, the fastest times have been observed with the RM. When FORM is used to evaluate probability, the RBF is slowest but becomes a lot faster when MCS is used.

Table 5-10: The failure probability at the best design.

Model	**Best Design** $\mu_{\mathbf{V}}^{best}$	**FORM**			**MCS**		
		Pr(F_1)	Pr(F_2)	**CPU Time**	Pr(F_1)	Pr(F_2)	**CPU Time**
Mechanistic	$\mu_{\mathbf{v}} = \begin{bmatrix} 27.60 \\ 18.00 \\ 6.329 \end{bmatrix}$	0.05986	0.06158	2s	0.05957	0.06097	< 1s
RM	$\mu_{\mathbf{v}} = \begin{bmatrix} 27.60 \\ 18.01 \\ 6.328 \end{bmatrix}$	0.05965	0.06147	2s	0.05865	0.06144	< 1s
Kriging	$\mu_{\mathbf{v}} = \begin{bmatrix} 27.6 \\ 18.00 \\ 6.330 \end{bmatrix}$	0.06070	0.06172	6s	0.06024	0.06166	~ 1.5s
RBF	$\mu_{\mathbf{v}} = \begin{bmatrix} 27.60 \\ 18.02 \\ 6.323 \end{bmatrix}$	0.0597	0.0620	15s	0.0593	0.0622	~ 1s

Summary

Metamodels have been used to search for the means of the design variables that minimize the system failure. The designs found are the best that meet boundaries placed upon the means of the design variables. From Table 5-8, the CV-RMSE and CV-MAE estimates indicate that the best accuracy comes from the RM. These results coincide with those of Table 5-9 that shows the estimate of the MLFP found using each metamodel. From Table 5-8, the RM model has been found to be the best in terms of both speed and accuracy. The speed of Kriging and the RBF is very slow with FORM but must faster when the MCS is used to evaluate probability.

5.3 Example 3 - Voltage Divider

Consider the voltage divider circuit shown in Figure 5-3 taken from (Seshadri and Savage 2002).

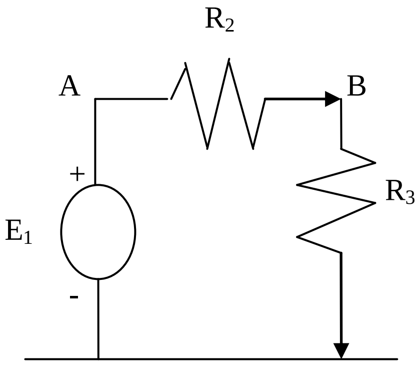


Figure 5-3: A Voltage Divider Circuit.

The potential source is a constant at E_1 = 5V (deterministic variable) and the two impedances R_2 and R_3 are the design variables (denoted as v_1 and v_2) each with a normal distribution $\left[\mu_i, \sigma_i^2 = \left({}^{tol\%}\!/_{300}\mu_i\right)^2\right]$. The potential at B and the flow in the network are the responses of interest. The target potential at B is 2.5 units and the target flow is 0.0025 units. The two responses are functions of the deterministic and design variables whose mechanistic models are shown in (5.11) and (5.12) respectively

$$z_1(\mathbf{v}) = \frac{v_2 E_1}{v_1 + v_2} \tag{5.11}$$

$$z_2(\mathbf{v}) = \frac{E_1}{v_1 + v_2} \tag{5.12}$$

For the design variables, the correlation coefficient is ρ = 0. The design parameters are the means and tolerances of the two impedance distributions and their limits are shown below

$$\begin{gathered} 900 \le \mu_i \le 1100 \\ 5 \le tol_i \le 10 \end{gathered} \tag{5.13}$$

Using the boundary of the mean, three levels of each design variable have been obtained and used to develop the training design based upon a 3^2 factorial design. The training design is then substituted into (5.11) and (5.12) to find the corresponding training responses for voltage and current. The metamodel parameters are estimated and the CV-RMSE and CV-MAE estimates are shown in Table 5-11. According to Table 5-11, the smallest predictive error comes from the RBF.

Table 5-11: The CV-RMSE and CV-MAE estimates for each metamodel.

Model	**Voltage**		**Current**	
	CV-RMSE	**CV-MAE**	**CV-RMSE**	**CV-MAE**
RM	4.48×10^{-4}	1.98×10^{-4}	6.017×10^{-4}	1.379×10^{-3}
Kriging	2.96×10^{-4}	1.84×10^{-4}	5.60×10^{-4}	8.05×10^{-4}
RBF	2.534×10^{-5}	7.035×10^{-5}	9.774×10^{-5}	2.256×10^{-4}

The upper and lower specifications for both responses are ± 4% of target giving rise to four limit-state functions

$$g_1(\mathbf{v}) = 2.6 - z_1(\mathbf{v}) \tag{5.14}$$

$$g_2(\mathbf{v}) = z_1(\mathbf{v}) - 2.4 \tag{5.15}$$

$$g_3(\mathbf{v}) = 0.0026 - z_2(\mathbf{v}) \tag{5.16}$$

$$g_4(\mathbf{v}) = z_2(\mathbf{v}) - 0.0024 \tag{5.17}$$

Figure 5-4 shows a plot of the LSSs at the original design. The LSS corresponding to g_4 is not shown since it falls outside of the plot area. Although g_1 and g_2 seem to be acceptable, g_3 is not since it lies very close to the origin. The first step is, therefore, to search for a feasible design using (4.30).

This problem is an integrated design problem, therefore, after finding the feasible design, the best design is found by minimizing the total cost subject to the constraints shown in (5.18).

$$\begin{aligned} &\min \quad C_T = C_P + C_{LQ} \\ &\text{Subject to:} \\ &\qquad \mathbf{u}_i \cdot null(\nabla_{\mathbf{u}} g_i(\mathbf{u}))_j = 0 \\ &\qquad 900 \le \mu_{\mathbf{V}} \le 1000 \\ &\qquad 5 \le \mathrm{tol}_{\mathbf{V}} \le 10 \end{aligned} \tag{5.18}$$

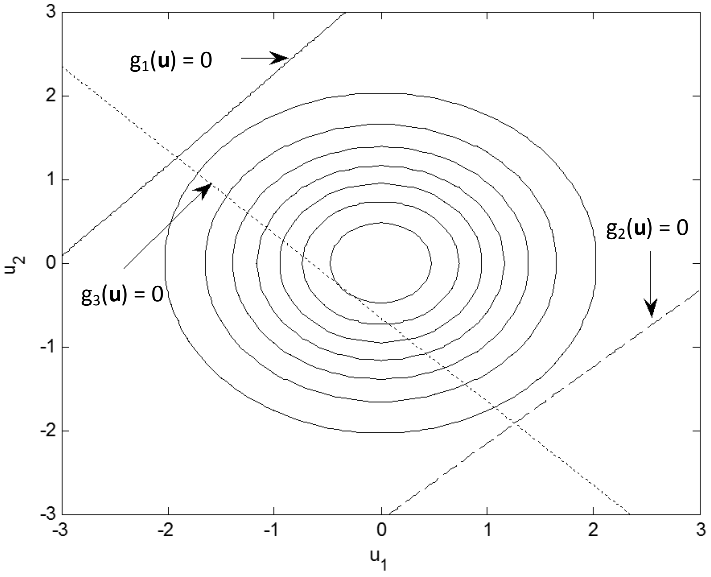


Figure 5-4: The limit-state surfaces at the initial design.

The total cost is the sum of the production and scrap costs. For the production cost, the two impedances are assumed to have reciprocal cost relations of the form, $\text{cost}_i = b_i/\text{tol}_i$. The cost of disposing the non-conforming product is C_s. The cost function for n_c circuits is then

$$C_T = \left(\frac{b_1}{tol_1} + \frac{b_2}{tol_2} + \Pr\{F(\mu_i, tol_i)\} * C_s \right) \times n_c \tag{5.19}$$

For this study, it is assumed that n_c = 1000, b_i = 0.3 and C_s = \$0.6. The optimal design parameters are shown in Table 5-12.

Table 5-12: The optimal parameters, found using each model, along with the corresponding conformance probability and cost.

Model	**Optimal Parameters**		**Probability of Conformance**	**Cost**
	Means	**Tolerances**		
Mechanistic	$\mu_V = [1001.4 \quad 1001.4]^T$	$\mathbf{tol}_V = [6.044 \quad 6.044]^T$	0.99	105.3
RM	$\mu_V = [1001.2 \quad 1001.4]^T$	$\mathbf{tol}_V = [6.023 \quad 6.023]^T$	0.99	105.6
Kriging	$\mu_V = [1001.5 \quad 1001.5]^T$	$\mathbf{tol}_V = [6.041 \quad 6.041]^T$	0.99	105.3
RBF	$\mu_V = [1001.4 \quad 1001.4]^T$	$\mathbf{tol}_V = [6.017 \quad 6.018]^T$	0.99	105.7

Summary

The optimal parameters found using each metamodel are similar. However, the design found using Kriging is closest to the results from the mechanistic model. Figure 5-5 shows the LSSs at the best design. A comparison of Figure 5-5 with Figure 5-4 shows a more balanced design than the original design since all LSSs are about the same distance from the origin.

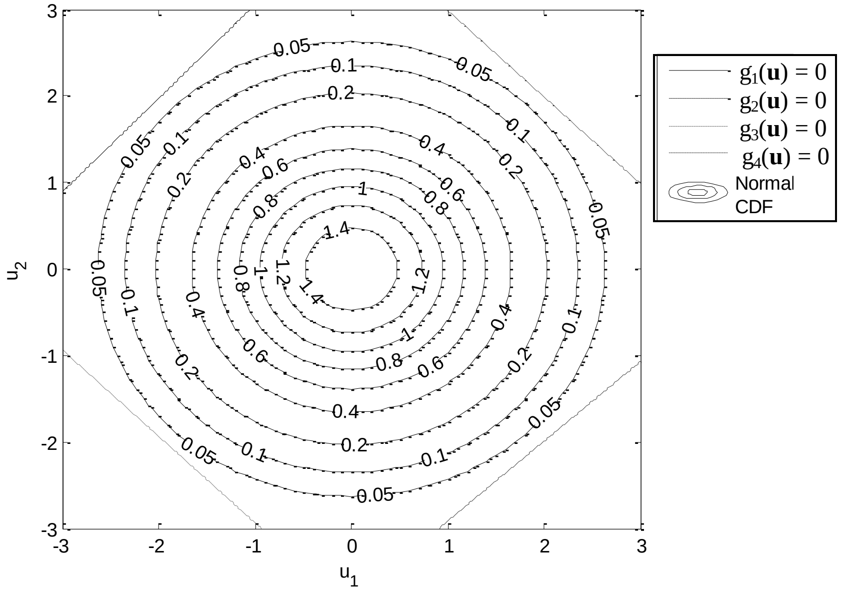


Figure 5-5: The limit-state surfaces at the best design.

5.4 Example 4 – Overrun Clutch Assembly

An automotive overrun clutch (Seshadri and Savage, 2002 and Son, 2006), shown in Figure (5-6), comprises four different parts: one hub, one cage, four rollers and four springs. The springs push the rollers out to remain in contact with both the cage and the hub. If the hub is turned counter-clockwise, relative to the cage, the rollers bind, causing the cage to turn with the hub. If the hub turns clockwise, relative to the cage, the rollers slip and there is no torque transmission. The contact angle, shown as Z, is important for proper operation. If the value of the angle is greater than the upper specification limit or less than the lower specification limit, the clutch does not work correctly and it must be reworked or scrapped.

The overrun clutch, shown in Figure 5-6, comprises three design variables v_1, v_2, and v_3 associated with the dimensions of the cage, the hub and the rollers.

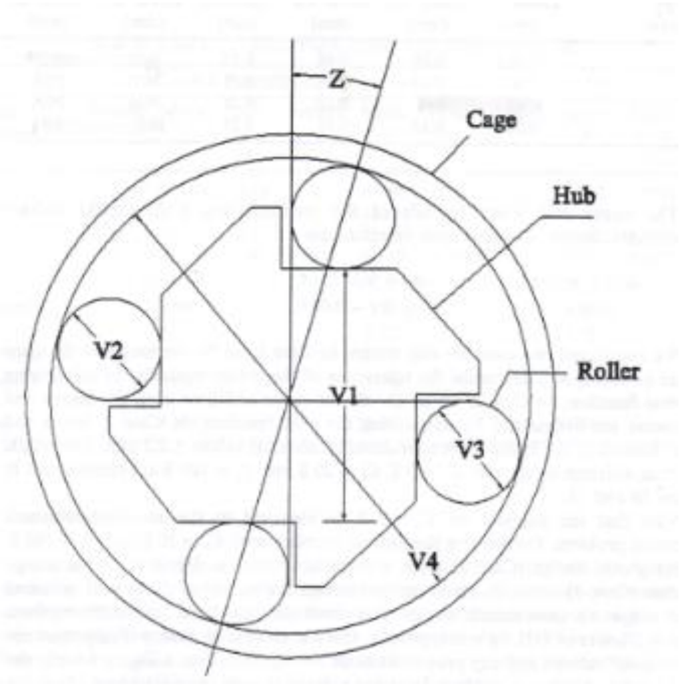


Figure 5-6: An overrun clutch assembly (Source: Seshadri and Savage, 2002).

Each of the design variables is assumed to be normal and the nominal values of the design variables are 55.29mm, 22.86mm, and 101.69mm. The response z is the angle given by the function

$$z(\mathbf{v}) = \cos^{-1}\left(\frac{v_1 + v_2}{v_3 - v_2}\right) \tag{5.20}$$

The angle has a target-is-best performance measure with upper and lower specification, 0.122±0.035rad respectively. These two design specifications lead to two LSFs shown in (5.21).

$$\begin{aligned} g_1(\mathbf{v}) &= 0.157 - z(\mathbf{v}) \\ g_2(\mathbf{v}) &= z(\mathbf{v}) - 0.087 \end{aligned} \tag{5.21}$$

From Son, 2006, the upper and lower bounds of the means are considered as nominal values ± 0.2mm. Using this relation, three levels of each design variables are obtained as shown in Table 5-13. A full factorial design is used to develop the training design where the 3 levels of each design variable are shown in Table 5-13.

Table 5-13: The three levels of each design variable.

Levels	x_1	x_2	x_3
Lower	55.09	22.66	101.49
Nominal	55.29	22.86	101.69
Upper	55.49	23.06	101.89

The parameters of each metamodel are estimated and the predictive error of each is shown in Table 5-14.

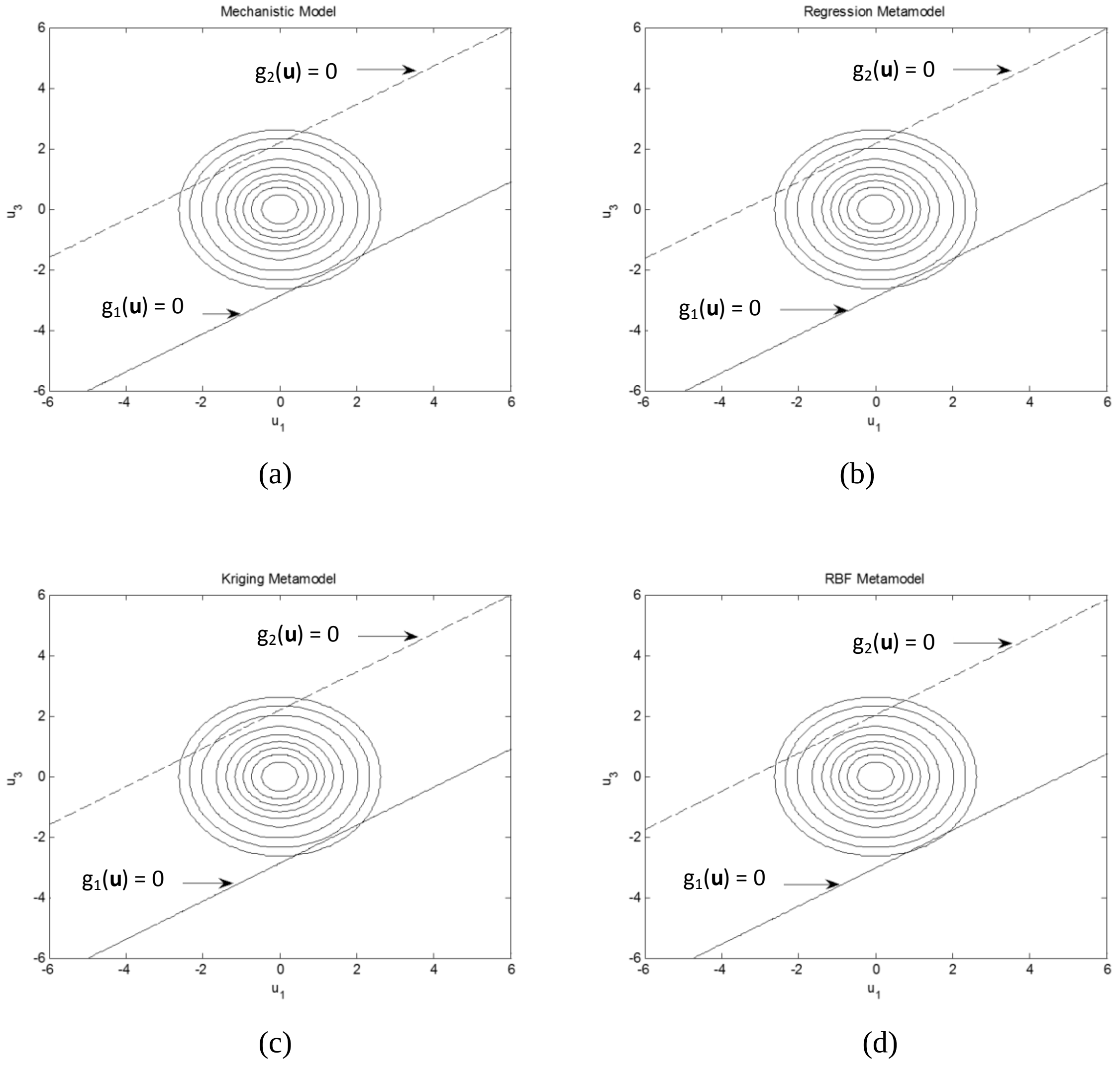
Table 5-14: The error estimates of the fit of the metamodels.

Model	**CV-RMSE**	**CV-MAE**
RM	3.657×10^{-6}	2.599×10^{-5}
Kriging	2.159×10^{-9}	4.514×10^{-9}
RBF	2.395×10^{-6}	4.380×10^{-5}

From Table 5-14, all metamodels give very small estimates of CV-RMSE and CVMAE. The best metamodel in this case is Kriging.

Holding v_2 constant at the mean value, the limit-state surfaces are plotted (Figure 5-7) for each model-type estimate of the LSF. All four figures show linear limit-state surfaces. Also, the metamodel approximation of the limit-state surfaces is very similar to the mechanistic model limit-state surface. This shows the accuracy of the metamodels. Now, FORM is used to check the failure probability at the initial design using each model.

From Table 5-15, the MLFP and reliability index estimates found using the mechanistic, regression and Kriging models are close. Also, the number of iterations required to converge to the MLFP is the same. For the RBF, however, the position of the MLFP found is not as accurate as those obtained using the other two metamodels. Also, for the RBF, the number of iterations required to converge to the MLFP is larger than for the RM or Kriging. The probability of failure and CPU time comparison is shown in Table (5-16).



(a)

(b)

(c)

(d)

Figure 5-7: Plot of the limit-state surfaces. (a) Mechanistic Model, (b) Regression Model, (c) Kriging and (d) RBF.

Table 5-15: The MLFP estimates, from each model, at the original design.

Model	FORM		Iterations
	MLFP and β_R		
Mechanistic	$\mathbf{u}_1^* = \begin{bmatrix} 0.4920 \\ 1.179 \\ -0.7842 \end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix} -0.3783 \\ -0.9023 \\ 0.5978 \end{bmatrix}$	3
	$\beta_1 = 1.499$	$\beta_2 = 1.147$	
RM	$\mathbf{u}_1^* = \begin{bmatrix} 0.4912 \\ 1.177 \\ -0.7829 \end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix} -0.3791 \\ -0.9043 \\ 0.5991 \end{bmatrix}$	3
	$\beta_1 = 1.496$	$\beta_2 = 1.149$	
Kriging	$\mathbf{u}_1^* = \begin{bmatrix} 0.4920 \\ 1.178 \\ -0.7842 \end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix} -0.3783 \\ -0.9024 \\ 0.5978 \end{bmatrix}$	3
	$\beta_1 = 1.498$	$\beta_2 = 1.147$	
RBF	$\mathbf{u}_1^* = \begin{bmatrix} 0.4950 \\ 1.185 \\ -0.7804 \end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix} -0.3730 \\ -0.8838 \\ 0.5930 \end{bmatrix}$	15
	$\beta_1 = 1.503$	$\beta_2 = 1.128$	

From Table 5-16, when FORM is used to estimate probability, the best estimates are found using either the RM or Kriging. The same conclusion can be drawn when MCS is used for probability evaluation. In terms of CPU time, the fastest time is observed when FORM is used with the RM and the slowest from the RBF. For MCS, the RBF is not as slow.

Table 5-16: The probability of failure estimates at the original design.

Method	$\Pr(g_1(\mathbf{U}) < 0)$	$\Pr(g_2(\mathbf{U}) < 0)$	**CPU Time**
Mechanistic and FORM	0.0670	0.1258	2s
Mechanistic and MCS (50,000 runs)	0.06693	0.1242	1s
Metamodel FORM			
Regression	0.0673	0.1253	2s
Kriging	0.0671	0.1257	7s
RBF	0.0665	0.1297	36s
Metamodel and MCS (50,000 runs)			
RM	0.06614	0.1242	1s
Kriging	0.0670	0.1261	2s
RBF	0.0685	0.1284	2s

This problem aims to reduce the total cost which is the sum of the production and loss of quality costs. The production and loss of quality costs for the clutch assembly, (Seshadri and Savage 2002), are calculated using (5.22) and (5.23) respectively. Notice that the production cost uses a reciprocal cost-tolerance model.

$$C_p = \left(3.5 + \frac{0.75}{tol_1}\right) + \left(3.0 + \frac{0.65}{tol_2}\right) + \left(0.5 + \frac{0.88}{tol_4}\right) \tag{5.22}$$

$$C_{LQ} = \Pr(\mathbf{F}) \times C_s \tag{5.23}$$

The problem is formulated as shown in (5.24) where the lower and upper bounds of the means are provided in Table (5-13) and there are also limits on the tolerances. The scrap cost, C_s, is set as $20 and the goal is to search for the set of means and tolerances, of each variable, that minimizes the total cost.

$$\begin{aligned}
&\min \quad C_T = C_{LQ} + C_P \\
&s.t. \\
&\quad g_i(\mathbf{u}) = 0 \quad i = 1,2 \\
&\quad \mathbf{u}_i \cdot null(\nabla_{\mathbf{u}} g_i(\mathbf{u}))_j = 0 \\
&\quad \mu_{\mathbf{V}}^{LB} \leq \mu_{\mathbf{V}} \leq \mu_{\mathbf{V}}^{UB} \\
&\quad \begin{bmatrix} 0.12 \\ 0.08 \\ 0.2 \end{bmatrix} \leq tol_i \leq \begin{bmatrix} 0.25 \\ 0.3 \\ 0.4 \end{bmatrix}
\end{aligned} \tag{5.24}$$

From Table 5-18, using the results obtained from the Mechanistic model as the true or most accurate results, the best mean and tolerance, of each design variable, has been found to be almost identical when the Regression Model or Kriging is used. The same accuracy has not been observed with the RBF. Cost estimates are also the same using Kriging or the RM.

Table 5-17: The best means and tolerances found using each metamodel.

Model	Mean	Tolerance	FORM		Cost
			MLFP and β_R		
Mechanistic	$\mu_V = \begin{bmatrix}55.497\\22.878\\101.89\end{bmatrix}$	$tol_V = \begin{bmatrix}0.25\\0.2082\\0.3667\end{bmatrix}$	$\mathbf{u}_1^* = \begin{bmatrix}0.6847\\1.139\\-1.000\end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix}-0.6884\\-1.140\\0.9972\end{bmatrix}$	$C_T = 17.45$ $C_p = 15.52$ $C_{lq} = 1.925$
			$\beta_1 = 1.663$	$\beta_2 = 1.664$	
RM	$\mu_V = \begin{bmatrix}55.497\\22.878\\101.89\end{bmatrix}$	$tol_V = \begin{bmatrix}0.25\\0.2082\\0.3667\end{bmatrix}$	$\mathbf{u}_1^* = \begin{bmatrix}0.6847\\1.139\\-1.000\end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix}-0.6884\\-1.140\\0.9972\end{bmatrix}$	$C_T = 17.45$ $C_p = 15.52$ $C_{lq} = 1.925$
			$\beta_1 = 1.663$	$\beta_2 = 1.664$	
Kriging	$\mu_V = \begin{bmatrix}55.497\\22.878\\101.89\end{bmatrix}$	$tol_V = \begin{bmatrix}0.25\\0.2082\\0.3666\end{bmatrix}$	$\mathbf{u}_1^* = \begin{bmatrix}0.6849\\1.138\\-1.000\end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix}-0.6883\\-1.140\\0.9973\end{bmatrix}$	Ct = 17.45 $C_P = 15.52$ $C_{LQ} = 1.925$
			$\beta_1 = 1.663$	$\beta_2 = 1.664$	
RBF	$\mu_V = \begin{bmatrix}55.317\\22.881\\101.73\end{bmatrix}$	$tol_V = \begin{bmatrix}0.25\\0.2285\\0.3682\end{bmatrix}$	$\mathbf{u}_1^* = \begin{bmatrix}0.6482\\1.1940\\-0.9516\end{bmatrix}$	$\mathbf{u}_2^* = \begin{bmatrix}-0.5937\\-1.0766\\0.8424\end{bmatrix}$	$C_t = 17.568$ $C_P = 15.24$ $C_{LQ} = 2.33$
			$\beta_1 = 1.659$	$\beta_2 = 1.490$	

Summary

This example has attempted to use metamodels to search for the best design of an overrun clutch assembly. From the results found in Table 5-17, the final design found using Kriging is closest to that obtained using the mechanistic model. The RBF produced the least accurate design. In terms of speed, the number of iterations required to find the MLFP at the original design has been recorded. Both Kriging and the RM have been found to converge quickly. The same has not been observed for the RBF. The accuracy of the FORM-based probability estimates has been compared with the MCS-based probability estimates and the percentage error has been recorded in Table 5-16. From this table, although all percentage errors have been found to be very small, the largest error has been observed from the RBF. This indicates that, perhaps, the using FORM with the RBF is not as accurate as with the other metamodels.

5.5 Conclusions

In this chapter, the optimization of static systems with uncertain design variables has been performed using a metamodel approximation of the mechanistic model and FORM to estimate probability of failure. A MCS has also been performed to check the accuracy of the design found using FORM. Results have been compared and a brief summary following each example has been provided. Generally, the metamodels seem to have a lot of potential for probability-based design optimization. In terms of speed, in all cases, the RM has been found to be the fastest when both FORM and the MCS has been used to estimate probability. This is expected since the metamodel is not as complicated as either Kriging or the RBF. When FORM has been used to estimate probability, the RBF has been found to be the slowest of the three metamodels. However, the RBF has produced a faster computation time when the MCS has been used to estimate probability. In these static examples, the RM has been found to be the best in terms of speed and its good accuracy.

Chapter 6

Probabilistic Design of Dynamic Systems

6.1 Introduction and Problem Description

This chapter shows how SVD and metamodels are used to approximate a time-invariant performance measure based upon a time-variant response through the use of a position-control servo mechanistic. FORM will be used to estimate probability and the final design will be the combination of the means of the design variables that result in a reduced failure probability. Since there is no cost component for this example, tolerance design is performed by lowering the tolerance of the variables and searching for the means that meet the optimization goals.

The servo is to be operated as a position control system whereby the final rotational angle of the shaft is important to the over-all mechanism. A system schematic is shown in Figure 6-1 and comprises mainly an amplifier, a motor and an angular position detector.

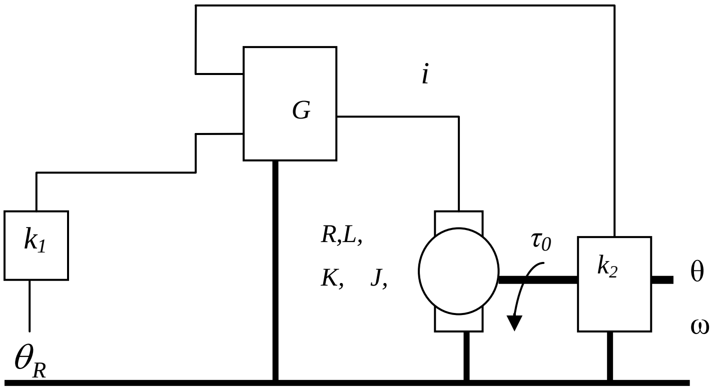


Figure 6-1: A Schematic of the Position-Control Servo

A load torque T_0 (*N-m*) models the subsystem attached to the drive shaft. The input is the reference angle θ_R and the output is the shaft angle θ although angular speed ω and motor current i are measureable as well. The dynamic lifetime, $\boldsymbol{\tau}_{L,}$ is identified as the time at which the angular speed becomes steady-state. The prospective design variables are the amplifier gain *G* (*V/V*), the system rotational friction *B* (*N-m-s/rad*), the system rotational inertia *J* (*kg-m*^{2}), the motor winding inductance *L (H)*, the winding resistance *R (*Ω*)* and the motor torque constant *K (N-m/A)*. A model of the servo is formed as a set of state equations in terms of the shaft angle, the angular speed and the motor current and has the form

$$\begin{bmatrix} \dot{\theta} \\ \dot{\omega} \\ \dot{i} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -B/J & K/J \\ -G/L & -K/L & -R/L \end{bmatrix} \begin{bmatrix} \theta(\tau) \\ \omega(\tau) \\ i(\tau) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 1/J \\ G/L & 0 \end{bmatrix} \begin{bmatrix} \theta_R \\ T_0 \end{bmatrix} \tag{6.1}$$

The winding resistance, *R*, and the friction, *B*, are prone to degradation and their values increase over time. Equation (6.1) has been simulated using Simulink® in MATLAB®.

For an arbitrary initial design, an importance analysis ranks the normalized sensitivities and then is used to separate the variables into a) four *deterministic* variables *L*, *J*, *R* and *B*, that subsequently remain deterministic, and b), two *design* variables *G* and *K* that retain their uncertainty. The performance measures of interest are functions of angle and its angular speed.

For the probabilistic design of the position control servo, three performance measures that include the angular position only and these are a) the integral of squared error (ISE), b) the angular position at peak time (i.e. τ_{pk} = 0.035s) and c) the angular position at settling time(i.e. τ_L = 0.1s). The most common index is the integral of the squared error, ISE (Ogata 1990). The ISE is a performance index that indicates the “goodness” of a system performance. A control system is considered optimal if the values of the parameters of the design variables are chosen so that the selected performance index is minimum or maximum (Ogata 1990). For this example, we attempt to minimize the ISE. Given an input signal $z_R(\tau)$ and the system response *z*, the ISE index is defined as

$$ISE = \int_0^{\tau_L} \left[z_R(\tau) - z(\mathbf{v}, \tau) \right]^2 d\tau \tag{6.2}$$

Or in sampled mode

$$ISE = \Delta\tau \sum_{l=0}^{C} \left[z_R(\tau_l) - \hat{z}(\mathbf{v}, \tau_l) \right]^2 \tag{6.3}$$

With vector notation, the time-invariant ISE is

$$\begin{aligned} ISE(\mathbf{v}) &= \Delta\tau \left[\mathbf{z}_R(\boldsymbol{\tau}) - \hat{\mathbf{z}}(\mathbf{v}, \boldsymbol{\tau}) \right]^T \left[\mathbf{z}_R(\boldsymbol{\tau}) - \hat{\mathbf{z}}(\mathbf{v}, \boldsymbol{\tau}) \right] \\ &= \psi(\mathbf{v}) \end{aligned} \tag{6.4}$$

The derivative of (6.4) is then

$$\frac{\partial(ISE)}{\partial \mathbf{v}^T} = -2\Delta\tau \left[\mathbf{z}_R(\boldsymbol{\tau}) - \hat{\mathbf{z}}(\mathbf{v}, \boldsymbol{\tau}) \right]^T \frac{\partial \hat{\mathbf{z}}(\mathbf{v}, \boldsymbol{\tau})}{\partial \mathbf{v}^T} \tag{6.5}$$

The performance measures are written in terms of the metamodels at appropriate times. The initial design gives $ISE = 11.0 \times 10^{-3}$. For the desired angular position of one radian, the design criteria set the maximum over-shoot shaft angle as 1.11 radians and the minimum shaft angle at settling time as 0.978 radians. Now we can write three LSFs

$$\begin{aligned} g_1 &= 11.0 \times 10^{-3} - ISE \\ g_2 &= 1.11 - \theta(t_{pk}) \\ g_3 &= \theta(t_L) - .978 \end{aligned} \tag{6.6}$$

For probabilistic design we let

$$\boldsymbol{\mu}_{\mathbf{V}} = \begin{bmatrix} \mu_G \\ \mu_K \end{bmatrix}$$

And using the statistical tolerance (i.e. 3σ) as a percentage of the mean, then

$$\mathrm{cov}_{\mathbf{V}} = \begin{bmatrix} \left(\dfrac{tol \times \mu_G}{300} \right)^2 & 0 \\ 0 & \left(\dfrac{tol \times \mu_K}{300} \right)^2 \end{bmatrix}$$

$$\begin{aligned}
&\underset{\mu_{\mathbf{V}}}{\text{Minimize}} \quad \Pr(g_1(\mu_{\mathbf{V}}) < 0) \\
&\quad s.t. \\
&\Phi(-\beta_i) \geq 2.5 \quad i = 1,2,3 \\
&g_i(\mathbf{u}) = 0 \;\; i = 1,2,3 \\
&\mathbf{u}_{\mathrm{i}} \cdot null(\nabla g_i(\mathbf{u}_i))_{\mathrm{j}} = 0 \quad i = 1,2,3 \; j = 1 \\
&L_G + 3\sigma_G \leq \mu_G \leq H_G - 3\sigma_G \\
&L_K + 3\sigma_K \leq \mu_K \leq H_K - 3\sigma_K
\end{aligned} \tag{6.7}$$

Since variables are assumed to be independent and follow a Normal distribution, the transformation to **u**-space looks like

$$v_i = \mu_i + \sigma_i u_i$$

6.2 Training Design

In the initial design, the deterministic noise variables are set as follows: resistance $R = 4\ \Omega$, the motor friction $B = 10^{-4}$ *N-m-s/rad*, the system rotational inertia $J = 10^{-6}$ *kg-m²* and the motor inductance $L = 18/10000$ *H*. The conversion variables k_1 and k_2 are 1 *V/rad*. The load torque is set to a constant $\tau_0 = 2/10000$ *N-m*. The two control variables *G* and *K* are assigned independent normal distributions and their nominal values are 5.0 and 8/1000 respectively.

For design, a wide enough range should be selected so the metamodel fits a large enough range of the design variables to search for an improved design. Using the relation $\mu_i \pm 6\sigma_i$, five levels of each variable, *G* and *K*, are selected and denoted as *L (Low), LN (Low-Nominal), N (Nominal), HN (High-Nominal)* and *H (High)*.

$$\begin{matrix} x_1^L \\ x_1^{LN} \\ x_1^N \\ x_1^{HN} \\ x_1^H \end{matrix} = \begin{bmatrix} 4.5 \\ 4.75 \\ 5.0 \\ 5.25 \\ 5.50 \end{bmatrix} \qquad \begin{matrix} x_2^L \\ x_2^{LN} \\ x_2^N \\ x_2^{HN} \\ x_2^H \end{matrix} = \begin{bmatrix} 0.0072 \\ 0.0076 \\ 0.008 \\ 0.0084 \\ 0.0088 \end{bmatrix}$$

The reference angle is one radian, entered as a step, and the dynamic life time is 0.1s.

When provided with the training design, the first step is to determine how many cycle time increments are necessary. In order to obtain this information, the angular position response is recorded at various possible cycle time increments. The ISE is then estimated using (6.4) and recorded. When the ISE estimate starts to converge, that cycle time increment is chosen. Table 6-1 shows the ISE estimates for three possible cycle time increments that have been recorded at 5 sample training sets.

Table 6-1: The ISE estimates using three possible cycle time increments at 5 training sets.

G	**K**	$\Delta\tau = 0.002$	$\Delta\tau = 0.001$	$\Delta\tau = 0.0005$
4.5	0.0072	0.012916	0.012416	0.012166
4.75	0.0076	0.012191	0.011691	0.011441
5.00	0.0080	0.011568	0.011068	0.010818
5.25	0.0084	0.011027	0.010528	0.010278
5.50	0.0088	0.010554	0.010054	0.009804

From Table 6-1, the difference between the estimates at $\Delta\tau = 0.001$ and $\Delta\tau = 0.0005$ is small. It is important to note that the number of time steps a particular cycle time increment produces should also be taken into consideration. Although SVD is used to reduce the number of metamodels to be built, finding a metamodel to represent the ISE requires that a metamodel at every cycle time increment is obtained that is then subtracted from the desired input signal. As $\Delta\tau$ gets smaller, the number of increments gets larger and the process to build the ISE metamodel gets slower. Since $\Delta\boldsymbol{\tau}_L = 0.1$, the three possible time increments would lead to 51, 101 and 201 time increments respectively. Based on accuracy and speed of computation, $\Delta\tau$ is chosen to be 0.001. Computer experiments and time sampling with $\Delta t = 0.001$ provide the angular position, angular speed and motor current matrices $[\boldsymbol{\Theta}]_{25\times101}$ and $[\boldsymbol{\Omega}]_{25\times101}$ and $[\mathbf{I}]_{25\times101}$ respectively. Herein, only the angular position matrix $[\boldsymbol{\Theta}]_{25\times101}$ is considered.

After application of SVD to the matrix $[\boldsymbol{\Theta}]_{25\times101}$, the second step is to determine the number of significant singular values, *s*. Now, using various possible value of *s*, the response matrix is estimated using the expression

$$\hat{\boldsymbol{\Theta}} = [\mathbf{D}_{\Theta}]_{25\times s}[\mathbf{Q}_{\Theta}^{T}]_{s\times101} \tag{6.8}$$

Using $\hat{\boldsymbol{\Theta}}$, ISE is estimated and compared with the ISE estimates using $\boldsymbol{\Theta}$. Results are shown in Table 6-2 for a sample of 5 training sets. The best results are obtained when $s = 4$ or $s = 5$. Since the result is exact to 6 decimal places when s = 5, this value is chosen to be the number of significant values. A metamodel is now fit for each of the five columns of **D**.

Table 6-2: ISE estimate using three possible values of s.

G	K	Mechanistic Model	Number of columns of S		
			s = 3	s = 4	s = 5
4.5	0.0072	0.012416	0.012398	0.012417	0.012416
4.75	0.0076	0.011691	0.011693	0.011689	0.011691
5	0.008	0.011068	0.011072	0.011069	0.011068
5.25	0.0084	0.010528	0.010526	0.010529	0.010528
5.5	0.0088	0.010054	0.010051	0.01005	0.010054

$$\mathbf{D}_{\mathbf{\Theta}} = \begin{bmatrix} -8.948 & -0.4534 & -0.04937 & 0.006255 & -0.0009667 \\ -9.005 & -0.3469 & -0.02792 & 0.002084 & -0.0002849 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ -9.331 & 0.4316 & -0.05372 & -0.0004822 & 0.004245 \end{bmatrix}_{25\times 5} \tag{6.9}$$

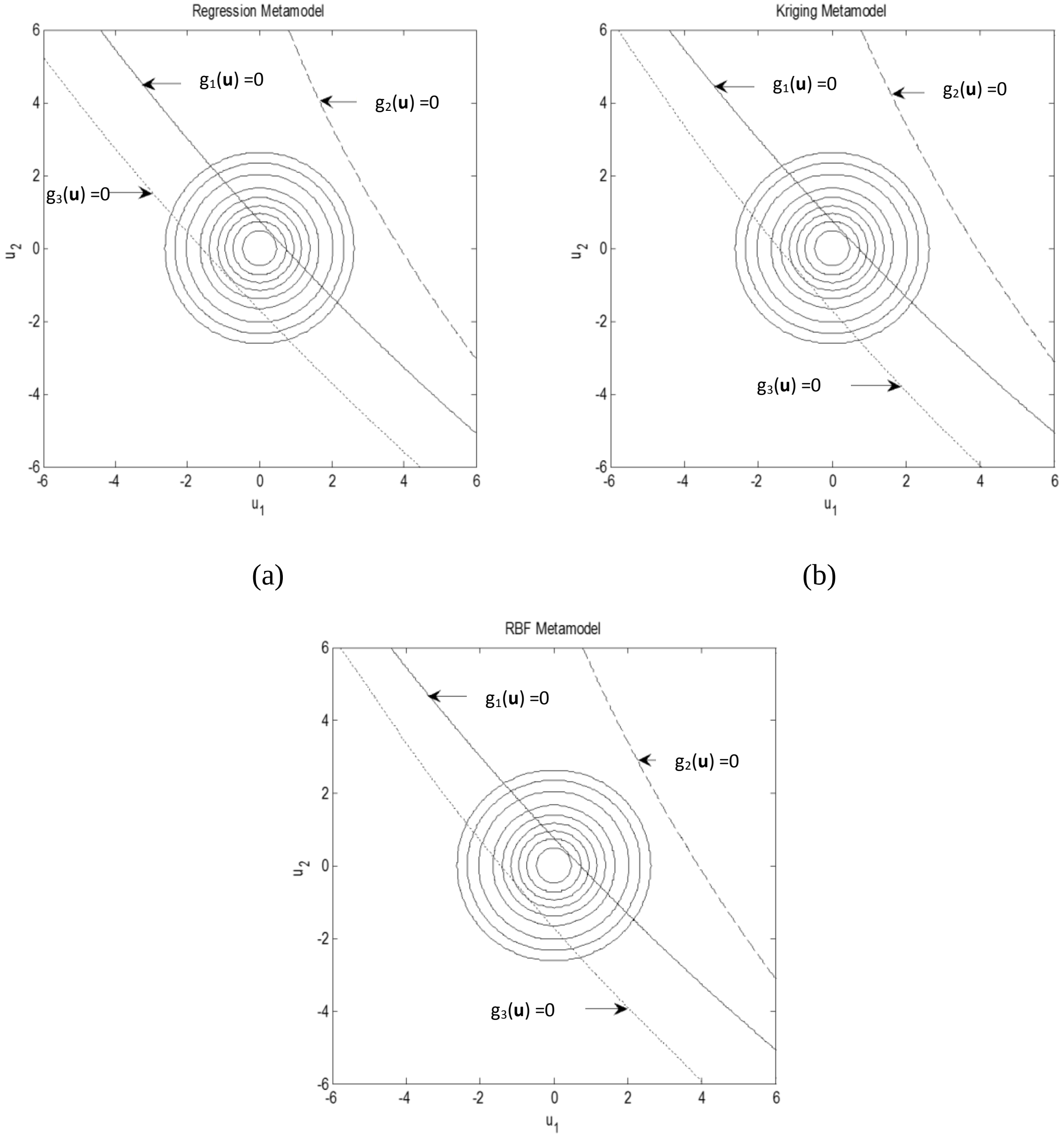
6.3 Results

Experiment 1: Tolerance =5%

Starting with an initial design, $\mu_G = 5, \mu_K = 8/1000$ let tol = 5 in the variance terms. The LSFs at this initial design are built, transformed to **u**-space and their tangent hyper-planes plotted as shown in Figure 6-2. The surfaces estimated by the metamodels are all very similar in appearance and fall at approximately the same position relative to the normal CDF contours. From the figures, the g_1 LSS has the worst failure probability.

To determine the position of the failure region, the sign of the LSF at the origin is determined. At the original design, these signs are a_1 = -1, a_2 = 1, a_3 = 1. Since a_1 = -1, this means that the failure region corresponding to $g_1(\mathbf{u})$ covers the origin. The LSSs at the original design, plotted using a RM approximation of the LSF, are shown in Figure 6-2(a). Figures 6-2(b) and 6-2(c) show similar results when the Kriging and RBF metamodels are used to approximate the LSS.

From the figures, $g_2(\mathbf{u})$ seems to be at an acceptable location because it is quite far away from the origin thus producing a small failure region. The other two surfaces, however, are still too close to the origin especially g$_1$(**u**) that is almost on the origin. The goal is, therefore, to push the failure surfaces further away from the origin.



(a)

(b)

(c)

Figure 6-2: The limit-state surfaces at the original design. (a) Regression Metamodel, (b) Kriging Metamodel (c) RBF Metamodel.

In order to check the actual failure probability, the MLFP at the initial design is found for each metamodel. These results are shown in Table 6-3. The position of the MLFP estimated using each metamodel is very similar. All metamodels converge in 5 iterations; however, the time to converge differs as observed in Table 6-4. In order to check the accuracy of the metamodels, the failure probability at the initial design is estimated using a MCS of the mechanistic model and compared with probability estimates obtained using the metamodels with FORM and MCS. These results are shown in Table 6-4.

From Table 6-4, the best probability estimates are observed at the RBF. Although slow when used with FORM, the accuracy is maintained when MCS is used and the CPU time is a lot faster. Even though FORM with the RBF is so slow, it is still six times faster than using a MCS simulation of the mechanistic model. An increase in speed is further obtained using a MCS of the metamodels.

Table 6-3: The MLFP and reliability index estimates from each metamodel.

Model	**MLFP and β_R**		
RM	$\mathbf{u}_1 = \begin{bmatrix} -0.1699 \\ -0.1592 \end{bmatrix}$	$\mathbf{u}_2 = \begin{bmatrix} 3.087 \\ 1.743 \end{bmatrix}$	$\mathbf{u}_3 = \begin{bmatrix} -0.8623 \\ -0.8116 \end{bmatrix}$
	$\beta_{R_1} = 0.2328$	$\beta_{R_2} = 3.545$	$\beta_{R_3} = 1.185$
Kriging	$\mathbf{u}_1 = \begin{bmatrix} -0.1668 \\ -0.1559 \end{bmatrix}$	$\mathbf{u}_2 = \begin{bmatrix} 3.115 \\ 1.737 \end{bmatrix}$	$\mathbf{u}_3 = \begin{bmatrix} -0.8510 \\ -0.7143 \end{bmatrix}$
	$\beta_{R_1} = 0.2283$	$\beta_{R_2} = 3.567$	$\beta_{R_3} = 1.111$
RBF	$\mathbf{u}_1 = \begin{bmatrix} -0.1671 \\ -0.1559 \end{bmatrix}$	$\mathbf{u}_2 = \begin{bmatrix} 3.118 \\ 1.741 \end{bmatrix}$	$\mathbf{u}_3 = \begin{bmatrix} -0.8572 \\ -0.7127 \end{bmatrix}$
	$\beta_{R_1} = 0.2285$	$\beta_{R_2} = 3.571$	$\beta_{R_3} = 1.115$

Table 6-4: Probability comparisons at the initial design.

Model	$\Pr(g_1(\mathbf{U}) < 0)$	$\Pr(g_2(\mathbf{U}) < 0)$	$\Pr(g_3(\mathbf{U}) < 0)$	Time
Metamodel and FORM				
RM	0.4080	1.962×10^{-4}	0.1180	15s
Kriging	0.4097	1.806×10^{-4}	0.1333	200s
RBF	0.4096	1.779×10^{-4}	0.1325	500s
Metamodel and MCS (100,000 runs)				
RM	0.4111	2.2×10^{-4}	0.1350	3s
Kriging	0.4120	1.8×10^{-4}	0.1345	4s
RBF	0.4109	2.0×10^{-4}	0.1317	4s
Mechanistic Model and MCS (100,000 runs)	0.4089	1.6×10^{-4}	0.1323	3000s

The next step is to obtain a feasible design using (4.30). Since the accuracy of the Regression Model is quite good and is the fastest, this model is used to search for a feasible design that is then used by all metamodels to search for the best design. The limit-state surfaces at the feasible design are shown in Figure 6-3. A test of the sign of the LSSs at the origin show that a_1 = 1, a_2 = 1 and a_3 = 1which means that the origin does not fall in any failure region and, therefore, the design is feasible. From Figure 6-3, $g_1(\mathbf{u})$ is still too close to the origin. The algorithm to find the best design attempts to push the failure surfaces as far away from the origin as possible.

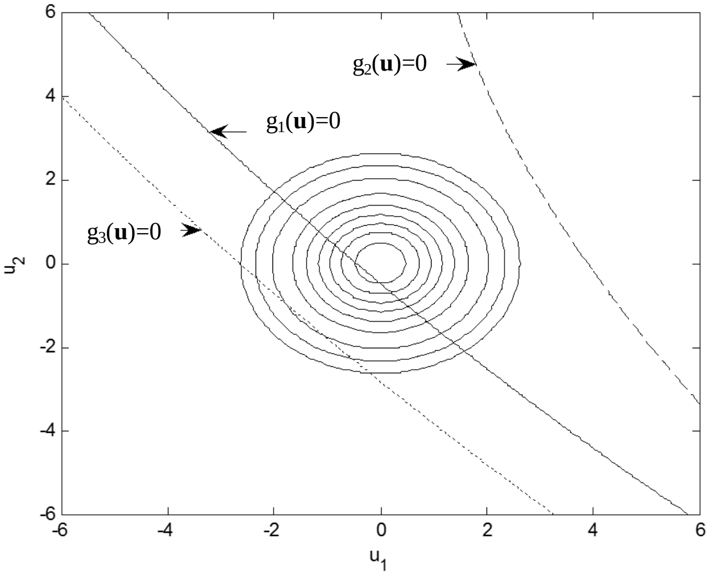


Figure 6-3: The limit-state surfaces at the feasible design.

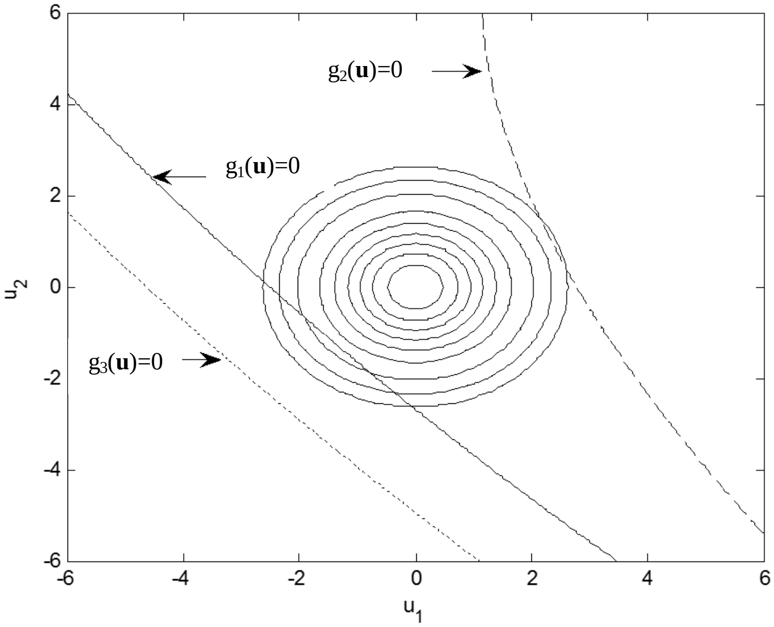


Figure 6-4: The limit-state surfaces, at the best design when tolerance = 5%, plotted using the Regression metamodel.

The LSSs at the best design, when *tol* = 5%, is plotted in Figure 6-4 and the results are shown in Table 6-5. All designs are very close but there is some difference in the location of the MLFP and the reliability index. In order to check the accuracy, a MCS of the mechanistic model is used where $\mu_{\mathbf{v}} = [4.97 \quad 0.0084]^T$ to estimate failure probability. These results are shown in Table 6-6.

Table 6-5: The best design, found using each metamodel, when tolerance = 5%.

Model	**FORM Results**			
	Best Design	**MLFP and β_R**		
RM	$\mu_{\mathbf{v}} = \begin{bmatrix} 4.968 \\ 0.0084 \end{bmatrix}$	$\mathbf{u}_1 = \begin{bmatrix} -1.410 \\ -1.309 \end{bmatrix}$	$\mathbf{u}_2 = \begin{bmatrix} 2.242 \\ 1.107 \end{bmatrix}$	$\mathbf{u}_3 = \begin{bmatrix} -2.092 \\ -1.956 \end{bmatrix}$
		$\beta_{R_1} = 1.924$	$\beta_{R_2} = 2.500$	$\beta_{R_3} = 2.864$
Kriging	$\mu_{\mathbf{v}} = \begin{bmatrix} 4.970 \\ 0.0084 \end{bmatrix}$	$\mathbf{u}_1 = \begin{bmatrix} -1.418 \\ -1.314 \end{bmatrix}$	$\mathbf{u}_2 = \begin{bmatrix} 2.229 \\ 1.132 \end{bmatrix}$	$\mathbf{u}_3 = \begin{bmatrix} -2.063 \\ -1.691 \end{bmatrix}$
		$\beta_{R_1} = 1.933$	$\beta_{R_2} = 2.500$	$\beta_{R_3} = 2.668$
RBF	$\mu_{\mathbf{v}} = \begin{bmatrix} 4.978 \\ 0.0084 \end{bmatrix}$	$\mathbf{u}_1 = \begin{bmatrix} -1.463 \\ -1.371 \end{bmatrix}$	$\mathbf{u}_2 = \begin{bmatrix} 2.329 \\ 0.9093 \end{bmatrix}$	$\mathbf{u}_3 = \begin{bmatrix} -2.031 \\ -1.980 \end{bmatrix}$
		$\beta_{R_1} = 2.009$	$\beta_{R_2} = 2.500$	$\beta_{R_3} = 2.836$

From Table 6-6, the failure probability estimates using the RBF with MCS are the best in two cases; Pr(g_2(**U**) < 0) and Pr(g_3(**U**) < 0). Again, RBF with FORM is the slowest but fastest with the RM. The probability of failure of the ISE limit-state function, g_1, has been best estimated using the RM.

Table 6-6: Probability and CPU time comparisons.

Model	$\Pr(g_1(\mathbf{U}) < 0)$	$\Pr(g_2(\mathbf{U}) < 0)$	$\Pr(g_3(\mathbf{U}) < 0)$	**CPU Time**
Metamodel and FORM				
RM	0.0272	0.0062	0.0021	11s
Kriging	0.0266	0.0062	0.0038	115s
RBF	0.0223	0.0062	0.0023	230s
Metamodel and MC				
(100,000 runs)				
RM	0.02807	0.00594	0.00454	3s
Kriging	0.02626	0.00582	0.00348	4s
RBF	0.0219	0.00615	0.00253	4s
Mechanistic Model and MC				
(100,000 runs)	0.02728	0.0652	0.00280	3000s

In order to find a better design, the tolerance is lowered to 3%. As in the previous case for tolerance = 5%, the feasible design is found and then used to search for the best design. The results are shown in Table 6-6. The best design found using the RM and Kriging metamodels are the same but the difference lies in computing the MLFP. This difference, however, is quite small and finally leads to almost indistinguishable estimates of the reliability index, β.

The next step is to check the accuracy of the final design by MCS of the mechanistic model at the best design found using each metamodel. There are two possible designs produced; one from using PR and Kriging, and the other when the RBF is used. A sample of 100,000 runs is generated at each of the two possible designs and the responses estimated using the mechanistic model. The results are shown in Table 6-8. In this case, using Kriging and RBF with MCS produces much better results than if FORM was used.

Table 6-7: The best design, found using each metamodel, when tolerance = 3%.

Model	**FORM Results**			
	Best Design	**MLFP and β_R**		
RM	$\mu_v = \begin{bmatrix} 4.931 \\ 0.00856 \end{bmatrix}$	$\mathbf{u}_1 = \begin{bmatrix} -2.881 \\ -2.663 \end{bmatrix}$	$\mathbf{u}_2 = \begin{bmatrix} 3.678 \\ 1.678 \end{bmatrix}$	$\mathbf{u}_3 = \begin{bmatrix} -4.018 \\ -3.740 \end{bmatrix}$
		$\beta_1 = 3.924$	$\beta_2 = 4.042$	$\beta_3 = 5.489$
Kriging	$\mu_v = \begin{bmatrix} 4.940 \\ 0.00856 \end{bmatrix}$	$\mathbf{u}_1 = \begin{bmatrix} -2.973 \\ -2.746 \end{bmatrix}$	$\mathbf{u}_2 = \begin{bmatrix} 3.473 \\ 1.700 \end{bmatrix}$	$\mathbf{u}_3 = \begin{bmatrix} -4.008 \\ -3.237 \end{bmatrix}$
		$\beta_1 = 4.040$	$\beta_2 = 3.867$	$\beta_3 = 5.151$
RBF	$\mu_v = \begin{bmatrix} 4.940 \\ 0.00856 \end{bmatrix}$	$\mathbf{u}_1 = \begin{bmatrix} -2.968 \\ -2.738 \end{bmatrix}$	$\mathbf{u}_2 = \begin{bmatrix} 3.477 \\ 1.692 \end{bmatrix}$	$\mathbf{u}_3 = \begin{bmatrix} -4.010 \\ -3.241 \end{bmatrix}$
		$\beta_1 = 4.038$	$\beta_2 = 3.867$	$\beta_3 = 5.156$

Table 6-8: The failure probability comparisons.

Model	$\Pr(g_1(\mathbf{U}) < 0)$	$\Pr(g_2(\mathbf{U}) < 0)$	$\Pr(g_3(\mathbf{U}) < 0)$	CPU Time
Metamodel and FORM				
RM	4.363×10^{-5}	2.649×10^{-5}	2.023×10^{-8}	11s
Kriging	2.676×10^{-5}	5.508×10^{-5}	1.293×10^{-7}	115s
RBF	2.693×10^{-5}	5.503×10^{-5}	1.259×10^{-7}	230s
Metamodel and MC (100,000 runs)				
RM	4×10^{-5}	3×10^{-5}	0	3s
Kriging	3×10^{-5}	7×10^{-5}	0	4s
RBF	4×10^{-5}	6×10^{-5}	0	4s
Mechanistic and MC (100,000 runs)	5×10^{-5}	7×10^{-5}	0	3000s

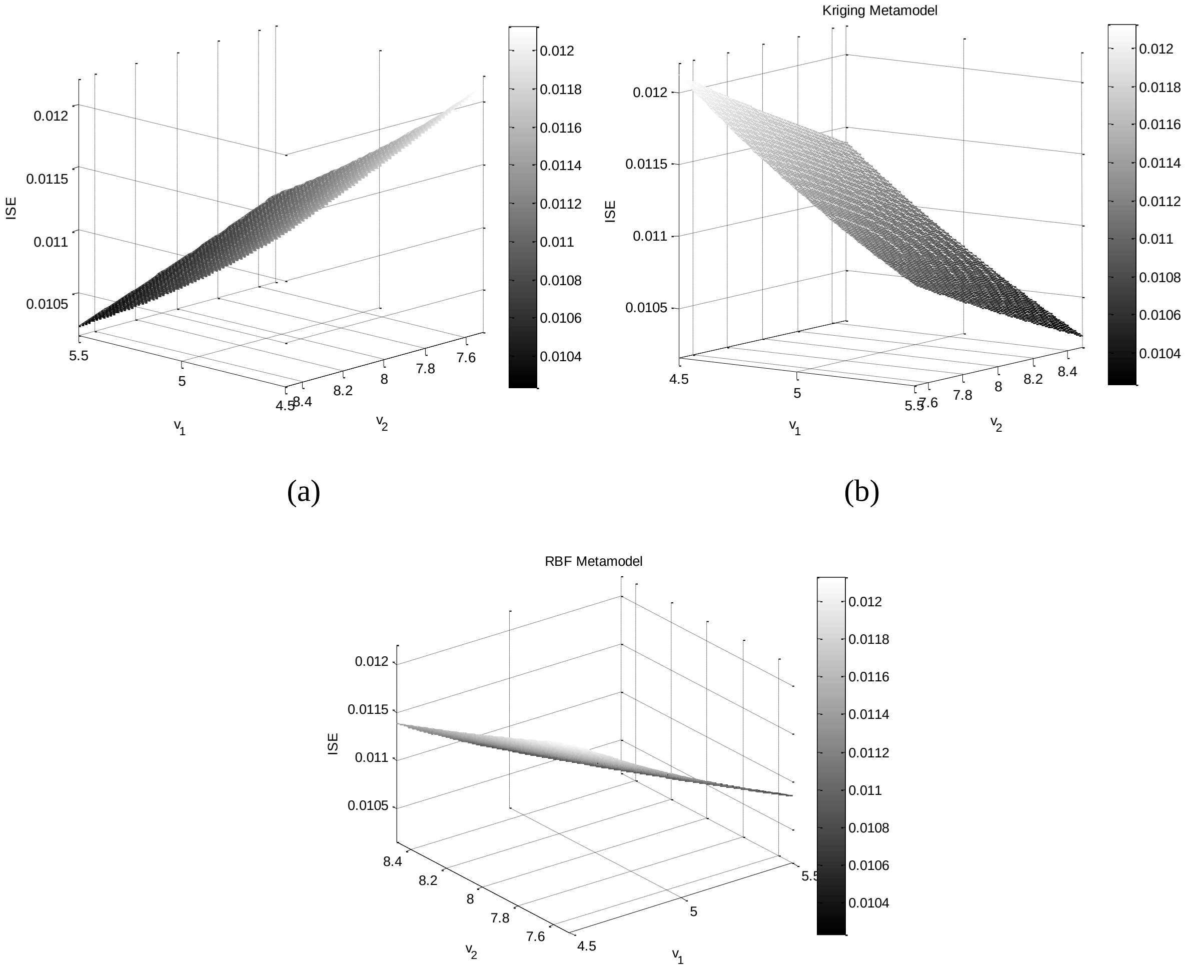
6.4 Conclusions

Based on results that have been obtained using a MCS of the mechanistic model, there has been some error from the results obtained using the metamodels. Since the original angular position response is continuous and dynamic, converting the response into a set of static responses at discrete time steps introduces the first source of error. Intuitively, as the number of time steps increases, the accuracy for estimating the ISE increases. However, since a metamodel has been developed at each time step, as $\Delta\tau$ increases, the number of metamodels increases and, thus, the computation time for developing a metamodel for ISE also increases. This point is especially evident for metamodels like Kriging and RBF that uses every training-set to develop the metamodel. Therefore, a balance must be struck between accuracy and speed.

The next possible source of error is in selecting the appropriate number of columns of **S** that are significant. Analysis that has been done at the start of the chapter shows that 5 columns are appropriate since $\hat{\mathbf{Z}}$ is accurate to six decimal places when compared with **Z**. The third source of error comes from the metamodel itself. The only way to reduce this error is to change the metamodel or to change the training design. The fit of each metamodel has been tested by checking the prediction accuracy using CV-RMSE and CV-MAE. These results are shown in Table 6-9. All metamodels have provided good accuracy. In terms of speed, the RM has been found to be the fastest of all metamodels.

Table 6-9: The CV-RMSE and CV-MAE estimates of each column of $\mathbf{D}_{\boldsymbol{\theta}}$.

	RM		**Kriging**		**RBF**	
	CV-RMSE	**CV-MAE**	**CV-RMSE**	**CV-MAE**	**CV-RMSE**	**CV-MAE**
Column 1	3.248×10^{-5}	1.007×10^{-4}	5.922×10^{-6}	4.067×10^{-6}	6.218×10^{-6}	2.083×10^{-5}
Column 2	6.572×10^{-5}	2.199×10^{-4}	3.796×10^{-6}	2.578×10^{-6}	3.786×10^{-7}	1.268×10^{-6}
Column 3	3.197×10^{-5}	1.223×10^{-4}	7.070×10^{-6}	4.838×10^{-6}	4.908×10^{-7}	1.647×10^{-6}
Column 4	3.693×10^{-5}	1.255×10^{-4}	7.455×10^{-7}	4.677×10^{-7}	1.240×10^{-7}	4.168×10^{-7}
Column 5	7.348×10^{-5}	2.422×10^{-4}	8.882×10^{-7}	4.797×10^{-7}	3.519×10^{-7}	1.185×10^{-6}



(a)

(b)

(c)

Figure 6-5: Plots of the ISE performance measure function. (a) Regression Metamodel, (b) Kriging Metamodel and (c) RBF Metamodel.

Chapter 7
Dynamic Systems with Degradation

A degrading dynamic system has the general form in eq. (1.3) where one or more of the components (design variables) degrade as time passes. Path-tracing sampling methods have previously been used to find failures of performance measures in times. The set-theory method provides a more efficient approach and is the method of choice in this thesis to estimate the cumulative distribution function of the time to failure, *Cdf,* when degradation of the components is assumed. Both methods have been successfully applied to systems with static responses (Stewart and Rosowsky 1998); (Van den Bogaard et al, 2003); (Savage and Son, 2011). A goal of this thesis is to show how the set-theory method can be simply adapted to dynamic systems with degradation. The *Cdf* is obtained by summing incremental probabilities of failure as the limit-state functions move through discrete "service times". Other probability functions, such as the density function, follow from the *Cdf*. It is important to note that the form of the metamodels in static and dynamic systems can be adapted to degrading systems. This is shown in the next section.

7.1 General Metamodel for Dynamic Systems with Degradations

Degradation rates may be random. Let consider service time at discrete intervals Δt then $\mathbf{t} = t_0, t_1 \cdots t_l \cdots t_L$ where $t_l = t_{l-1} + \Delta t$. It follows that we can produce a sequence of response matrices, similar to (3.5), at each service time and we write

$$\mathbf{Z}_0 \quad \mathbf{Z}_1 \quad \cdots \mathbf{Z}_l \quad \cdots \quad \mathbf{Z}_L \tag{7.1}$$

Then following the work with dynamic systems in Chapter 3 we obtain a sequence of responses

$$\hat{\mathbf{z}}_0(\mathbf{v},\boldsymbol{\tau}) \quad \hat{\mathbf{z}}_1(\mathbf{v},\boldsymbol{\tau}) \quad \cdots \hat{\mathbf{z}}_l(\mathbf{v},\boldsymbol{\tau}) \quad \cdots \quad \hat{\mathbf{z}}_L(\mathbf{v},\boldsymbol{\tau}) \tag{7.2}$$

Where

$$\hat{\mathbf{z}}_l(\mathbf{v},\boldsymbol{\tau}) = \mathbf{Q}_l\mathbf{W}_l\bar{\mathbf{r}}(\mathbf{v}) \tag{7.3}$$

or

$$\hat{\mathbf{z}}_l(\mathbf{v},\boldsymbol{\tau}) = \mathbf{Q}_l(\mathbf{W}_l\bar{\mathbf{I}})\mathbf{r}(\mathbf{v}) = \mathbf{T}_l\mathbf{r}(\mathbf{v}) \tag{7.4}$$

Note how the column of functions of the design variables is service-time invariant. In gradient-based optimization and FORM, the derivative of the response with respect to the design variables is required. Since, at each service time, the fitting parameters in **W** and the time contributions in **Q** are constants, the derivatives of interest from (7.4) are

$$\frac{\partial\hat{\mathbf{z}}_l(\mathbf{v},\mathbf{t})}{\partial\mathbf{v}^T} = \mathbf{Q}_l\left(\mathbf{W}_l\bar{\mathbf{I}}\right)\frac{\partial\mathbf{r}(\mathbf{v})}{\partial\mathbf{v}^T} \tag{7.5}$$

7.2 Degradation Model LSF

A service time-dependent limit-state function is denoted as $g(\mathbf{v},t)$ where t denotes the service time. As stated in Chapter 4, positive values, of the LSF, correspond to the safe domain and negative values to the failure domain. For a dynamic system, consider an arbitrary service time t_l; a model of the performance measure that comprises either a dynamic response evaluated at cycle-time τ_k is denoted as $\psi(\mathbf{v},\tau_k,t_l)$or one evaluated over all cycle times $\boldsymbol{\tau}$, is denoted as $\psi(\mathbf{v},\boldsymbol{\tau},t_l)$. When provided with an upper or lower specification limit, ζ, for the performance measure, the LSF, for a response evaluated over all cycle times, is written as

$$g(\mathbf{v},t_l) = \pm(\zeta - \psi(\mathbf{v},\boldsymbol{\tau},t)) \tag{7.6}$$

The failure at the component level within the lifetime $[0,t_L]$ gives the event

$$\Pr(F_T) = \Pr\{g(\mathbf{v},t) \leq 0 \text{ for } \exists t \,\epsilon[0,t_L]\} \tag{7.7}$$

When time is a fixed parameter, say $t = t_l$, the probability of failure at t_l is

$$\Pr(\mathbf{F}(t_l)) = \Pr(g(\mathbf{V},t_l) \leq 0) \tag{7.8}$$

In general, this probability evaluation is not the same as (7.7) since it does not take into account the time history of the system, in particular the possible failure before *t*. For the general case, let us see how to combine the simplicity of (7.8) with the correctness in (7.7).

Consider Figure 7-1 where, for example, the u_1 axis represents an initial random variable and axis u_2 its random degradation rate. Three service-time progressions of a common limit-state surface (solid lines here) are shown where hatching represents the failure regions. Notice that region B is the actual failure region emerging at time t_2 as the LSS progresses through time t_1 from time t_0. The samples in region B contain all of the available initial values and all of the available degradation rates. This area and the joint probability density function of our random variables provide the incremental probability of failure that contributes to the *Cdf*. A conceptually simple and efficient approach to solve the time-history probability problem - called the set-theory method (STM) exists (Savage and Son, 2011). The STM features are as follows:

a) No planned time need be explicitly specified,

b) The method is conceptually simple since it includes the random degradation rates in the joint *pdf* of **V**,

c) Probabilities are found contiguously at successive service time increments.

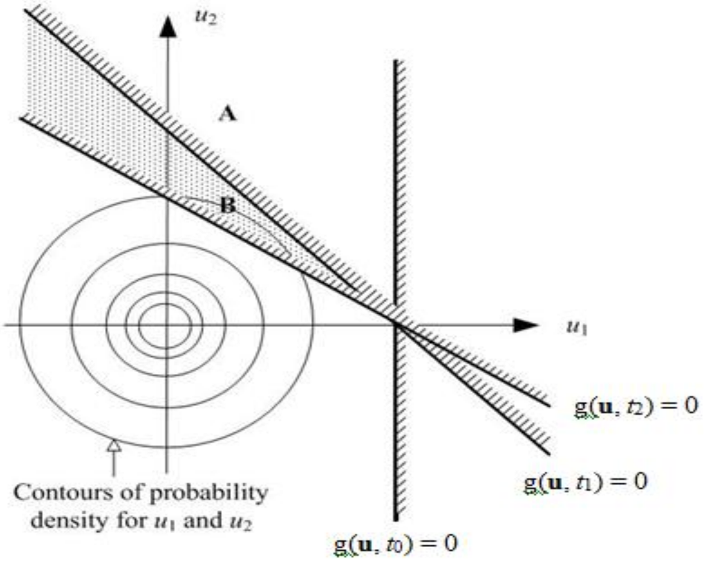


Figure 7-1: Time-variant limit-state surfaces (Source: Savage and Son, 2011).

7.3 Set-Theory Method

Let there be n_{LF} LSFs. Consider, a desired product lifetime, $t_{L,}$ that is made up of a set of discrete time steps using the service-time increment Δt. For a time index l = 0, 1,...L, where L is the number of time steps to the planned time, the time at the l^{th} step is $t_l = l \times \Delta t$. For a performance measure based upon a dynamic response, an approximation to (7.7) using discrete time events based upon a finite time step is written as

$$\Pr\left(F_T\left(t_L\right)\right)=\Pr\left\{\bigcup_{l=0}^{L}\left(\bigcup_{i=1}^{n_{LF}}\left(g_i\left(\mathbf{V},t_l\right)\leq 0\right)\right)\right\} \tag{7.9}$$

A set that represents the *instantaneous failure* region of the i^{th} LSF at any selected discrete service time, t_l is defined as

$$F_{l,i}=\left\{\mathbf{v}\in\mathbf{V}:g_i\left(\mathbf{v},t_l\right)\leq 0\right\} \tag{7.10}$$

Following the system failure discussions in Chapter 4, for a series system configuration, the *system* instantaneous failure region at service time t_l is defined to be the set

$$\mathbf{F}_l=F_{l,1}\cup F_{l,2}\cup\ldots\cup F_{l,n_{LF}}=\bigcup_{i=1}^{n_{LF}}F_{l,i} \tag{7.11}$$

Next, the *system cumulative failure* set A_l is defined as the set that represents the accumulation of all system instantaneous failure regions for all discrete times up to t_l. This set extends (7.11) and is written as

$$A_l=\mathbf{F}_0\cup\mathbf{F}_1\cup\mathbf{F}_2\cup\ldots\cup\mathbf{F}_l=\bigcup_{q=0}^{l}\mathbf{F}_q \tag{7.12}$$

The safe set is denoted as $\overline{A}_l$. The emergence of the incremental failure region from a safe region, from time t_l during time interval Δt, is defined as

$$\mathbf{B}_l=A_{l+1}\cap\overline{A}_l=\left(\bigcup_{q=0}^{l+1}\mathbf{F}_q\right)\cap\left(\bigcap_{q=0}^{l}\overline{\mathbf{F}}_q\right) \tag{7.13}$$

Using the distribution and complement law, $\mathbf{F}_q\cap\left(\overline{\mathbf{F}}_0\cap\ldots\cap\overline{\mathbf{F}}_q\cap\ldots\cap\overline{\mathbf{F}}_l\right)=\emptyset$. Applying this relation to (7.13) for q = 0, 1,...,l, a simpler relation is

$$\mathbf{B}_l = \mathbf{F}_{l+1} \bigcap_{q=0}^{l} \overline{\mathbf{F}}_q = \mathbf{F}_{l+1} \cap \overline{A}_l \tag{7.14}$$

It is straightforward to show both that (a) the failure time history can be written as

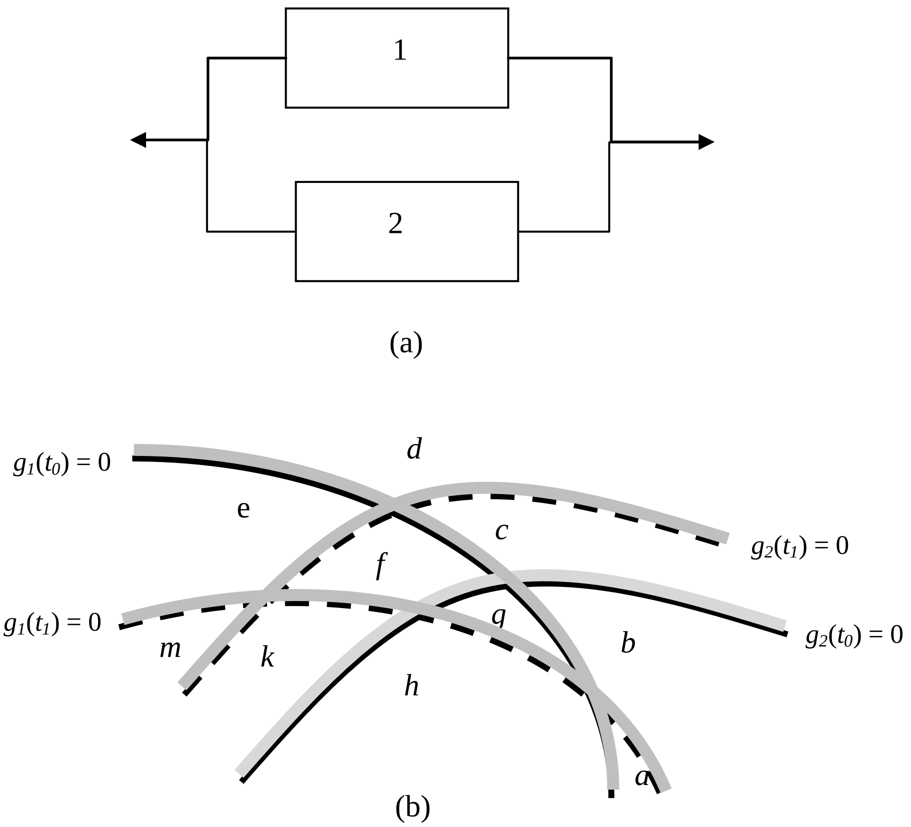
$$\begin{aligned} A_{l+1} &= \bigcup \mathbf{F}_{l+1} = \mathbf{F}_0 \cup \left(\mathbf{F}_1 \cap \overline{\mathbf{F}}_0\right) \cup \left(\mathbf{F}_2 \cap \overline{\mathbf{F}_1 \cup \mathbf{F}_0}\right) \cup \ldots \cup \left(\mathbf{F}_{l+1} \cap \overline{A}_l\right) \\ &= \mathbf{F}_0 \cup \mathbf{B}_0 \cup \mathbf{B}_1 \cup \ldots \cup \mathbf{B}_l \end{aligned}$$

and (b) that all events above are mutually exclusive. The probability of failure is then

$$\Pr(F(t_L)) = \Pr(\mathbf{F}_0) + \Pr(\mathbf{B}_0) + \Pr(\mathbf{B}_1) + \ldots + \Pr(\mathbf{B}_l) + \ldots + \Pr(\mathbf{B}_{L-1})$$

The following example will show that the incremental failure region and its probability is correct and applicable to complex systems.

Example: Consider the system shown in Figure 7-2 that consists of two components in a parallel arrangement. There are two time-variant limit-state functions g_1 and g_2. Their limit-state surfaces in 2-D standard normal space, at two times t_0 and t_1, are shown in Figure 7-2(b). The movement of the surfaces over time are denoted as solid for t_0 and dashed for t_1. The failure regions with respect to each LSS are shown as grey areas. The ten mutually exclusive events are indicated as the unique areas *a*,*b*,...,*m* in Figure 7-2(b).



(a)

(b)

Figure 7-2: A parallel subsystem and the corresponding failure regions: (a) System arrangement (b) failure regions of g_1 and g_2 at t_0 and t_1. (Source: Savage and Son, 2011).

Table 7-1: The sets corresponding to Figure 7-2.

	t_0	t_1
g_1	$F_{0,1} = a \cup b \cup c \cup d$	$F_{1,1} = b \cup c \cup d \cup e \cup f \cup g$
g_2	$F_{0,2} = c \cup d \cup e \cup f \cup k \cup m$	$F_{1,2} = d \cup e \cup m$
Instantaneous Failure	$\mathbf{F}_0 = F_{0,1} \cap F_{0,2} = c \cup d$	$\mathbf{F}_{1,1} = F_{1,1} \cap F_{1,2} = d \cup e$
Cumulative Safe	$\overline{A}_0 = \overline{\mathbf{F}}_0$ $= a \cup b \cup e \cup f \cup g \cup h \cup k \cup m$	$\overline{A}_1 = \overline{\mathbf{F}_0 \cup \mathbf{F}_1}$ $= a \cup b \cup f \cup g \cup h \cup k \cup m$

The right of (7.14) may be reduced so that instantaneous failure sets at only two contiguous time intervals t_l and t_{l+1} are needed. Then the corresponding incremental failure probability becomes

$$\Pr(\mathbf{B}_l) = \Pr(\mathbf{F}_{l+1} \cap \overline{\mathbf{F}}_l) - \Pr((\mathbf{F}_{l+1} \cap \overline{\mathbf{F}}_l) \cap A_l) \tag{7.15}$$

where $\Pr((\mathbf{F}_{l+1} \cap \overline{\mathbf{F}}_l) \cap A_l) << \Pr(\mathbf{F}_{l+1} \cap \overline{\mathbf{F}}_l)$. To illustrate this argument, consider Figure 7-3 that shows a time-variant limit-state surface at three contiguous discrete times (t_0, t_1 and t_2). Dark areas indicate failure regions. The sets of interest are shown in Table (7-2). From Table 7-2, $\boldsymbol{F}_1 \cap \overline{\boldsymbol{F}}_0 = a$ and $(\mathbf{F}_1 \cap \overline{\mathbf{F}}_0) \cap A_0 = \emptyset$. Using equation (7.15), $\mathbf{B}_0 = a$ which coincides with the correct evaluation of $\mathbf{B}_0$ using the equation $\boldsymbol{B}_l = A_{l+1} \cap \bar{A}_l$. Next, for $\mathbf{B}_1$, from Table (7-2), $\boldsymbol{F}_2 \cap \overline{\boldsymbol{F}}_1 = b \cup d$ and $(\boldsymbol{F}_2 \cap \overline{\boldsymbol{F}}_1) \cap A_1 = d$. Since the correct result is $\mathbf{B}_1 = b$, there is a positive and conservative probability error due to the region d. Since $d \ll b \cup d$, the probability error is small. If, however, the time step is reduced, the region, d, begins to vanish. Therefore,

$$\Pr(\mathbf{B}_l) \leq \Pr(\mathbf{F}_{l+1} \cap \overline{\mathbf{F}}_l) \tag{7.16}$$

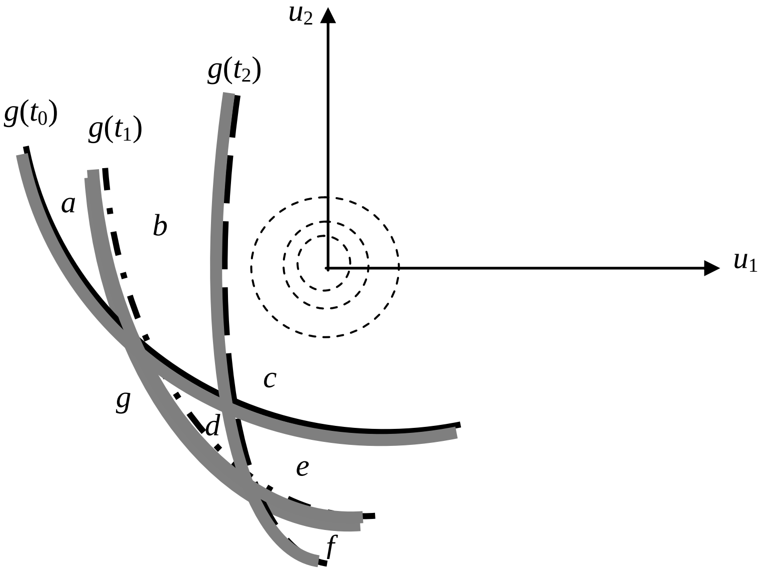


Figure 7-3: A time-variant limit-state surface as it moves through time.

Table 7-2: Sets for Figure 7-3.

t_0	t_1	t_2
$\mathbf{F}_0 = (d \cup e \cup f \cup g)$	$\mathbf{F}_1 = (a \cup f \cup g)$	$\mathbf{F}_2 = (a \cup b \cup d \cup g)$
$\overline{\mathbf{F}}_0 = (a \cup b \cup c)$	$\overline{\mathbf{F}}_1 = (b \cup c \cup d \cup e)$	$\overline{\mathbf{F}}_2 = (c \cup e \cup f)$
$A_0 = (d \cup e \cup f \cup g)$	$A_1 = (a \cup d \cup e \cup f \cup g)$	$A_2 = (a \cup b \cup d \cup e \cup f \cup g)$

A more convenient probability evaluation follows. By invoking the identity

$$\mathbf{F}_{l+1} \cup \mathbf{F}_l = \mathbf{F}_l \cup \left(\mathbf{F}_{l+1} \cap \overline{\mathbf{F}}_l\right) \tag{7.17}$$

and replacing the last intersection in (7.17) with $\mathbf{B}_l$, (7.17) becomes

$$\mathbf{F}_{l+1} \cup \mathbf{F}_l = \mathbf{F}_l \cup \mathbf{B}_l \tag{7.18}$$

Finally the incremental failure probability in terms of two contiguous service time increments becomes

$$\Pr(\mathbf{B}_l) \leq \Pr(\mathbf{F}_{l+1} \cup \mathbf{F}_l) - \Pr(\mathbf{F}_l) \tag{7.19}$$

Probability evaluation by Monte Carlo is straightforward and follows.

7.3.1 Probability Evaluation

A Monte Carlo evaluation provides the safe-fail elemental state for any LSF g_i at time t_l as

$$e_{l,i} = \begin{cases} 0 & \text{if } g_i(\mathbf{v}, t_l) \leq 0 \\ 1 & \text{otherwise} \end{cases}$$

Then for a system at time t_l we have the state

$$\lambda_l = \begin{cases} 0 & \text{if the system is in a failure state} \\ 1 & \text{if the system is in a safe state} \end{cases}$$

For example, given a parallel system comprising n_{LF} elements, the system state is calculated as

$$\lambda_l = 1 - \prod_{i=1}^{n_{LF}} (1 - e_{l,i})$$

For example for n_{LF} elements in series, the system state is

$$\lambda_l = \prod_{i=1}^{n_{LF}} (e_{l,i})$$

Let us define two test functions at the system level in terms of the system states to help enumerate the incremental probabilities of system failure: these are,

$$\phi_1(\mathbf{v},t_l)=\begin{cases}1 \text{ if } \lambda_l=0\\ 0 \text{ otherwise}\end{cases}$$

$$\phi_2(\mathbf{v},t_l,t_{l+1})=\begin{cases}1 \text{ if } \lambda_l=0 \,\text{or}\, \lambda_{l+1}=0\\ 0 \text{ otherwise}\end{cases}$$

For N sample sets, with $\mathbf{v}^{(k)}$ denoting the k^{th} sample set, the number of systems that fail at time t_l is calculated as

$$N_f(t_l)=\sum_{k=1}^{N}\phi_1\left(\mathbf{v}^{(k)},t_l\right)$$

Similarly, the number of systems that fail at time t_l or t_{l+1} is

$$N_f(t_l,t_{l+1})=\sum_{k=1}^{N}\phi_2\left(\mathbf{v}^{(k)},t_l,t_{l+1}\right)$$

From equation (7.19), it follows that the predicted incremental failure over time Δt is evaluated as

$$\Pr(\mathbf{B}_l)=\frac{N_f(t_l,t_{l+1})-N_f(t_l)}{N} \tag{7.20}$$

With the help of (7.20), the relation between the true and predicted *Cdf* is written as

$$\Pr(F_T(t_L))\leq\Pr(F(t_L))=\Pr(\mathbf{F}_0)+\sum_{l=0}^{L-1}(\Pr(\mathbf{B}_l)) \tag{7.21}$$

7.4 Error Analysis

The accuracy of the *Cdf* depends upon the accuracy of the incremental probabilities. The true incremental failure is represented as

$$\Pr(\mathbf{B}_l)_T=\Pr(\mathbf{B}_l)+\varepsilon_{Total}(t_l,t_{l+1}) \tag{7.22}$$

Where the total error, $\varepsilon_{Total}(t_l,t_{l+1})$ is written as the sum of three error sources

1. $\varepsilon_1(t_l,t_{l+1})$ due to sample size N too small (MCS)
2. $\varepsilon_2(t_l,t_{l+1})$ due to using only two contiguous events

3. $\varepsilon_3(t_l,t_{l+1})$ due to making finite time step Δt too large.

The error ε_1 is due to the number of random samples taken to estimate probability. This error can be reduced by increasing N. The error ε_2 occurs when the historical time events, in set A_l, are simplified using only two contiguous time events, $\mathbf{F}_{l+1}$ and $\mathbf{F}_l$. Finally, the error ε_3 is due to approximating continuous time using discrete time events with step size Δt. In general, as the value of Δt decreases, so does the magnitude of ε_3. A smaller time step size, however, leads to more evaluations and to a longer computation time over t_L. Typically, the optimum step size is determined by estimating the correlation coefficient from the positions of a particular LSF, say g_i, at contiguous times t_l and t_{l+1} (Savage and Son, 2011).

7.5 Summary

This chapter has shown how the general form of the metamodels is adapted for dynamic degrading systems. By the selection of performance measures, cycle-time invariant LSFs have been obtained. In **u**-space a contiguous movement of the LSSs over service time has been observed. The incremental failure regions have been identified by the set-theory method. Probabilities of the incremental failure region have been used to estimate the *Cdf*. The set-theory method has been illustrated using a few examples and a Monte Carlo evaluation of the incremental failure probability has been presented.

Chapter 8

Case Study: Estimating the *Cdf* for the Servo

8.1 Introduction

The servo control system presented in Chapter 6 will be used to illustrate the methodology for approximating the cumulative failure when degradation is assumed. The design variables follow from Chapter 6 along with their nominal values. Shaft angle, angular speed and current were available from the model although only shaft angle was used. For this example, the performance measures of interest are the settling angle, θ_s and the energy, E, required to drive the load. The settling angle response is easily found. However, the energy supplied by the servo is not explicitly known, it must be estimated using the responses available. Let us write energy over cycle lifetime as

$$E = \int_0^{\tau_L} P_\tau \, d\tau = \int_0^{\tau_L} T(\tau)\omega(\tau)d\tau \tag{8.1}$$

where ω(τ) represents the dynamic angular speed, τ_L represents the lifetime of the dynamic response and $T(\tau)$ represents the load torque. For ω(τ) recorded at incremental cycle times ω(τ_k) an approximation of (8.1) is expressed as

$$\hat{E} \cong \sum_{k=1}^{C} T(\tau_k)\omega(\tau_k)\Delta\tau \tag{8.2}$$

where C is the total number of incremental time steps that make up the lifetime, $\omega(\tau_k)$ is the angular speed at each time increment, $\Delta\tau$ is the cycle time increment and $T(\tau_k)$ is the torque at each time increment. In vector form (8.2) becomes

$$\hat{E} \cong \mathbf{T}^T \hat{\boldsymbol{\omega}}(\mathbf{v}, \boldsymbol{\tau})(\Delta\tau) = \psi(\mathbf{v}) \tag{8.3}$$

When the angular speed vector is replaced by (7.3) or (7.4) we now write, for service time t_l, the energy has a function of only the design variables

$$\hat{E} \cong E_l(\mathbf{v}) \tag{8.4}$$

Since the supply energy is required to drive the motor, a supply energy that is too low would not be able to drive the motor. Therefore, for energy the design specification a “larger-is-better” criterion required. This means that a lower limit is used when building the limit-state function that the Energy should not fall below. From some previous experimental work, the lower limit design specification for energy is shown in (8.5). The design specification for the settling angle is the same as in the previous example. Based on these requirements, the two limit-state functions for settling angle and supply energy, for any service time, are

$$\begin{aligned} g_1(\mathbf{v}) &= \theta_s - 0.978 \\ g_2(\mathbf{v}) &= E - 0.1959\times10^{-3} \end{aligned} \tag{8.5}$$

8.2 Problem Set-up

Selecting Δτ

Now, let us select Δτ that sets C the number of columns in the response matrix. We consider the following approach. As shown in (8.3), the cycle-time increment is important. Ideally, the smaller the time increment, the more accurate would be the energy estimation. In order to determine the most accurate number of time steps, the dynamic response is recorded at various time increments and the energy is computed using (8.2). The time increment is reduced until the estimated energy approaches an asymptotic value. Table (8-1) shows the energy estimate at five levels of the amplifier gain, G, and winding resistance, R, around their nominal values.

Table 8-1: The supply energy estimates using three possible cycle time increments.

G	R	Cycle-Time Increments			
		Δτ = 0.005	Δτ = 0.002	Δτ = 0.001	Δτ = 0.0005
4.966	3.82	0.000193966	0.000196141	0.000196242	0.000196234
5.083	3.91	0.000193737	0.00019559	0.000195686	0.000195675
5.2	4	0.000193936	0.000196149	0.000196256	0.000196248
5.317	4.09	0.00019431	0.000196668	0.00019679	0.000196789
5.434	4.18	0.000193908	0.000196156	0.000196268	0.000196261

From Table 8-1, we note that at $\Delta\tau = 0.001$ the energy is sufficiently accurate. Now from (3.4) $C = 101$.

Selecting the Training Design

The winding resistance, R¸ is assumed to degrade and, using a degradation path model, the degradation of the resistance is modelled as.

$$R = R_0(1 + K_R t) \tag{8.6}$$

where K_R represents the percentage degradation rate of the resistance. In the degradation path model, it is assumed that the degradation rate is also a random variable. Assume, for each service time increment, the three design variables, G (v_1), R_0 (v_2) and K_R (v_3) are assigned independent normal distributions.

Assuming tolerance 3% of the mean, five levels of each design variable are obtained using the relation $\mu_i \pm 4.5\sigma_i$.

$$\begin{bmatrix} x_1^L \\ x_1^{LN} \\ x_1^N \\ x_1^{HN} \\ x_1^H \end{bmatrix} = \begin{bmatrix} 4.966 \\ 5.083 \\ 5.200 \\ 5.317 \\ 5.434 \end{bmatrix} \quad \begin{bmatrix} x_2^L \\ x_2^{LN} \\ x_2^N \\ x_2^{HN} \\ x_2^H \end{bmatrix} = \begin{bmatrix} 3.82 \\ 3.91 \\ 4.00 \\ 4.09 \\ 4.18 \end{bmatrix} \quad \begin{bmatrix} x_3^L \\ x_3^{LN} \\ x_3^N \\ x_3^{HN} \\ x_3^H \end{bmatrix} = \begin{bmatrix} 0.00382 \\ 0.00391 \\ 0.004 \\ 0.00409 \\ 0.00418 \end{bmatrix}$$

Where $M = 125$. The experiments are run and the angular speed response matrix, $\mathbf{\Omega}$, is generated.

Selecting *s*

Now let us look at SVD at service time t_0. Application of SVD generates $\hat{\boldsymbol{\Omega}}$. Comparison of energy using $\boldsymbol{\Omega}$ and $\hat{\boldsymbol{\Omega}}$ indicates that five columns of **S** (hence **D)** are sufficient as shown in Table 8-2.

Table 8-2: The energy estimate using different values of *s*.

G	**R**	**Mechanistic Model**	**Number of columns of D.**			
			***s* = 3**	***s* = 4**	***s* = 5**	***s* = 6**
		($\times 10^{-4}$)	($\times 10^{-4}$)	($\times 10^{-4}$)	($\times 10^{-4}$)	($\times 10^{-4}$)
4.966	3.82	1.958585	1.959082	1.958589	1.958574	1.958568
5.083	3.91	1.958686	1.958979	1.958708	1.958678	1.958685
5.2	4	1.958784	1.958866	1.958808	1.958774	1.958785
5.317	4.09	1.958880	1.958748	1.958890	1.958864	1.958872
5.434	4.18	1.958973	1.958624	1.958957	1.958948	1.958945

Metamodels are produced for angular speed over cycle time. Performance measures are now readily written as functions of the design variables **v**.

Feasible Design

Before the *Cdf* is estimated, let us determine if a feasible design exists at t_0. Starting with the nominal values $G(v_1) = 5.2$ and $R(v_2) = 4$, the LSSs are plotted (Figure 8-1) and the reliability indices and signs are calculated. From Figure 8-1, the LSSs are quite far away from the origin indicating that the failure probability is small and the signs are positive. A quick calculation of β_R for each LSF gives $\beta_{R1} = 4.6429$ and $\beta_{R2} = 2.6914$ which gives failure probabilities of 1.718×10^{-6} and 0.0036 respectively. Since this design is feasible, the *Cdf* is now estimated.

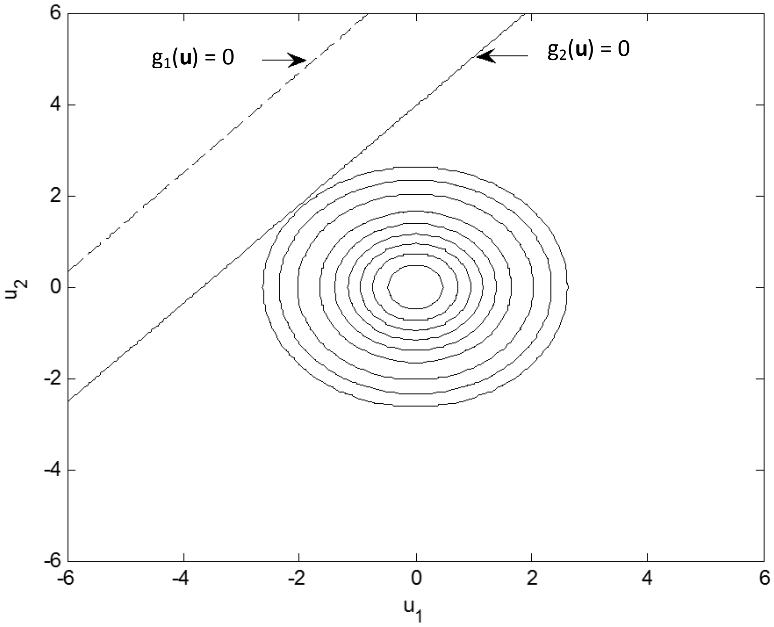


Figure 8-1: The LSSs at the initial design.

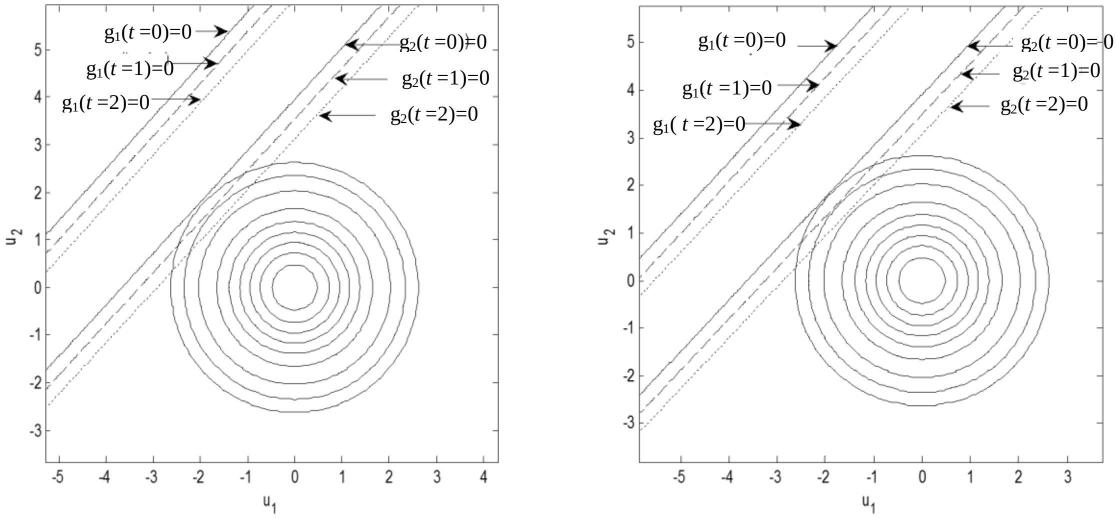
8.3 Building a *Cdf*

The product lifetime is assumed to be 5 years ($t_L = 5$) and, for $\Delta t = 0.2$, a total of 26 service time increments is produced. At each service time increment, the SIMULINK model of the mechanistic model is run and a new set of responses at the training design is obtained. Metamodels are then built and FORM or MCS, along with the set-theory method, is used to estimate the system and incremental failure probabilities. The results are compared with a MCS of the mechanistic model.

In order to illustrate how the limit-state surfaces move through time, the limit-state surfaces are plotted at three service time increments as shown in Figures 8-2(a), (b) and (c). Each figure represents the limit-state surface plotted when the different metamodels are used to approximate the LSF. From the figures, as the servo "ages", the surfaces move closer to the origin which means that the probability of failure increases. Since the surfaces are fully correlated (the angle between surfaces is equal to zero), the system failure is equal to the failure probability from g_2 or $\Pr\{g_2(\mathbf{U}) < 0\}$. Also, the failure regions move through time in one spatial direction indicating that the time-varying failure regions are also fully correlated. In other words, the failure region of g_2 at 1 year comprises of the entire failure regions of g_2 at the start of the product lifetime.

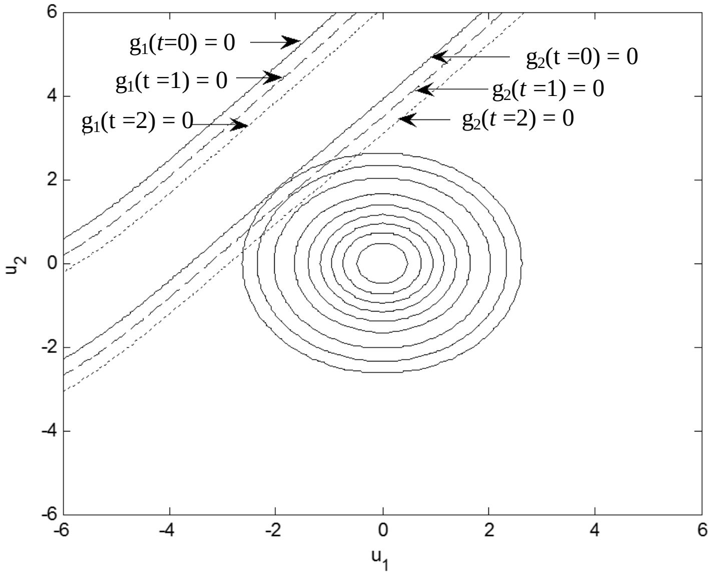
From the three figures, the surfaces are mostly straight lines (exact for the RBF estimate where there is some curvature), therefore, the failure probability estimate using FORM should be almost exactly the same as the failure probability obtained using a MCS. At each service time, the failure regions corresponding to g_1 and g_2 are fully correlated; therefore, the system failure is calculated by estimating $\Pr(g_2 < 0)$. Also, since the LSSs have a constant spatial direction, then the instantaneous probability is the same as the cumulative probability (Savage and Son 2011).

Finally, the failure surfaces from the three graphs are very similar; therefore, the FORM probability of failure estimates should be very close.



(a)

(b)



(c)

Figure 8-2: The limit-state surfaces at three service time increments. (a) RM approximation, (b) Kriging and (c) RBF.

8.3.1 Experiment 1: μ(K$_R$) = 0.4%

Table (8-3) shows the failure probability at each service time estimated using each metamodel with FORM. The second column shows the instantaneous failure probability using a MCS of the mechanistic model. These results are assumed to be the most accurate and are used to determine the accuracy of the metamodel estimates. From a visual inspection of the probability estimates, the metamodels produce good results. For a better analysis of the errors, the percentage error of the FORM and MCS estimates are plotted in Figures (8-3) and (8-4) where the error is calculated as

$$\%\hat{\varepsilon}_l = \left(\frac{[\mathbf{F}_l]_{\text{Mechanistic}} - [\mathbf{F}_l]_{\text{Metamodel}}}{[\mathbf{F}_l]_{\text{Mechanistic}}} \right) \times 100 \tag{8.7}$$

If the error is positive, then the metamodel estimates are “optimistic” and for negative errors, the metamodel estimates are “conservative”. From Figure 8-3, regression-based FORM estimates are always optimistic. For Kriging and the RBF, the error estimates are conservative up to 3 years and then they become optimistic. In terms of overall accuracy, the errors of Regression Model estimates are larger than those obtained using Kriging or the RBF. When a MCS is used, with the metamodels, to estimate probability, the trend is the same and, again, the Regression Model produces the largest error. Based on Figures 8-3 and 8-4, the FORM-based probability estimates for the Kriging and RBF metamodels are larger than the MCS-based estimates. Another trend that is observed is large errors at the start of the produce lifetime. Since probability of failure at the start is quite small (order 10^{-3}), following (8.7), dividing by a small number results in a large percentage error. A comparison of the *Cdf*'s shown in Figures 8-5 and 8-6 show that the errors are really very small at the start of the service-time.

Table 8-3: The cumulative failure probability estimated using FORM and each metamodel.

Time	**Mechanistic**	**RM**		**Kriging**		**RBF**	
	MCS	**FORM**	**MCS**	**FORM**	**MCS**	**FORM**	**MCS**
0	0.0039	0.0036	0.0039	0.0040	0.0039	0.0031	0.0040
0.2	0.0044	0.0042	0.0045	0.0046	0.0044	0.0050	0.0047
0.4	0.0051	0.0049	0.0053	0.0054	0.0051	0.0053	0.0056
0.6	0.0064	0.0058	0.0060	0.0063	0.0061	0.0062	0.0065
0.8	0.0075	0.0068	0.0068	0.0074	0.0072	0.0079	0.0077
1	0.0086	0.0079	0.0079	0.0086	0.0085	0.0085	0.0089
1.2	0.0095	0.0092	0.0092	0.0100	0.0099	0.0098	0.0100
1.4	0.0113	0.0106	0.0106	0.0115	0.0113	0.0113	0.0115
1.6	0.0128	0.0123	0.0123	0.0133	0.0131	0.0131	0.0133
1.8	0.0144	0.0142	0.0141	0.0153	0.0152	0.0151	0.0152
2	0.0171	0.0163	0.0163	0.0176	0.0173	0.0173	0.0173
2.2	0.0198	0.0187	0.0188	0.0201	0.0201	0.0199	0.0197
2.4	0.0222	0.0214	0.0214	0.0230	0.0229	0.0226	0.0225
2.6	0.0255	0.0244	0.0244	0.0262	0.0259	0.0258	0.0254
2.8	0.0298	0.0278	0.0278	0.0297	0.0290	0.0293	0.0289
3	0.0343	0.0315	0.0314	0.0336	0.0330	0.0332	0.0324
3.2	0.0391	0.0356	0.0354	0.0379	0.0376	0.0374	0.0369
3.4	0.0439	0.0402	0.0401	0.0427	0.0422	0.0422	0.0415
3.6	0.0495	0.0452	0.0454	0.0479	0.0473	0.0473	0.0468
3.8	0.0563	0.0507	0.0508	0.0536	0.0539	0.0529	0.0524
4	0.0621	0.0568	0.0568	0.0598	0.0603	0.0591	0.0590
4.2	0.0685	0.0633	0.0636	0.0666	0.0672	0.0658	0.0658
4.4	0.0772	0.0705	0.0708	0.0740	0.0746	0.0733	0.0732
4.6	0.0854	0.0783	0.0786	0.0820	0.0832	0.0811	0.0817
4.8	0.095	0.0867	0.0872	0.0906	0.0908	0.0897	0.0903
5	0.1041	0.0957	0.0961	0.0998	0.101	0.0990	0.100

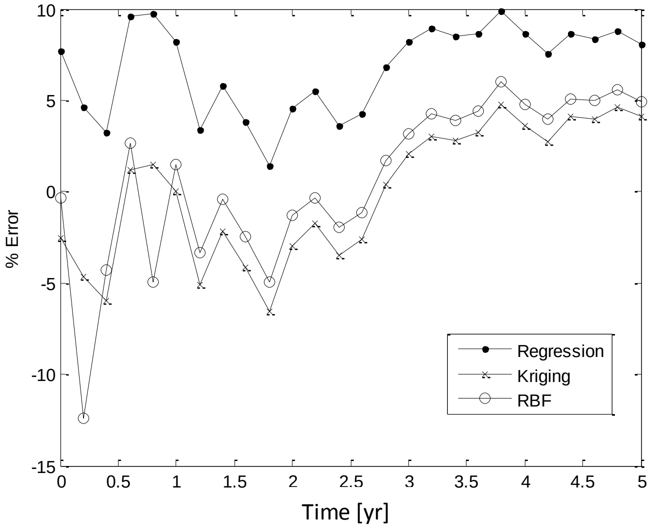


Figure 8-3: The metamodel-based FORM %error for each metamodel.

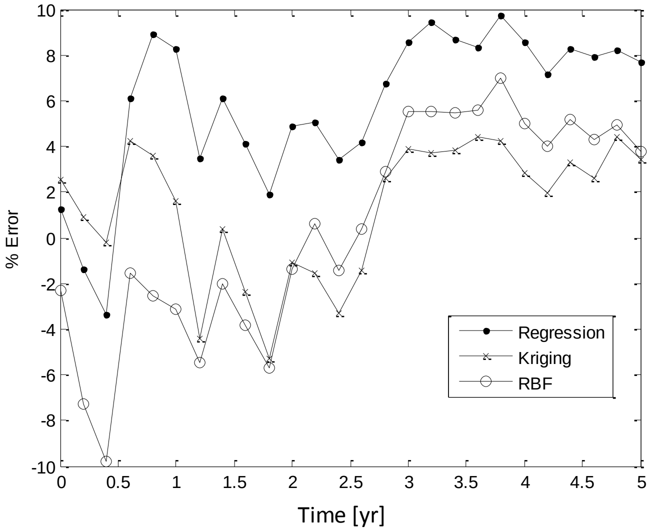


Figure 8-4: The %error from using MCS with the metamodels to estimate probability.

Now, the *Cdf* is plotted using the probability estimates from both FORM and MCS. Figure 8-5 shows the *Cdf* plot using FORM and the metamodels and is compared with the Mechanistic model results. The error plots discussed previously displayed large optimistic results from the Regression Model. This is observed in both Figures 8-5 and 8-6 where the Regression Model *Cdf* falls below the curves obtained from the mechanistic model, Kriging and the RBF. Generally, the curves are similar but the metamodel estimates are optimistic.

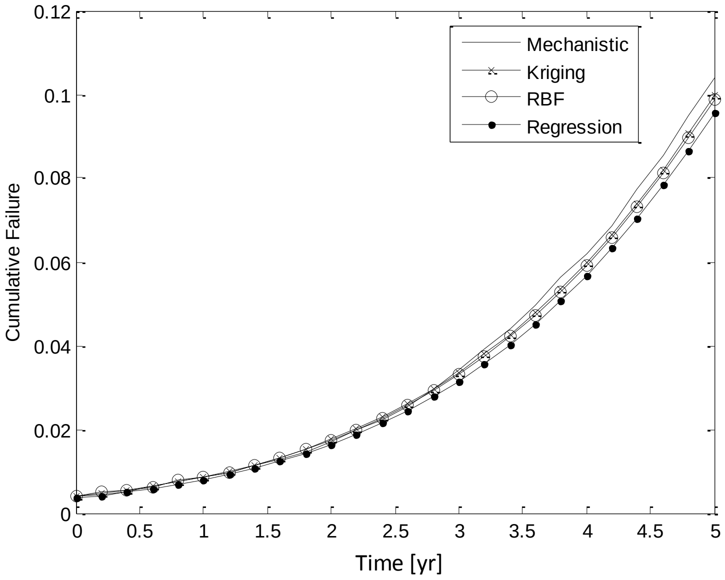


Figure 8-5: The approximate *Cdf* where probability is estimated using FORM with Metamodels.

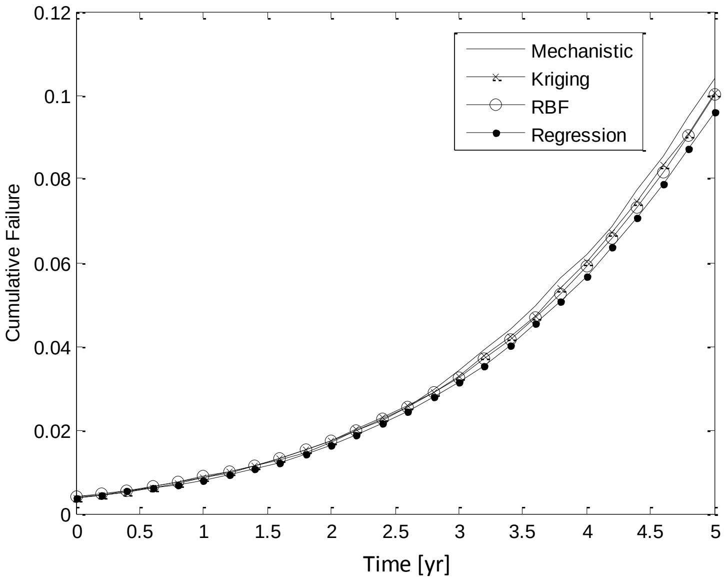


Figure 8-6: The approximate *Cdf* where probability is estimated using MCS with metamodels.

Error Analysis

Since metamodels are built for each column of **D**, the predictive error of each metamodel is estimated at each column of **D**. At each service time, a new response matrix, $\mathbf{\Omega}$ is generated

$$\mathbf{\Omega}(\mathbf{v},\boldsymbol{\tau},t_1)\ldots\mathbf{\Omega}(\mathbf{v},\boldsymbol{\tau},t_l)\ldots\mathbf{\Omega}(\mathbf{v},\boldsymbol{\tau},t_L)$$

which leads to a new $\mathbf{D}_\omega$ matrix and thus fits for the metamodels. A sample of the predictive error of each column of $\mathbf{D}_\omega$ for three chosen service times is shown in Table 8-4 and Table 8-5 shows the error for the settling angle. Generally, both Kriging and the RBF are better than the RM. This result has been observed in Figures 8-3 and 8-4.

Table 8-4: The CV-RMSE and CV-MAE at three service times for each metamodel.

Time	**Model**	**Method**	**Column of $\mathbf{D}_\omega$**				
			1	**2**	**3**	**4**	**5**
0	RM	CV-RMSE	9.22×10^{-03}	1.16×10^{-02}	2.34×10^{-02}	4.85×10^{-03}	1.99×10^{-02}
		CV-MAE	6.76×10^{-03}	9.28×10^{-03}	1.76×10^{-02}	3.43×10^{-03}	1.30×10^{-02}
	Kriging	CV-RMSE	3.49×10^{-04}	1.33×10^{-03}	1.51×10^{-03}	1.90×10^{-04}	1.64×10^{-04}
		CV-MAE	2.39×10^{-04}	9.22×10^{-04}	1.04×10^{-03}	1.30×10^{-04}	8.96×10^{-05}
	RBF	CV-RMSE	2.95×10^{-03}	1.05×10^{-03}	8.28×10^{-05}	1.52×10^{-05}	9.77×10^{-6}
		CV-MAE	2.07×10^{-03}	7.33×10^{-04}	5.65×10^{-05}	1.20×10^{-05}	6.93×10^{-6}
1	RM	CV-RMSE	9.18×10^{-03}	1.15×10^{-02}	2.33×10^{-02}	4.77×10^{-03}	1.98×10^{-02}
		CV-MAE	6.73×10^{-03}	9.24×10^{-03}	1.76×10^{-02}	3.39×10^{-03}	1.30×10^{-02}
	Kriging	CV-RMSE	3.46×10^{-04}	1.33×10^{-03}	1.50×10^{-03}	1.88×10^{-04}	1.63×10^{-04}
		CV-MAE	2.37×10^{-04}	9.18×10^{-04}	1.03×10^{-03}	1.28×10^{-04}	8.93×10^{-05}
	RBF	CV-RMSE	9.82×10^{-03}	6.40×10^{-04}	5.50×10^{-04}	1.30×10^{-05}	4.04×10^{-05}
		CV-MAE	6.30×10^{-03}	4.36×10^{-04}	3.74×10^{-04}	1.08×10^{-05}	2.81×10^{-05}
2	RM	CV-RMSE	9.14×10^{-03}	1.15×10^{-02}	2.32×10^{-02}	4.71×10^{-03}	1.97×10^{-02}
		CV-MAE	6.70×10^{-03}	9.19×10^{-03}	1.75×10^{-02}	3.34×10^{-03}	1.29×10^{-02}
	Kriging	CV-RMSE	3.44×10^{-04}	1.32×10^{-03}	1.49×10^{-03}	1.87×10^{-04}	1.62×10^{-04}
		CV-MAE	2.35×10^{-04}	9.15×10^{-04}	1.03×10^{-03}	1.27×10^{-04}	8.87×10^{-05}
	RBF	CV-RMSE	9.62×10^{-03}	6.18×10^{-04}	5.46×10^{-04}	1.28×10^{-05}	3.98×10^{-05}
		CV-MAE	7.10×10^{-03}	4.30×10^{-04}	3.73×10^{-04}	1.07×10^{-05}	2.76×10^{-05}

Table 8-5: CV-RMSE and CV-MAE estimates for each metamodel at three services times for the settling angle response.

Time	**Model**	**Predictive Error Method**	**Settling Angle**
			$\mu(K_R) = 0.004$
0 year	RM	CV-RMSE	3.32×10^{-05}
		CV-MAE	2.15×10^{-05}
	Kriging	CV-RMSE	2.76×10^{-07}
		CV-MAE	1.46×10^{-07}
	RBF	CV-RMSE	7.21×10^{-07}
		CV-MAE	4.86×10^{-06}
1 year	RM	CV-RMSE	3.32×10^{-05}
		CV-MAE	2.16×10^{-05}
	Kriging	CV-RMSE	2.65×10^{-07}
		CV-MAE	1.38×10^{-07}
	RBF	CV-RMSE	4.69×10^{-05}
		CV-MAE	3.09×10^{-05}
2 years	RM	CV-RMSE	3.318×10^{-05}
		CV-MAE	2.16×10^{-05}
	Kriging	CV-RMSE	2.53×10^{-07}
		CV-MAE	1.31×10^{-07}
	RBF	CV-RMSE	4.29×10^{-05}
		CV-MAE	2.88×10^{-05}

Figure 8-7 shows the error between the FORM and MCS estimates of probability. This error is calculated as follows

$$\%\hat{\varepsilon}_{FORM} = \left(\frac{[\mathbf{F}_l]_{FORM} - [\mathbf{F}_l]_{MCS}}{[\mathbf{F}_l]_{MCS}} \right) \times 100 \tag{8.8}$$

Since the LSSs are linear, this error should, ideally, be zero. The error also depends on the number of samples used in MCS. From Figure 8-7, the error starts off being large for all metamodels then reduces. This occurs because at the start of the product lifetime, probabilities are around the order of 10^{-3} (refer to Table 8-3) and following (8.8), dividing by a small number leads to a large percentage error. A comparison of the estimated *Cdf* shows that the errors from the metamodels are small. As the failure probability gets larger this error is reduced. From Figure 8-7, FORM errors are smallest with the RM.

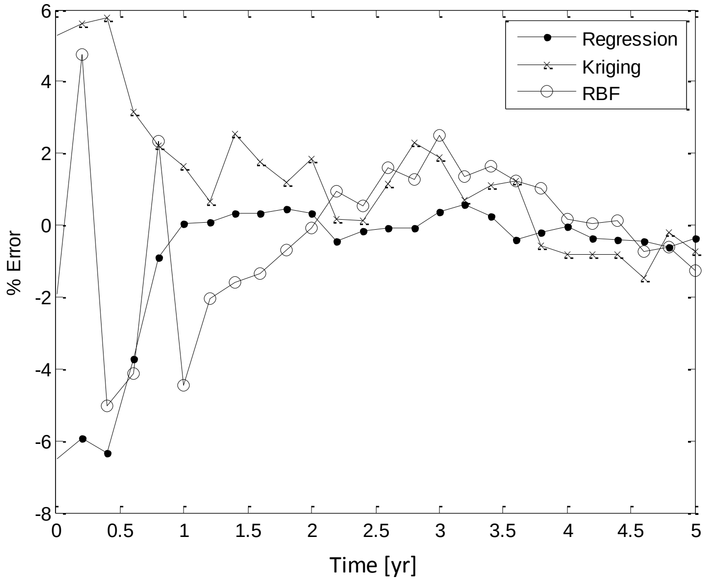


Figure 8-7: The %error between FORM and MCS probability estimates.

8.3.2 Experiment 2: μ(K$_R$) = 0.8%

For the second degradation rate, a new training design is required for v_3. Using the same μ ± 4.5σ range, the 5 levels for x_1 and x_2 remain the same. However, since the mean of the degradation rate is now different, 5 new levels for x_3 are obtained.

$$\begin{bmatrix} x_1^L \\ x_1^{LN} \\ x_1^N \\ x_1^{HN} \\ x_1^H \end{bmatrix} = \begin{bmatrix} 4.966 \\ 5.083 \\ 5.200 \\ 5.317 \\ 5.434 \end{bmatrix} \quad \begin{bmatrix} x_2^L \\ x_2^{LN} \\ x_2^N \\ x_2^{HN} \\ x_2^H \end{bmatrix} = \begin{bmatrix} 3.82 \\ 3.91 \\ 4.00 \\ 4.09 \\ 4.18 \end{bmatrix} \quad \begin{bmatrix} x_3^L \\ x_3^{LN} \\ x_3^N \\ x_3^{HN} \\ x_3^H \end{bmatrix} = \begin{bmatrix} 0.00764 \\ 0.00782 \\ 0.008 \\ 0.00818 \\ 0.00836 \end{bmatrix}$$

A new matrix of responses is obtained at the new training design and metamodels are again fit. Table 8-6 shows the instantaneous failure probability for each metamodel using FORM and MCS. A larger degradation rate would imply a shorter time to failure. Therefore, the probability at each time increment should be greater than the corresponding probability when the degradation rate mean is 0.4%. From Table 8-6, a quick observation shows that this is true since the failure at the end of 5 years is 0.5332 instead of 0.1041 in the previous example. Figures 8-8 and 8-9 show the error estimates when FORM and MCS are used to estimate probability.

Table 8-6: The cumulative system failure probability when $\mu_{kR} = 0.008$.

Time	Mechanistic	RM		Kriging		RBF	
	MCS	FORM	MCS	FORM	MCS	FORM	MCS
0	0.0039	0.0038	0.0038	0.0040	0.0040	0.0040	0.0041
0.2	0.0051	0.0047	0.0048	0.0056	0.0057	0.0055	0.0058
0.4	0.0075	0.0065	0.0065	0.0076	0.0077	0.0075	0.0078
0.6	0.0095	0.0089	0.00903	0.0102	0.0105	0.0101	0.0107
0.8	0.0128	0.0119	0.0121	0.0136	0.0137	0.0134	0.0139
1	0.0171	0.0158	0.0158	0.0180	0.0179	0.0177	0.0182
1.2	0.0222	0.0208	0.0208	0.0234	0.0233	0.0231	0.0236
1.4	0.0298	0.0271	0.0268	0.0302	0.0301	0.0298	0.0306
1.6	0.0391	0.0348	0.0349	0.0384	0.0384	0.0381	0.0390
1.8	0.0495	0.0443	0.0444	0.0485	0.0487	0.0481	0.0493
2	0.0621	0.0556	0.05611	0.0606	0.0609	0.0600	0.0614
2.2	0.0772	0.0693	0.0700	0.0748	0.0753	0.0742	0.0760
2.4	0.095	0.0853	0.08543	0.0914	0.0913	0.0907	0.0921
2.6	0.1141	0.1040	0.1041	0.1106	0.1104	0.1099	0.1115
2.8	0.1336	0.1254	0.1257	0.1324	0.1324	0.1317	0.1335
3	0.1591	0.1497	0.1501	0.1570	0.1572	0.1562	0.1585
3.2	0.1879	0.1769	0.1770	0.1845	0.1843	0.1836	0.1859
3.4	0.2169	0.2070	0.2076	0.2146	0.2145	0.2138	0.2163
3.6	0.2511	0.2400	0.2408	0.2474	0.2478	0.2465	0.2495
3.8	0.2865	0.2755	0.2761	0.2827	0.2829	0.2818	0.2846
4	0.3229	0.3134	0.3142	0.3201	0.3204	0.3192	0.3221
4.2	0.3594	0.3534	0.3534	0.3594	0.3592	0.3585	0.3612
4.4	0.4043	0.3949	0.3959	0.4002	0.4008	0.3994	0.4028
4.6	0.4454	0.4377	0.4383	0.4420	0.4422	0.4412	0.4442
4.8	0.4884	0.4811	0.4807	0.4844	0.4837	0.4837	0.4858
5	0.5332	0.5246	0.5242	0.5270	0.5264	0.5262	0.5283

From Figure 8-8, the FORM-based probability estimates are the worst with the RM. Again, failure estimates are much better using either Kriging or the RBF.

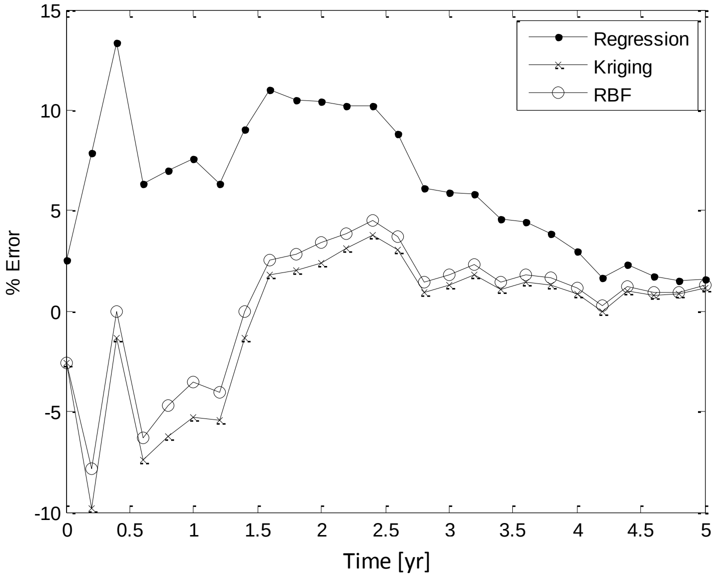


Figure 8-8: The error between the FORM estimate, using each metamodel, and the MCS of the mechanistic model.

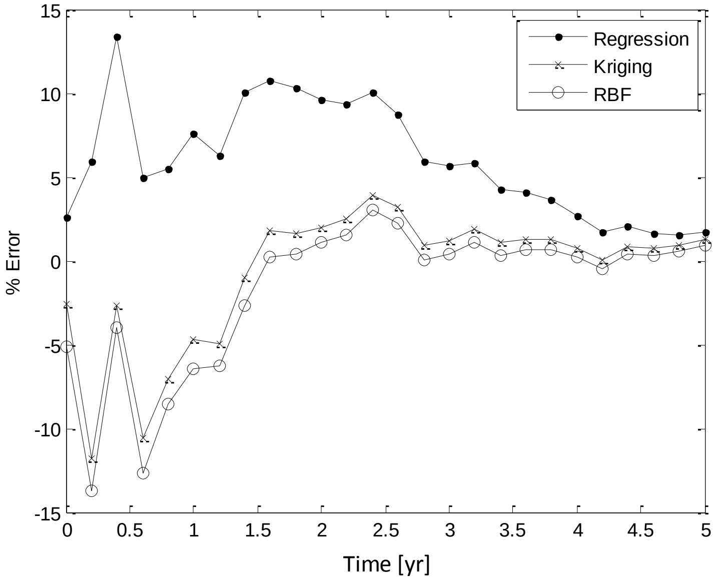


Figure 8-9: The error between MCS-based metamodel and MCS-based mechanistic model probability estimates.

As in the previous example, the percentage error between FORM and MCS probability estimates is plotted as shown in Figure 8-10. Here, the largest errors occur for the RBF. After 2.5 years, the %error is approximately consistent and not as erratic as the start of the product lifetime.

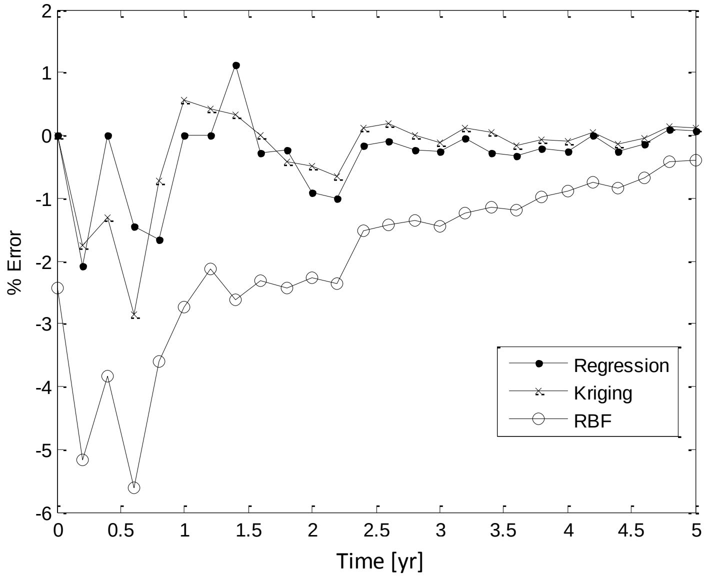


Figure 8-10: The %error between FORM and MCS failure estimates.

The *Cdf* is plotted using both FORM and MCS probability estimates and shown in Figures 8-11 and 8-12. This time, all metamodels provide very accurate plots. The *Cdf* estimates for the two degradation rates are shown in Figure 8-13. This figure is useful since it shows how a larger degradation rate results in a larger failure probability at the end of the service time.

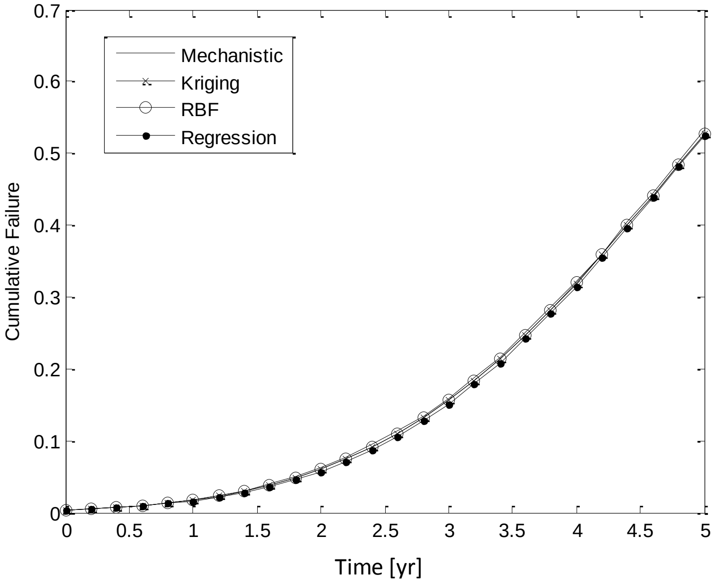


Figure 8-11: The *Cdf* estimate using FORM and each metamodel.

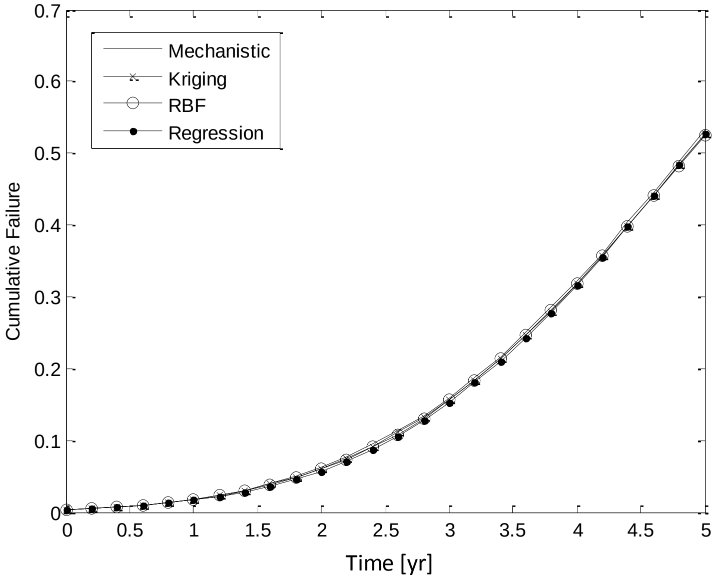


Figure 8-12: The *Cdf* estimate using MCS and each metamodel.

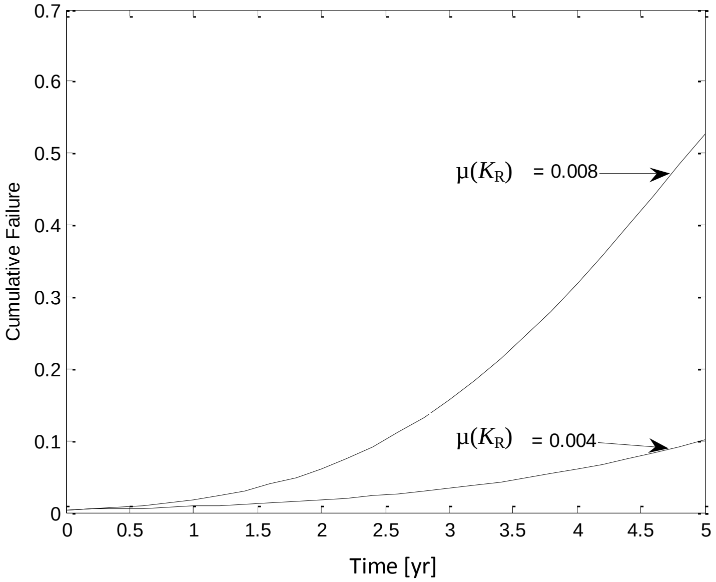


Figure 8-13: The *Cdf* estimates for the two degradation rates.

Finally, the CPU time required to build to *Cdfs* are shown in Table 8-7. From the table, the metamodel with FORM or MCS is much faster than using the Mechanistic Model. An increased speed is observed using MCS with the metamodels especially in the case of the RBF.

Table 8-7: The computation time for the various methods.

Model	Computation time to build CDF
Mechanistic Model and MC (100,000 runs)	60,000s
Metamodel and FORM RM Kriging RBF	 115s 550s 2000s
Metamodel and MCS (100,000 runs) RM Kriging RBF	 100s 450s 200s

Error Analysis for experiment 2

The fit of the metamodels is checked in a similar way as in section 8.2.1. Tables 8-8 and 8-9 show the CV-RMSE and CV-MAE estimates for each metamodel at three service times for $\mathbf{D}_\omega$ and $\boldsymbol{\theta}_s$. As before, Kriging and RBF are better than the RM. For the settling angle response, Kriging is almost always better.

Table 8-8: The CV-RMSE and CV-MAE estimates for each metamodel, built for each column of $\mathbf{D}_{\omega}$. at three service time increments,

Method	Time	Model	Column of $\mathbf{D}_\omega$				
			1	2	3	4	5
0	RM	CV-RMSE	9.22×10^{-03}	1.16×10^{-02}	2.34×10^{-02}	4.85×10^{-03}	1.99×10^{-02}
		CV-MAE	6.76×10^{-03}	9.28×10^{-03}	1.76×10^{-02}	3.43×10^{-03}	1.30×10^{-02}
	Kriging	CV-RMSE	3.49×10^{-04}	1.33×10^{-03}	1.51×10^{-03}	1.90×10^{-04}	1.64×10^{-4}
		CV-MAE	2.39×10^{-04}	9.22×10^{-04}	1.04×10^{-03}	1.30×10^{-04}	8.96×10^{-04}
	RBF	CV-RMSE	2.95×10^{-03}	1.05×10^{-03}	8.28×10^{-05}	1.52×10^{-05}	9.77×10^{-6}
		CV-MAE	2.07×10^{-03}	7.34×10^{-04}	5.65×10^{-05}	1.20×10^{-05}	6.93×10^{-6}
1	RM	CV-RMSE	9.14×10^{-03}	1.15×10^{-02}	2.32×10^{-02}	4.71×10^{-03}	1.97×10^{-02}
		CV-MAE	6.70×10^{-03}	9.19×10^{-03}	1.75×10^{-02}	3.34×10^{-03}	1.29×10^{-02}
	Kriging	CV-RMSE	3.45×10^{-04}	1.32×10^{-03}	1.49×10^{-03}	1.87×10^{-04}	1.62×10^{-04}
		CV-MAE	2.36×10^{-04}	9.16×10^{-04}	1.03×10^{-03}	1.28×10^{-04}	8.84×10^{-05}
	RBF	CV-RMSE	7.46×10^{-03}	5.88×10^{-04}	5.15×10^{-04}	1.46×10^{-05}	3.78×10^{-05}
		CV-MAE	4.74×10^{-03}	3.91×10^{-04}	3.44×10^{-04}	9.26×10^{-06}	2.51×10^{-05}

2	RM	CV-RMSE	9.06×10^{-03}	1.14×10^{-02}	2.31×10^{-02}	4.57×10^{-03}	1.95×10^{-02}
		CV-MAE	6.64×10^{-03}	9.10×10^{-03}	1.74×10^{-02}	3.25×10^{-03}	1.28×10^{-02}
	Kriging	CV-RMSE	3.41×10^{-04}	1.32×10^{-03}	1.48×10^{-03}	1.83×10^{-04}	1.60×10^{-04}
		CV-MAE	2.33×10^{-04}	9.10×10^{-04}	1.02×10^{-03}	1.25×10^{-04}	8.73×10^{-05}
	RBF	CV-RMSE	7.45×10^{-03}	5.85×10^{-04}	5.11×10^{-04}	1.42×10^{-05}	3.67×10^{-05}
		CV-MAE	4.73×10^{-03}	3.98×10^{-04}	3.42×10^{-04}	8.99×10^{-06}	2.43×10^{-05}

Table 8-9: The CV-RMSE and CV-MAE estimates for each metamodel built to model the settling angle response.

Time	**Model**	**Prediction Error method**	**Settling Angle**
			μ(K_R) = 0.008
0 year	RM	CV-RMSE	3.32×10^{-05}
		CV-MAE	2.15×10^{-05}
	Kriging	CV-RMSE	2.76×10^{-07}
		CV-MAE	1.46×10^{-07}
	RBF	CV-RMSE	7.21×10^{-06}
		CV-MAE	4.86×10^{-06}
1 year	RM	CV-RMSE	3.32×10^{-05}
		CV-MAE	2.16×10^{-05}
	Kriging	CV-RMSE	2.52×10^{-07}
		CV-MAE	1.32×10^{-07}
	RBF	CV-RMSE	3.43×10^{-05}
		CV-MAE	2.18×10^{-05}
2 years	RM	CV-RMSE	4.29×10^{-05}
		CV-MAE	2.88×10^{-05}
	Kriging	CV-RMSE	2.30×10^{-07}
		CV-MAE	1.21×10^{-07}
	RBF	CV-RMSE	2.30×10^{-07}
		CV-MAE	1.21×10^{-07}

8.4 An Investigation of Various Training Designs

In order to choose the training design, a range is selected which is then broken into a series of “levels”. The range is chosen using the relation $\mu_i \pm r_i\sigma_i$ where $r_i = 3, 4.5, 6$ or 10. Within the upper and lower boundaries specified by the aforementioned relation, the number of “levels” is chosen. These levels are training samples selected to make up the training design and follows from the specific design of experiments technique chosen. For example, if $\lambda = 3$, then the training site of each design variable is $\mu_i - r_i\sigma_i$, μ_i and $\mu_i + r_i\sigma_i$. In this example, 4 different ranges are investigated along with 2 values of λ; 3 and 5.

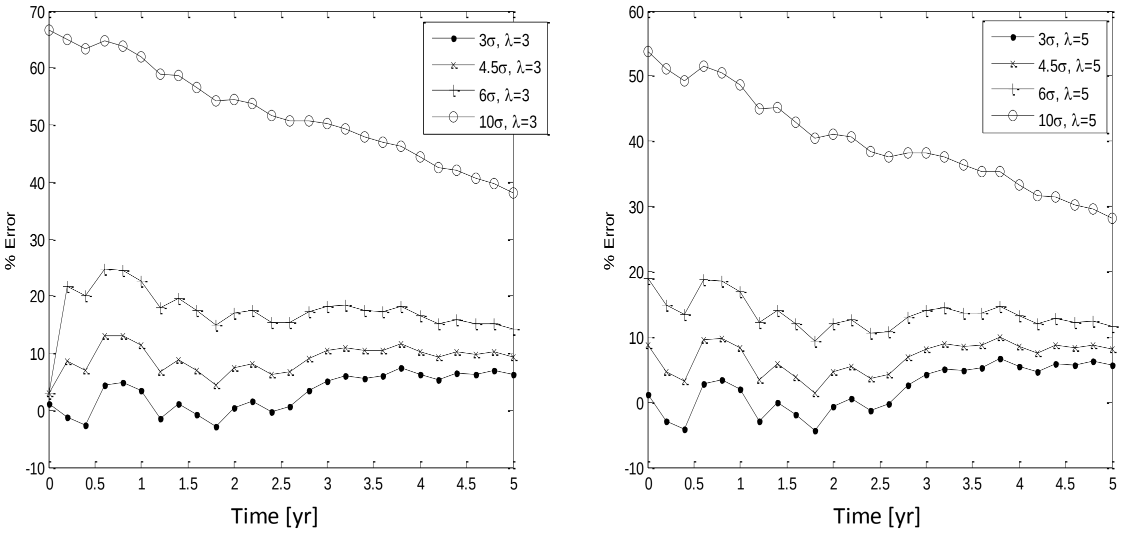
At each of the eight training designs (Appendix C), the cumulative failure probability is plotted and the accuracy and speed of estimating the CDF, using each metamodel, is compared. The failure probabilities are estimated using both FORM and the MCS. The results for each metamodel are first discussed independently and then an overview of all the results is presented.

Regression Model

At each of the eight training design, a new Regression Model is fit. FORM and MCS are then used to estimate the failure probability and the results are compared with the mechanistic model failure estimates. Figure 8-14 show the error estimates when FORM is used to estimate probability. Since the results from all eight training designs were not able to fit on one graph, two graphs are shown. Figure 8-14(a) shows the percentage error when 3 levels are used and Figure 8-14(b) shows the percentage error when 5 levels are used with the four ranges. The percentage error is estimated as

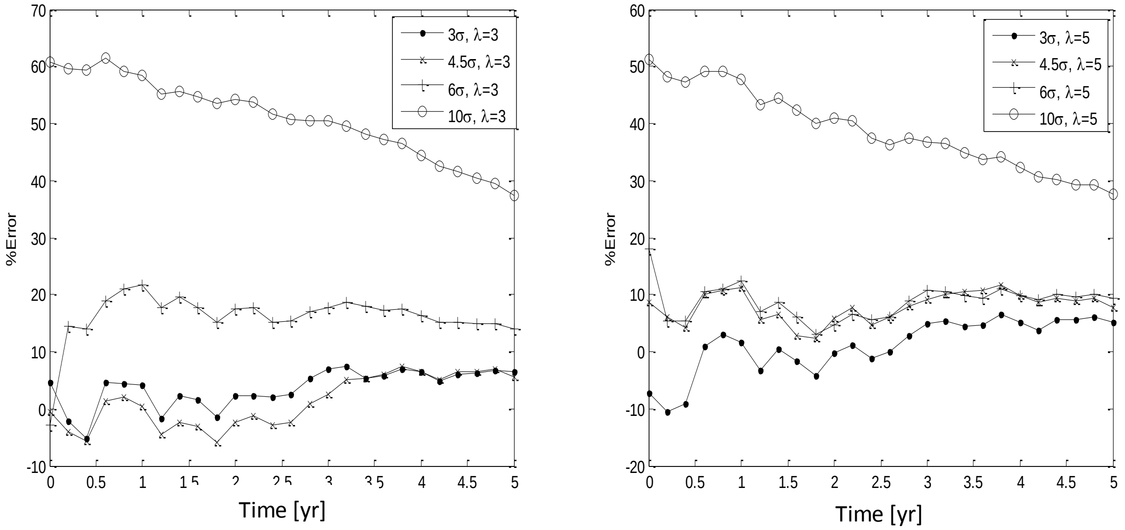
$$(\%Error)_l = \left(\frac{[\mathbf{F}_l]_{\text{Mechanistic}} - [\mathbf{F}_l]_{\text{Metamodel}}}{[\mathbf{F}_l]_{\text{Mechanistic}}} \right) \times 100$$

From Figure 8-14, as the range increases, the accuracy decreases. Also, a lower percentage error is observed when the number of levels is increased to 5 for the two larger ranges (6σ and 10σ). When MCS is used to estimate probability, the results are similar to those obtained using FORM. When 3 levels are used the accuracy starts to get very poor at the 6σ range and higher. However, when the number of levels is increased to 5, a 6σ range produces a percentage error of 10% as opposed to about 15% when 3 levels are used. Similar results are observed at the 10σ range; an increase in the number of levels results in improved accuracy.



(a) **(b)**

Figure 8-14: The %error of the probability estimate, using FORM, for various possible training designs. (a) 3 Levels and (b) 5 levels.



(a) **(b)**

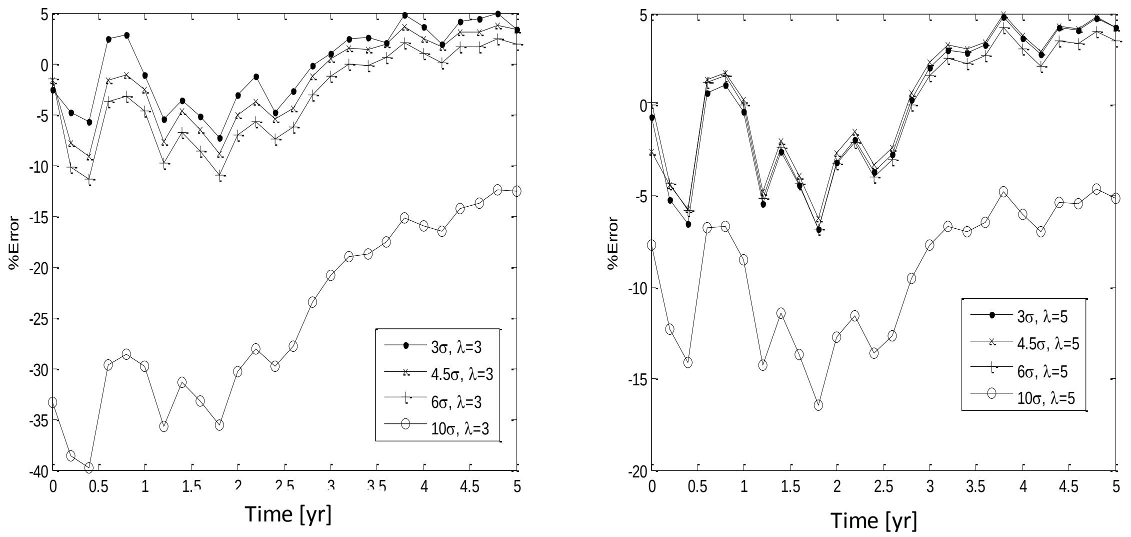
Figure 8-15: The error of the probability estimate, using MCS, at each service time increments for various possible training designs. (a) 3 Levels and (b) 5 Levels.

The process is repeated for the Kriging and RBF metamodels.

Kriging Metamodel

Now, a Kriging metamodel is fit for the responses at each training design and the error estimates using FORM and MCS for the mean degradation rate, K_R = 0.4%, is plotted. Figure 8-16 shows the percentage errors when FORM is used to estimate failure probability and Figure 8-17 shows the errors when MCS is used to estimate probability.

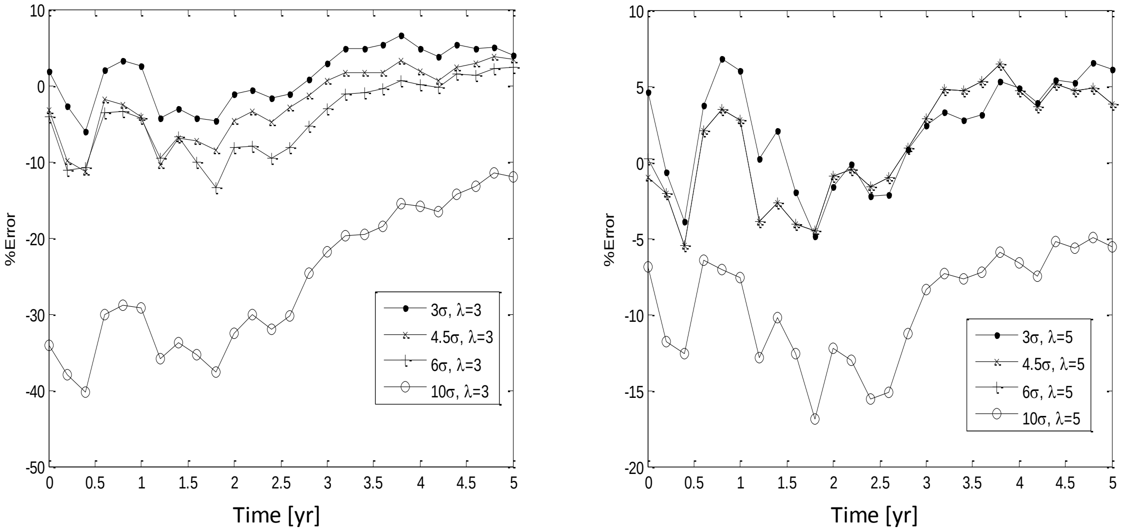
From Figure 8-16, the largest errors are observed at a 10σ range. Unlike the Regression Model, at a 10σ range training design, Kriging has become conservative since the errors are now negative. Like the Regression Model, an increase in accuracy for the 10σ range is obtained by increasing the number of levels to 5. When 5 levels are used, the accuracy among the 3 lower ranges is very small.



(a) (b)

Figure 8-16: The %error when FORM is used with Kriging, trained by each possible training design, to estimate probability. (a) 4 ranges each with 3 Levels (b) 4 ranges each with 5 Levels

When a MCS is used to estimate probability, the results are about the same as those obtained from FORM. For the 10σ range, the accuracy increases when the number of levels increases to 5.

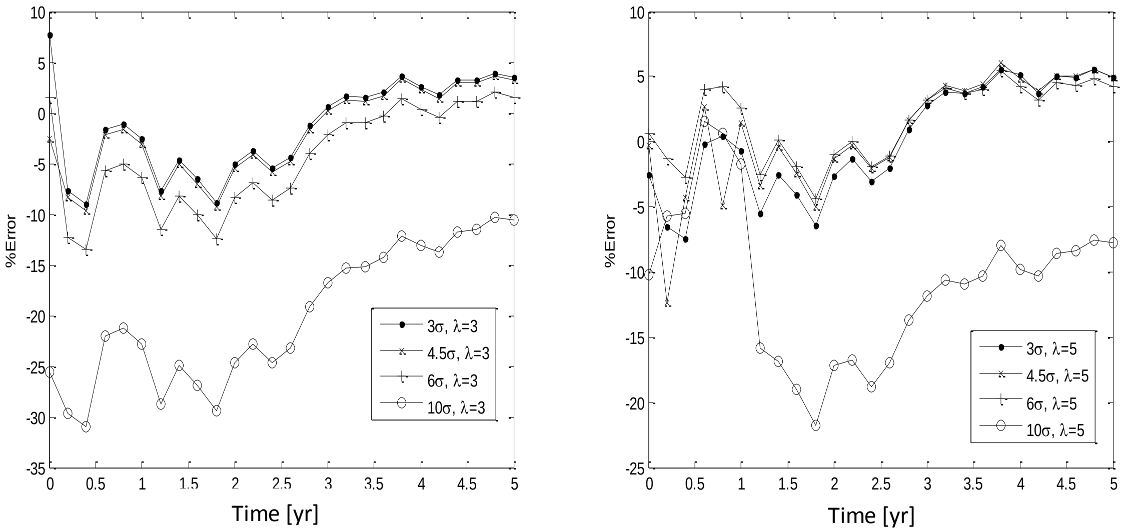


(a) (b)

Figure 8-17: The %error when MCS is used with Kriging, trained by the eight possible training designs, to estimate probability. (a) 4 ranges each with 3 Levels and (b) 4 ranges each with 5 levels.

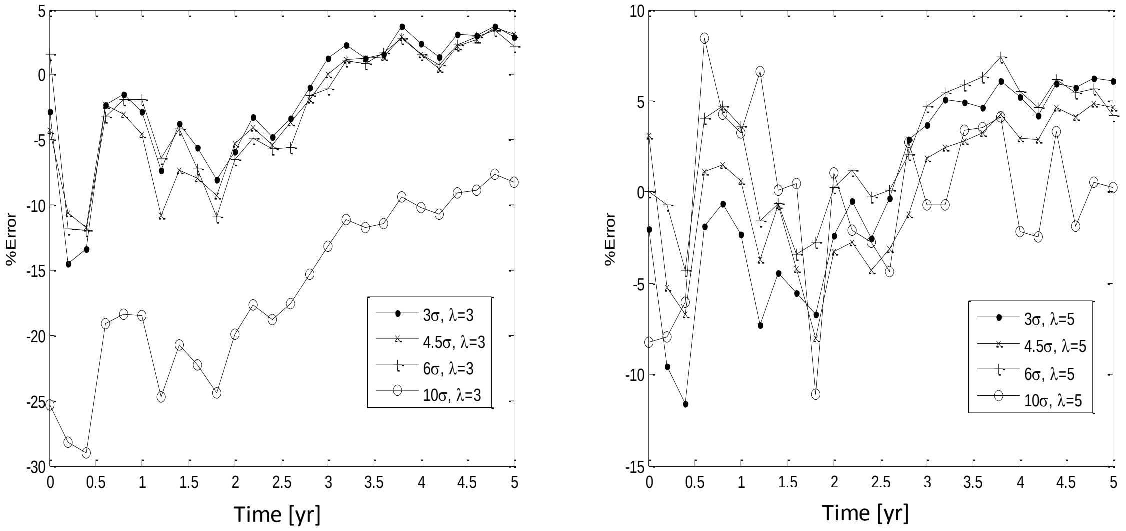
RBF Metamodel

For the final metamodel, the results are shown in Figures 8-18 and 8-19. These results are close to those obtained from Kriging with conservative failure estimates at the 10σ range.



(a) (b)

Figure 8-18: The %error when FORM is used with the RBF, trained by each possible training design, to estimate probability. (a) 4 ranges each with 3 Levels (b) 4 ranges each with 5 Levels



(a) (b)

Figure 8-19: The %error when MCS is used with Kriging, trained by the eight possible training designs, to estimate probability. (a) 4 ranges each with 3 Levels and (b) 4 ranges each with 5 levels.

Finally, the speed of the metamodels at the various training designs is tested. For all metamodels, the speed depends on the size of the matrix of training sets which depends on the number of levels. This is especially true for Kriging and the RBF that uses the training sets to build the metamodel. Therefore, the speed of the metamodels is recorded at the two test levels and compared. Table 8-10 shows these results. When FORM is used to estimate probability, the slowest metamodel is the RBF with the Regression Model being the fastest. For MCS-based probability estimates, the fastest metamodel is still the Regression Model but now Kriging and the RBF is the same speed.

Table 8-10: The computation time for the two levels.

Metamodel	**Levels**	**Time**	
		Metamodel and FORM	**Metamodel and MCS**
RM	$\lambda = 3$	100s	40s
	$\lambda = 5$	110s	100s
Kriging	$\lambda = 3$	200s	80s
	$\lambda = 5$	600s	400s
RBF	$\lambda = 3$	250s	80
	$\lambda = 5$	2000s	400s

8.4.1 Summary

For this degradation modelling problem, errors likely come from four sources; using SVD to approximate the performance measure, fitting the metamodel, using FORM to estimate failure probability and using the set theory method to estimate the *Cdf*. A discussion of these errors has been presented at the end of each experiment.

In summary, the error due to SVD has been controlled by comparing the estimate of $\mathbf{\Omega}$ or $\mathbf{\Theta}$ using dominant singular values with the true value. This error has been shown to be very small and is not significant. Since the metamodel is an approximate model of the mechanistic model, it is very difficult to completely eliminate this error. The only way is to reduce this error by either choosing a different metamodel or a different training design. Among the three metamodels, Kriging was the best followed closely by the RBF and then the Regression Model. These errors have been quantified using cross-validation to estimate the RMSE and MAE. To further compare the accuracy of the metamodels, the angular speed response is plotted, at the means of the design variables, using each metamodel. From Figure 8-20, the metamodels produce an indistinguishable response estimate. Further inspection of the error, at each cycle-time increment is shown in Figure 8-21 and it can be seen that the smallest errors occur when Kriging is used.

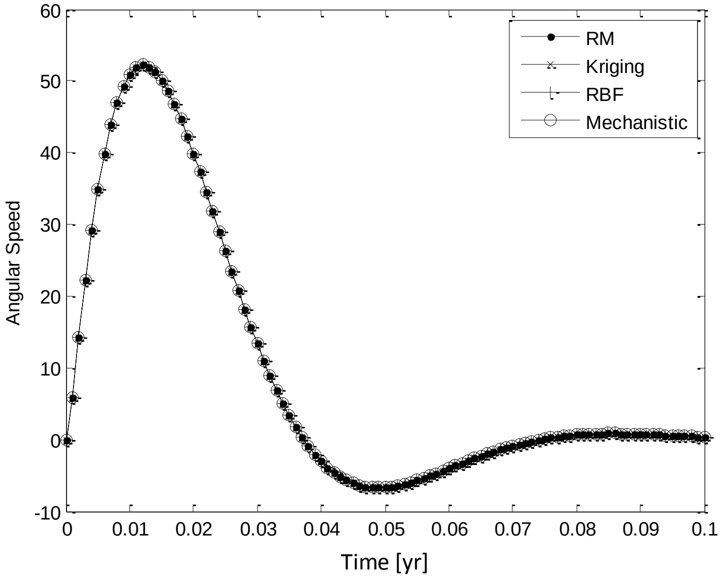


Figure 8-20: The angular speed response estimate, at the mean of each design variable, using each metamodel.

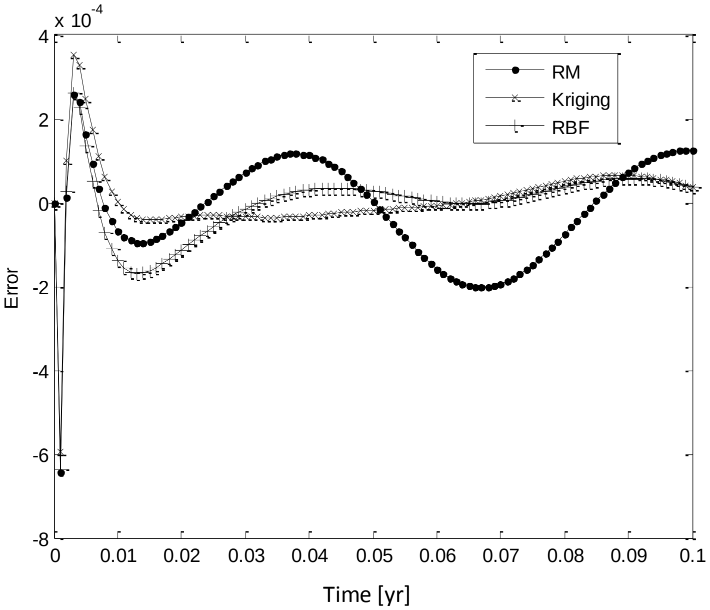


Figure 8-21: The error, at each cycle-time increment, of the angular speed response estimate using each metamodel.

Another source of error occurs when using FORM to estimate failure probability. A comparison of the FORM results with estimates obtained using the MCS show that the error due to FORM is small. The final source of error is from the set-theory method. For this problem, since the LSS does not turn but moves in the same direction as the component "fails", the incremental failure region is just

$$\Pr(\mathbf{B}_l) \leq \Pr(\mathbf{F}_{l+1}) - \Pr(\mathbf{F}_l)$$

Following the discussion in Chapter 7, since the movement of the failure surfaces is in a constant spatial direction, $\rho_i(t_l,t_{l+1}) = 1$ and $\varepsilon_3(t_l,t_{l+1}) = 0$. Any service time increment is suitable but a smaller Δt produces a smoother *Cdf*. Now, a marginal distribution model is used to plot the *Cdf*.

8.5 Degradation Modelling using the Marginal Distribution Model.

Another type of model that exists to model degradation of the response is the marginal distribution model. In this model, degradation is assumed to exist in the parameters of the design variables. Following the examples found in (Son, 2006) involving the distribution-based degradation model, for a variable V_i

$$\mu_i(t) = \mu_0(1 + K_i t) \tag{8.9}$$

$$\sigma_i(t) = \sigma_0(1 + K_i t) \tag{8.10}$$

Where $\sigma_i(t) = (tol_i\mu_i)/300$. As in the previous example, μ_1 =5.2 and μ_2 = 4.0 $tol_1 = tol_2$ = 3%. Two constant deterministic degradation rates, 0.4% and 0.8%, are investigated. To develop the training design, a 4.5σ range is used with λ = 5 to obtain a total of 25 unique design variable sets. As before, the service time increment is Δt = 0.2years.

8.5.1 Degradation rate = 0.4%

Figure 8-22 shows the error due to using FORM to estimate probability for each metamodel. This error is computed as before. The Regression Model results in a higher error than those found using either Kriging or the RBF. The Kriging and RBF errors are very similar, in some cases Kriging is better (3.5years and higher) and sometimes the RBF is better (0.5years to 2years). The RM produces conservative probabilities while Kriging and the RBF starts being optimistic then become conservative. The conservative estimates, however, are very small (less than 4%). The percentage error is also estimated when MCS is used to estimate failure probability and is shown in Figure 8-23.

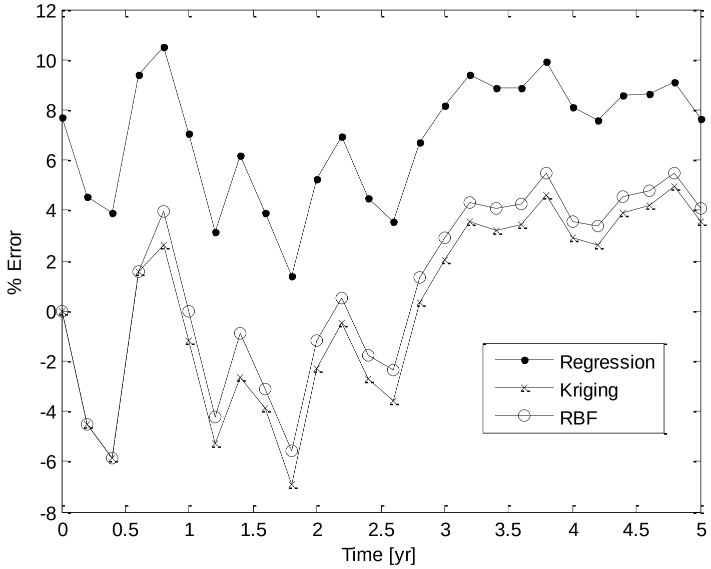


Figure 8-22: The %error when FORM is used to estimate failure probability.

Similar results to FORM estimates are found when MCS is used. The Regression Model produces the largest errors and Kriging and the RBF are close. The absolute error of Kriging and RBF estimates are less than 10% which is good for design. The MCS estimates follow a trend very close to FORM which indicates the probability estimates are similar. This is means that the error due to using FORM to estimate probability is small which is desirable if FORM is to be used for design.

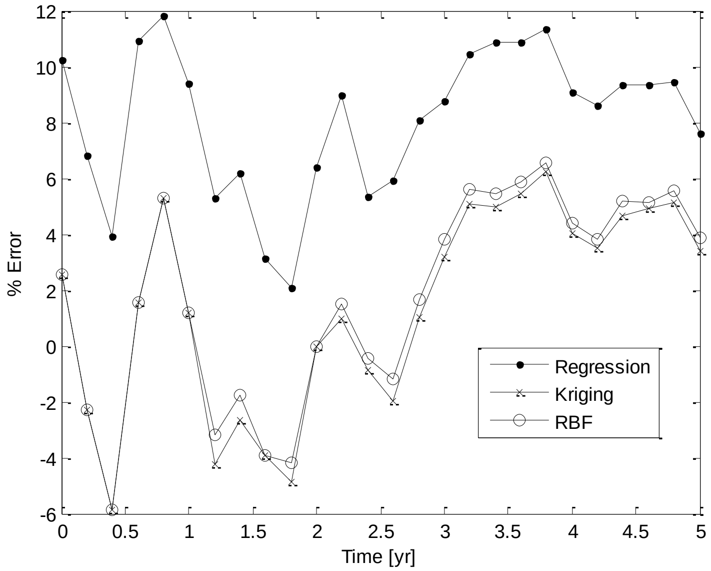


Figure 8-23: The %error when MCS is used to estimate failure probability.

Figure 8-24 shows the error due to FORM computed as the difference between FORM probability of failure estimates and MCS probability estimates. The errors are generally small since the worse case is only -3%.

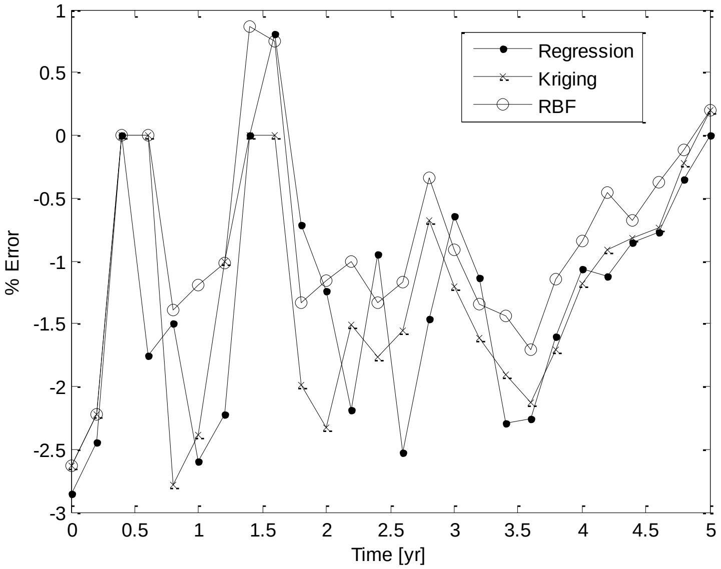


Figure 8-24: The %error between FORM and MCS probability estimates.

The *Cdf* is shown in Figure 8-25 and is similar to the degradation path model. The *Cdf* reflects the error results observed in Figures 8-22 and 8-23 where the Regression Model produces the largest error. From Figure 8-24, this is again observed as the *Cdf* curve from the regression falls below Kriging and RBF that are closer to the mechanistic model results. The CPU times using the various methods to estimate the *Cdf* is shown in Table 8-11.

From Table 8-11, using the metamodels with either FORM or MCS is more than 100 times faster than using the mechanistic model. When the metamodels are used, between FORM and MCS, MCS is even faster for Kriging and the RBF.

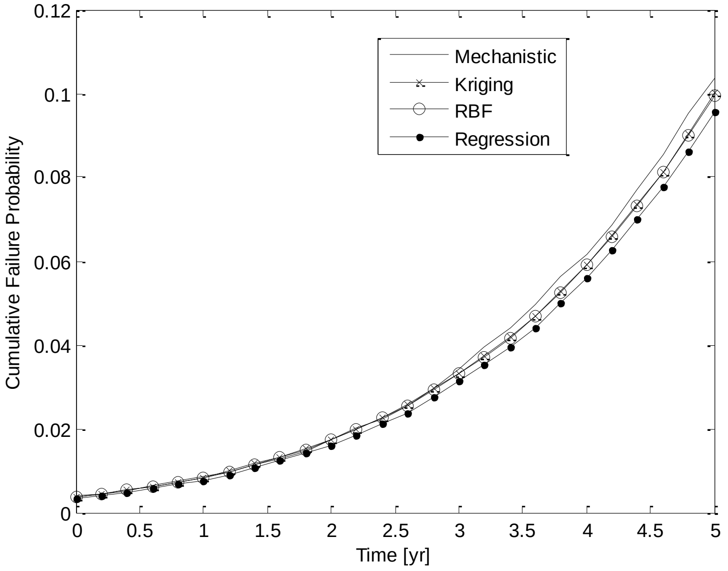


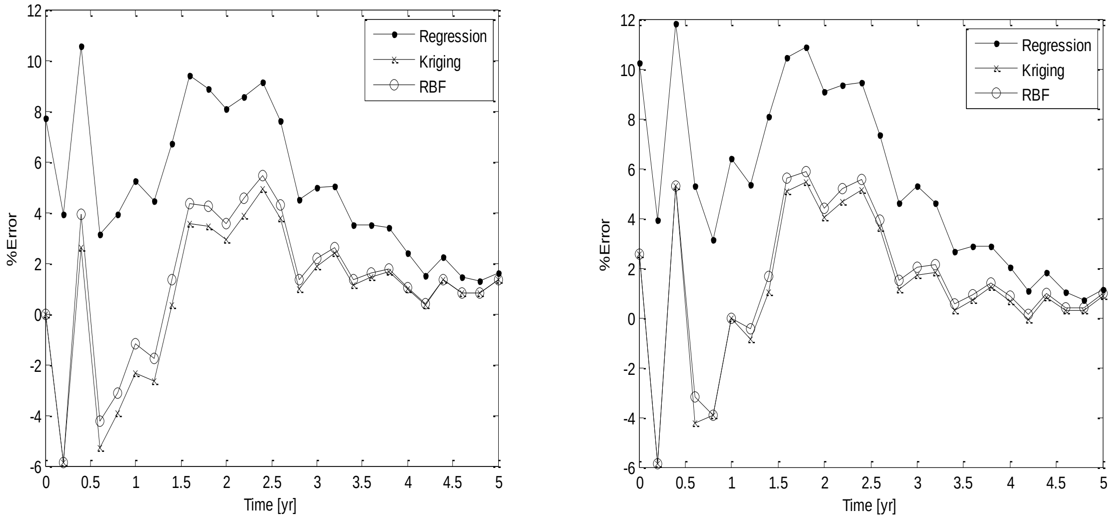
Figure 8-25: The *Cdf* estimated using MCS of each metamodel and the mechanistic model when K_R = 0.004.

Table 8-11: The CPU time comparison.

Model	**Time to build CDF**
Mechanistic with MC (100,000 runs)	60,000s
Metamodel with FORM	
RM	37s
Kriging	120s
RBF	150s
Metamodel with MC (100,000 runs)	
RM	35s
Kriging	60s
RBF	70s

8.5.2 Degradation rate = 0.8%.

A second degradation rate, K = 0.8%, is also investigated. Errors are computed when the metamodel with FORM is used to estimate failure probability and when the metamodel is used with MCS to estimate probability. These results are shown in Figures 8-26 and 8-27. As in the previous examples, the least accuracy comes from the Regression Model and Kriging and the RBF is about the same. This pattern is observed when both FORM and MCS is used to estimate failure probability.



(a)

(b)

Figure 8-26: The %error of the (a) FORM-based metamodel and (b) MCS-based metamodel estimates of failure probability.

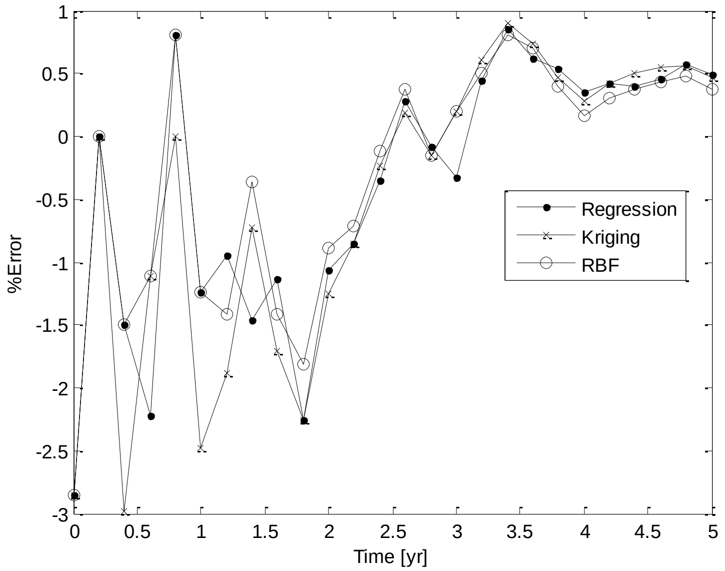


Figure 8-27: The %error between FORM and MCS probability estimates.

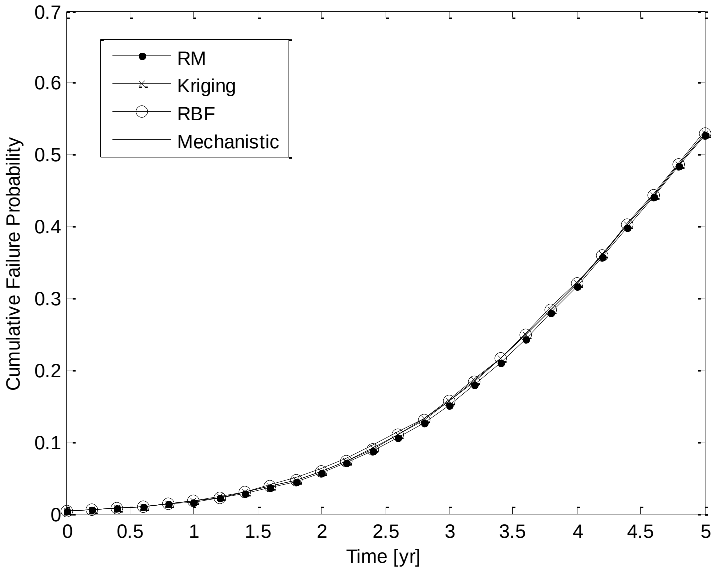


Figure 8-28: The *Cdf* estimated using MCS of each metamodel and the mechanistic model when K_R = 0.008.

The CPU time comparison is the same as those shown in Table 8-11. Notice, these times are faster than those obtained when the degradation path model is used. This is because, for the distribution-based model, only two random variables are involved. Like the previous example, the largest CPU time occurs when the mechanistic model has been used, with MCS, to estimate failure probability. When FORM is used with the metamodels, the Regression Model, again, results in the lowest CPU time with RBF being the longest. This time, the difference in computation time between RBF and Kriging is not as large as in the previous example. A plot of the LSSs as they move through time is shown in Figure 8-29.

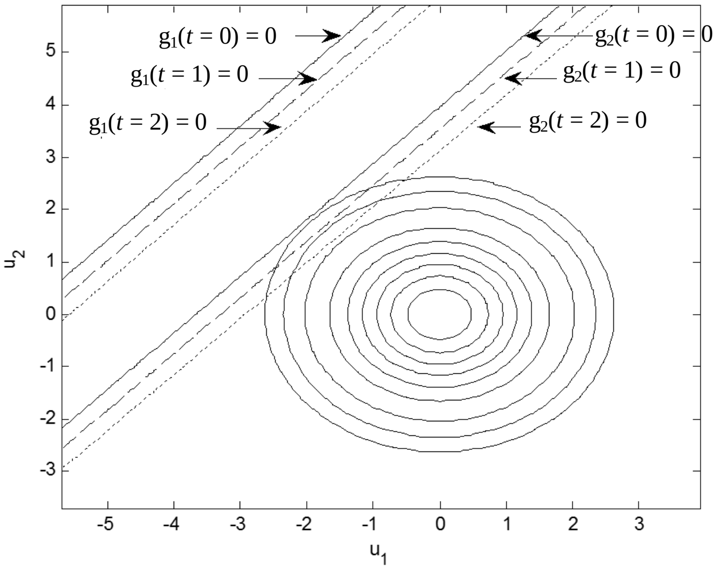


Figure 8-29: The LSSs, estimate using the Regression Model, as they move through time.

8.6 Conclusions

This chapter has shown how metamodels are combined with set-theory to build a *Cdf* of a dynamic degrading position-control servo mechanism. Two types of degradation models have been used to model degradation and, for each model-type, two possible degradation rates have been chosen. Using several possible training designs, it has been found that the choice of training design can affect the accuracy of the metamodels. Also, the speed of FORM with the metamodels, MCS with the metamodels and MCS of the mechanistic model has been compared. Based on the results, metamodels provide a substantial decrease in computation time. Times for probability calculations with FORM and MCS are metamodel dependent. In general, our vector form of the metamodels has resulted in faster computation times using MCS than FORM. For all experiments, Kriging and RBF have provided more accurate results than the RM. Two different degradation rates have shown how the failure increases as the components degrade and if the degradation rate increases. A comparison of the *Cdf*s built with different degradation rates has shown the usefulness of the methodology.

Chapter 9

Conclusions and Future Work

The main contribution of this thesis has been to present a methodology that uses metamodels for the design optimization of dynamic systems when the design variables are uncertain (due to varying environmental or manufacturing conditions) and to plot the *Cdf* (cumulative distribution function of time to failure) when degradation of the design variables is assumed. Metamodels have been found to provide simple alternatives to otherwise complex mechanistic models and have been easily used with probability evaluation methods such as MCS (Monte Carlo Simulation) or FORM (First-Order Reliability Method). Metamodels have also been used with the set-theory method to build the *Cdf* when degradation of the design variables is assumed. A few examples have been used to illustrate the methodology and to compare the speed and accuracy of three popular metamodels, the RM (Regression Model), Kriging and the RBF (Radial Basis Function). A general metamodel form has been developed that has been found to be applicable to model static systems, dynamic systems and dynamic degrading systems. Examples have been presented to illustrate flexibility of the general metamodel form.

Errors have been shown to come from six sources; the choice of cycle-time increment, the choice of the number of singular values, s, using FORM or the MCS to estimate probability, the choice of the service-time increment and using two contiguous service time increments in the set-theory method to estimate the *Cdf*. An example has been presented to show how the cycle-time increment and the number of columns of $\mathbf{S}$ are chosen. Reducing the error from the fit of the metamodel has been found to be difficult. The only way is to perhaps choose another metamodel or stick with the mechanistic model. In the set-theory method, errors come from using only two contiguous time steps and making the time step too large. Previous research has shown that if the value of Δt is small enough, using two events is sufficient. In order to determine the appropriate size of Δt, a correlation coefficient from two position of a particular limit-state function at contiguous time t_l and t_{l+1} is computed. Ideally, this value should be greater than 0.995.

Examples have been presented to show the probabilistic-design of static systems using metamodels, probabilistic-design of dynamic systems using metamodels and dynamic degradation modelling using the set-theory method and metamodels. For the simple static examples, the Regression Model has been found to be both fast and accurate. The RBF has been found to be the slowest metamodel when FORM has been used to estimate probability of failure. The speed of the RBF has been found to increase when MCS has been used. For probabilistic design optimization of the dynamic systems, the metamodels have provided a simple analytical function for the time-invariant performance measure. The metamodel approximation has been found to be accurate and to provide an optimal design much faster than would normally be found using the mechanistic model. Kriging has been found to be accurate and converges in the same number of iterations as the RM. However, Kriging is much slower than the RM.

Finally, metamodels have been used with the set-theory to plot the approximate *Cdf* when degradation of the design variables has been assumed. The degradation path and marginal distribution models have been used to model degradation of the Resistance of a position-control servo. A comparison of the FORM-based metamodel and MCS-based metamodel results with the MCS-based mechanistic model results have shown that metamodels combined with the set-theory method and have achieved good accuracy and a better computation time than the mechanistic model. For probability evaluation, MCS with the metamodels has been found to be feasible due to its accuracy and efficiency.

Overall, it has been found that metamodels can provide a suitable alternative to the mechanistic model for both design optimization or degradation modelling.

9.1 Contributions and Goals

This thesis has developed a methodology (see Figure 1-3) that uses metamodels for probability-based design optimization and for estimating the *Cdf*, using the set-theory method, when degradation of the design variables is assumed. The major contributions have been

- Developing a general metamodel that can be quickly built for static, dynamic or dynamic degrading systems.
- Combining SVD with metamodels to efficiently estimate a time-invariant performance measure based upon a dynamic response.
- Using the metamodel approximation to build a LSF to allow reliability-based design optimization using parameter design, tolerance or integration design methods.
- Combining metamodels with the set-theory method to estimate the incremental failure probability and to estimate the *Cdf* when degradation of the variables is assumed.
- Comparison of the Regression Model, Kriging and RBF, in terms of speed and accuracy, for probabilistic-based design optimization and estimating the *Cdf*.

9.2 Comments about the Methodology

The examples presented have assumed normal and independently distributed design variables. For the case where the variables are correlated and do not follow a normal distribution, a transformation method can be used to convert the variables to u-space when FORM is to be used to estimate the failure probability. A popular transformation method is the Rosenblatt transformation (Rosenblatt 1952); (Robinson 1998). The Rosenblatt transformation is illustrated in Appendix D along with an example.

Examples presented have also used a range wider than the design space to select the training design. However, there may be cases where this is not possible or not feasible. Other training design selection methods may include space-filling designs, such as the Latin Hypercube Design, that allow the design space to be thoroughly searched. For deterministic design variables, the tolerance can be set to zero and the variable is set to the mean value. Optimization then becomes a parameter design problem.

9.3 Future Research and Extensions

Since the *Cdf* can be built using a metamodel, one of the major future directions of this work is to use reliability-based design to search for a design to have a desired product lifetime. Therefore, the future of this work is in dependability design where the distribution parameters of the design variables, in degrading systems, are selected that reduces cost and minimizes the failure probability. Some work has been done in this area and applied to static systems but none with dynamic degrading systems.

This thesis has found that the RM is very fast compared with Kriging or the RBF. Although the efficiency of the model is desirable, the accuracy is sometimes not as good as Kriging. A future direction may also search for ways to improve the accuracy of the RM or using the RM to quickly locate a good design and then using Kriging to plot an accurate *Cdf*. Investigating space-filling designs, instead of the factorial type design that has been used, with metamodels can also be a future direction.

Appendix A
Kriging Model Background

Let the set of M known inputs (training points) to the computer model be $\mathbf{X}_d = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M] \subset \Omega$ where Ω is the set of all possible inputs to the model that result in an output. For an individual column in **D**, the resulting outputs are $\mathbf{d} = [\mathbf{d}(\mathbf{x}_1) \quad \ldots \quad \mathbf{d}(\mathbf{x}_M)]$. Given these sampled outputs or observations of the computer model, consider a linear predictor of the output at any point $\mathbf{v} \in \Omega$.

$$\hat{d}(\mathbf{v}) = \tilde{\mathbf{w}}^T(\mathbf{v})\mathbf{d} \tag{A.1}$$

Kriging treats $\hat{d}(\mathbf{v})$ as a random function and finds the best linear unbiased prediction (BLUP), $\tilde{\mathbf{w}}^T(\mathbf{v})\mathbf{d}$, that minimizes the mean square error (MSE) of the prediction (A.2) subject to the unbiasedness constraint (A.3) (Sacks, et al. 1989).

$$\mathrm{MSE}\left[\hat{d}(\mathbf{v})\right] = \mathrm{E}\left[\tilde{\mathbf{w}}^T(\mathbf{v})\mathbf{d} - d(\mathbf{v})\right]^2 \tag{A.2}$$

$$E\left[\tilde{\mathbf{w}}^T(\mathbf{v})\mathbf{d}\right] = E\left[d(\mathbf{v})\right] \tag{A.3}$$

The MSE of (A.2) becomes (A.4)

$$\sigma_{KG}^2\left[1 + \tilde{\mathbf{w}}^T(\mathbf{v})\mathbf{\Gamma}\tilde{\mathbf{w}}(\mathbf{v}) - 2\tilde{\mathbf{w}}^T(\mathbf{v})\tilde{\mathbf{r}}(\mathbf{v})\right] \tag{A.4}$$

From equation (A.4), **Γ** is a correlation matrix composed of the spatial correlation between each possible combination of the training sets. Therefore, the M training sets produce the M x M matrix of correlation pairs

$$\mathbf{\Gamma} = \begin{bmatrix} \gamma(\mathbf{x}_1, \mathbf{x}_1) & \gamma(\mathbf{x}_1, \mathbf{x}_2) & \ldots & \gamma(\mathbf{x}_1, \mathbf{x}_M) \\ \gamma(\mathbf{x}_2, \mathbf{x}_1) & \gamma(\mathbf{x}_2, \mathbf{x}_2) & \ldots & \gamma(\mathbf{x}_2, \mathbf{x}_M) \\ \vdots & \vdots & \ddots & \vdots \\ \gamma(\mathbf{x}_M, \mathbf{x}_1) & \gamma(\mathbf{x}_M, \mathbf{x}_2) & \ldots & \gamma(\mathbf{x}_M, \mathbf{x}_M) \end{bmatrix}_{M \times M} \tag{A.5}$$

The matrix, **Γ**, is symmetric since $\gamma(\mathbf{x}_1, \mathbf{x}_2) = \gamma(\mathbf{x}_2, \mathbf{x}_1)$ and the diagonal consists of all ones. The elements of the vector, $\tilde{\mathbf{r}}$, represents the correlation between a vector of unknown points, $\mathbf{v} \in \Omega$, and the M known training sets points, $\mathbf{x}_{\mathrm{j}}$

$$\tilde{\mathbf{r}}^T(\mathbf{v}) = [\gamma(\mathbf{v}, \mathbf{x}_1) \quad \dots \quad \gamma(\mathbf{v}, \mathbf{x}_M)]_{1\times M}^T \tag{A.6}$$

The unbiasedness constraint (A.3) becomes (A.7) where $\mathbf{v}_R = \lfloor 1 \quad v_1 \quad \dots \quad v_\eta \rfloor$.

$$\mathbf{X}^T \tilde{\mathbf{w}}(\mathbf{v}) = \mathbf{v}_R \tag{A.7}$$

Introducing Lagrange multipliers L($\mathbf{v}$)for the constrained minimization of the MSE, the coefficient $\tilde{\mathbf{w}}(\mathbf{v})$of the BLUP must satisfy

$$\begin{pmatrix} 0 & \mathbf{X}_d^T \\ \mathbf{X}_d & \mathbf{\Gamma} \end{pmatrix} \begin{pmatrix} L(\mathbf{v}) \\ \mathbf{w}(\mathbf{v}) \end{pmatrix} = \begin{pmatrix} \mathbf{v}_R \\ \tilde{\mathbf{r}}(\mathbf{v}) \end{pmatrix} \tag{A.8}$$

Then, by inverting the partitioned matrix, the BLUP can be written as equation (A.9) which is the standard form for the Universal Kriging model.

$$\hat{d}(\mathbf{v}) = \beta_0 + \tilde{\boldsymbol{\beta}}^T \mathbf{v} + \tilde{\mathbf{r}}^T(\mathbf{v}) \mathbf{\Gamma}^{-1} (\mathbf{d} - \mathbf{X}_b \boldsymbol{\beta}) \tag{A.9}$$

Appendix B

Computation of the Bivariate Normal Integral

To evaluate the normal integral, we define L($h,k;\rho$) in such a way that the following quality holds (Drezner and Wesolowsky 1990)

$$\Phi(h,k,\rho)=L(h,k,\rho)+\Phi(h)+\Phi(k)-1 \tag{B.1}$$

Here $L(h,k,\rho)$ is

$$L(h,k,\rho)=\int_h^\infty\int_k^\infty \frac{1}{2\pi\sqrt{1-\rho^2}}\exp\left(-\frac{1}{2}\cdot\frac{x^2-2\rho xy+y^2}{1-\rho^2}\right)dxdy \tag{B.2}$$

L(h,k,ρ) is selected over the CDF Φ($h,k;\rho$) because it is more commonly calculated. Equation (B.2) is reduce to the one-dimensional integral

$$L(h,k;\rho)=\frac{1}{2\pi}\int_{\cos^{-1}\rho}^{\pi}\exp\left\{-\left[h^2+k^2-2hk\cos z\right]/\left[2\sin^2 z\right]\right\}dz \tag{B.3}$$

Differentiating (B.2) by ρ, the following relationship is obtained

$$\frac{\partial L}{\partial\rho}=\frac{1}{2\pi\sqrt{1-\rho^2}}\exp\left\{-\left[h^2-2\rho hk+k^2\right]/\left[2\left(1-\rho^2\right)\right]\right\} \tag{B.4}$$

By equation (B.1), $\frac{\partial\Phi}{\partial\rho}=\frac{\partial L}{\partial\rho}$. Since L($h,k;\rho$) = Φ(-h)Φ(-k)

$$L(h,k;\rho)=\int_0^\rho\frac{\partial L}{\partial\rho}d\rho+\Phi(-h)\Phi(-k) \tag{B.5}$$

From equations (B.3) and (B.4)

The Gaussian quadrature formulae based on Legendre polynomials with K = 5 points can be used to evaluate (). The subroutine used to evaluate the probability with K = 5 points is given. It is named *bv1* and the inputs are h, k and ρ denoted by h1, hk and r respectively. The subroutine *phi* calculates normal probabilities. MATLAB® code for subroutine bv1 and phi are given (Seshadri 2002).

% Calculation of bivariate normal probabilities

```
function l2 = bv1(h1,hk,r)
```

```
% Points of function evaluations - Legendre Polynomials
x = [0.04691008,0.23076534,0.50,0.76923466,0.95308992];

% The weights
w = [0.018854042,0.038088059,0.0452707394,0.038088059,0.018854042];
l2 = 0;
h2 = hk;
h12 = (h1*h1 + h2*h2)/2;
h3 = h1*h2;
r2 = 1-r*r;
if(r2==0)
    if(r>0)
        l2 = phi(max(h1,h2));
    else
        h2 = -h2;
        l2 = max(0,phi(h1)-phi(h2));
    end
else
    for i = 1:5
        r1 = r*x(i);
        rr2 = 1-r1*r1;
        l2 = l2+w(i)*exp((r1*h3-h12)/rr2)/sqrt(rr2);
    end
    l2 = phi(h1)*phi(h2)+r*l2;
    if(l2>0) l2 = 0;
    end
end

% Subroutine for Calculation of normal probabilities
function prob = phi(z)

a = [-0.72657601,0.71070688,-0.142248368,0.127414796];
x = 1/(1+0.23164189*abs(z));
g = 0.53070271;
for i = 1:4
    g = g*x + a(i);
end
prob = g*x*exp(-z*z/2);
if(z<0) prob = 1-prob;
end
```

Appendix C

Tables and Figures for Chapter 8 Case Study

Table C.1: The energy estimates when the response is broken into different time steps.

Time	**Energy**			
	$\Delta\tau = 0.005$	**$\Delta\tau = 0.002$**	**$\Delta\tau = 0.001$**	**$\Delta\tau = 0.0005$**
1	0.000193966	0.000196141	0.000196242	0.000196234
2	0.000193966	0.000196141	0.000196242	0.000196234
3	0.000193966	0.000196141	0.000196242	0.000196234
4	0.000193737	0.000195865	0.000195963	0.000195953
5	0.000193737	0.000195865	0.000195963	0.000195953
6	0.000193737	0.000195865	0.000195963	0.000195953
7	0.000193507	0.00019559	0.000195686	0.000195675
8	0.000193507	0.00019559	0.000195686	0.000195675
9	0.000193507	0.00019559	0.000195686	0.000195675
10	0.00019415	0.000196415	0.000196527	0.000196522
11	0.00019415	0.000196415	0.000196527	0.000196522
12	0.00019415	0.000196415	0.000196527	0.000196522
13	0.000193936	0.000196149	0.000196256	0.000196248
14	0.000193936	0.000196149	0.000196256	0.000196248
15	0.000193936	0.000196149	0.000196256	0.000196248
16	0.000193715	0.000195878	0.000195982	0.000195973
17	0.000193715	0.000195878	0.000195982	0.000195973
18	0.000193715	0.000195878	0.000195982	0.000195973
19	0.00019431	0.000196668	0.00019679	0.000196789
20	0.00019431	0.000196668	0.00019679	0.000196789
21	0.00019431	0.000196668	0.00019679	0.000196789
22	0.000194115	0.000196416	0.000196533	0.000196529
23	0.000194115	0.000196416	0.000196533	0.000196529
24	0.000194115	0.000196416	0.000196533	0.000196529
25	0.000193908	0.000196156	0.000196268	0.000196261
26	0.000193908	0.000196156	0.000196268	0.000196261
27	0.000193908	0.000196156	0.000196268	0.000196261

Table C.2: The energy estimate for various columns of **S**.

M	**Energy**			
	Original	**3 Columns**	**4 Columns**	**5 Columns**
1	1.958585E-04	1.959082E-04	1.958589E-04	1.958574E-04
2	1.955712E-04	1.956125E-04	1.955790E-04	1.955698E-04
3	1.952945E-04	1.953175E-04	1.953011E-04	1.952939E-04
4	1.950323E-04	1.950145E-04	1.950160E-04	1.950321E-04
5	1.947875E-04	1.946966E-04	1.947166E-04	1.947851E-04
6	1.961554E-04	1.961869E-04	1.961477E-04	1.961558E-04
7	1.958686E-04	1.958979E-04	1.958708E-04	1.958678E-04
8	1.955869E-04	1.956195E-04	1.956064E-04	1.955865E-04
9	1.953150E-04	1.953422E-04	1.953444E-04	1.953165E-04
10	1.950564E-04	1.950579E-04	1.950763E-04	1.950602E-04
11	1.964395E-04	1.964532E-04	1.964288E-04	1.964414E-04
12	1.961593E-04	1.961594E-04	1.961432E-04	1.961594E-04
13	1.958784E-04	1.958866E-04	1.958808E-04	1.958774E-04
14	1.956022E-04	1.956242E-04	1.956305E-04	1.956021E-04
15	1.953349E-04	1.953631E-04	1.953830E-04	1.953378E-04
16	1.967054E-04	1.967192E-04	1.967143E-04	1.967071E-04
17	1.964375E-04	1.964102E-04	1.964091E-04	1.964392E-04
18	1.961631E-04	1.961326E-04	1.961380E-04	1.961626E-04
19	1.958880E-04	1.958748E-04	1.958890E-04	1.958864E-04
20	1.956170E-04	1.956269E-04	1.956517E-04	1.956169E-04
21	1.969486E-04	1.969949E-04	1.970143E-04	1.969451E-04
22	1.966983E-04	1.966613E-04	1.966799E-04	1.967006E-04
23	1.964357E-04	1.963692E-04	1.963902E-04	1.964368E-04
24	1.961669E-04	1.961064E-04	1.961324E-04	1.961654E-04
25	1.958973E-04	1.958624E-04	1.958957E-04	1.958948E-04

Table C.3: The incremental and instantaneous failure probabilities, using the degradation path model with $\mu(k_R) = 0.004$ using FORM.

Time	**Mechanistic**	**RM**		**Kriging**		**RBF**	
	MCS	**FORM**	**Pr(B$_l$)**	**FORM**	**Pr(B$_l$)**	**FORM**	**Pr(B$_l$)**
0	0.0039	0.0036		0.004		0.003913	
0.2	0.0044	0.004196	0.000596	0.004604	0.000604	0.004945	0.001032
0.4	0.0051	0.004935	0.000739	0.005405	0.000801	0.005318	0.000372
0.6	0.0064	0.005787	0.000853	0.006325	0.000919	0.006231	0.000914
0.8	0.0075	0.006768	0.000981	0.007389	0.001064	0.007869	0.001638
1	0.0086	0.007893	0.001125	0.008599	0.00121	0.008474	0.000605
1.2	0.0095	0.009179	0.001286	0.009984	0.001385	0.009817	0.001342
1.4	0.0113	0.01064	0.001465	0.011545	0.00156	0.01135	0.001532
1.6	0.0128	0.01231	0.001665	0.013329	0.001784	0.01311	0.001765
1.8	0.0144	0.01419	0.001885	0.015339	0.00201	0.01512	0.002003
2	0.0171	0.01632	0.002128	0.017609	0.00227	0.01732	0.002201
2.2	0.0198	0.01872	0.002395	0.02014	0.002531	0.01986	0.002546
2.4	0.0222	0.02141	0.002688	0.022967	0.002827	0.02264	0.002776
2.6	0.0255	0.02441	0.003006	0.026164	0.003197	0.02580	0.003163
2.8	0.0298	0.02776	0.003352	0.029686	0.003522	0.02930	0.003496
3	0.0343	0.03149	0.003726	0.033596	0.003909	0.03321	0.003913
3.2	0.0391	0.03562	0.004129	0.037908	0.004312	0.03742	0.004212
3.4	0.0439	0.04018	0.004561	0.042671	0.004763	0.04219	0.004765
3.6	0.0495	0.045203	0.005023	0.047889	0.005217	0.04732	0.005129
3.8	0.0563	0.050717	0.005514	0.053593	0.005705	0.05292	0.005598
4	0.0621	0.056752	0.006035	0.059842	0.006249	0.05912	0.006205
4.2	0.0685	0.063336	0.006584	0.066619	0.006777	0.06579	0.006668
4.4	0.0772	0.070498	0.007162	0.074017	0.007398	0.07331	0.007524
4.6	0.0854	0.078263	0.007766	0.081988	0.007971	0.08111	0.007795
4.8	0.095	0.086658	0.008394	0.090609	0.008621	0.08972	0.008614
5	0.1041	0.095703	0.009045	0.09979	0.009181	0.09895	0.009226

Table C.4: The Pr($\mathbf{B}_l$) estimates when MCS is used to estimate failure probability.

	Mechanistic	**RM**		**Kriging**		**RBF**	
Time	**MCS**	**Pr(g2<0)**	Pr($\mathbf{B}_l$)	**Pr(g2<0)**	Pr($\mathbf{B}_l$)	**Pr(g2<0)**	Pr($\mathbf{B}_l$)
0	0.0039	0.00385		0.0038		0.00399	
0.2	0.0044	0.00446	0.00061	0.00436	0.00056	0.00472	0.00073
0.4	0.0051	0.00527	0.00081	0.00511	0.00075	0.0056	0.00088
0.6	0.0064	0.00601	0.00074	0.00613	0.00102	0.0065	0.0009
0.8	0.0075	0.00683	0.00082	0.00723	0.0011	0.00769	0.00119
1	0.0086	0.00789	0.00106	0.00846	0.00123	0.00887	0.00118
1.2	0.0095	0.00917	0.00128	0.00992	0.00146	0.01002	0.00115
1.4	0.0113	0.01061	0.00144	0.01126	0.00134	0.01153	0.00151
1.6	0.0128	0.01227	0.00166	0.0131	0.00184	0.01329	0.00176
1.8	0.0144	0.01413	0.00186	0.01516	0.00206	0.01522	0.00193
2	0.0171	0.01627	0.00214	0.01729	0.00213	0.01733	0.00211
2.2	0.0198	0.0188	0.00253	0.02011	0.00282	0.01968	0.00235
2.4	0.0222	0.02144	0.00264	0.02294	0.00283	0.02252	0.00284
2.6	0.0255	0.02443	0.00299	0.02587	0.00293	0.0254	0.00288
2.8	0.0298	0.02779	0.00336	0.02902	0.00315	0.02893	0.00353
3	0.0343	0.03137	0.00358	0.03297	0.00395	0.0324	0.00347
3.2	0.0391	0.03541	0.00404	0.03764	0.00467	0.03693	0.00453
3.4	0.0439	0.04008	0.00467	0.04221	0.00457	0.0415	0.00457
3.6	0.0495	0.04538	0.0053	0.04731	0.0051	0.04675	0.00525
3.8	0.0563	0.05082	0.00544	0.05391	0.0066	0.05237	0.00562
4	0.0621	0.05677	0.00595	0.06033	0.00642	0.05901	0.00664
4.2	0.0685	0.06358	0.00681	0.06716	0.00683	0.06577	0.00676
4.4	0.0772	0.07079	0.00721	0.07464	0.00748	0.07321	0.00744
4.6	0.0854	0.07863	0.00784	0.0832	0.00856	0.08172	0.00851
4.8	0.095	0.08718	0.00855	0.09078	0.00758	0.09028	0.00856
5	0.1041	0.09607	0.00889	0.10053	0.00975	0.1002	0.00992

Table C.5: The training sets, for each variable, for each range using λ = 3.

Range	Training Design	
$\mu \pm 3\sigma$	$\mathbf{x}_1 = \begin{bmatrix} 5.044 \\ 5.200 \\ 5.356 \end{bmatrix}$ $\mathbf{x}_2 = \begin{bmatrix} 3.88 \\ 4.00 \\ 4.12 \end{bmatrix}$	$\mathbf{x}_3 = \begin{bmatrix} 0.00388 \\ 0.00400 \\ 0.00412 \end{bmatrix}$
$\mu \pm 4.5\sigma$	$\mathbf{x}_1 = \begin{bmatrix} 4.966 \\ 5.200 \\ 5.434 \end{bmatrix}$ $\mathbf{x}_2 = \begin{bmatrix} 3.82 \\ 4.00 \\ 4.18 \end{bmatrix}$	$\mathbf{x}_3 = \begin{bmatrix} 0.00382 \\ 0.00400 \\ 0.00481 \end{bmatrix}$
$\mu \pm 6\sigma$	$\mathbf{x}_1 = \begin{bmatrix} 4.888 \\ 5.200 \\ 5.512 \end{bmatrix}$ $\mathbf{x}_2 = \begin{bmatrix} 3.76 \\ 4.00 \\ 4.24 \end{bmatrix}$	$\mathbf{x}_3 = \begin{bmatrix} 0.00376 \\ 0.00400 \\ 0.00424 \end{bmatrix}$
$\mu \pm 10\sigma$	$\mathbf{x}_1 = \begin{bmatrix} 4.68 \\ 5.20 \\ 5.72 \end{bmatrix}$ $\mathbf{x}_2 = \begin{bmatrix} 3.60 \\ 4.00 \\ 4.40 \end{bmatrix}$	$\mathbf{x}_3 = \begin{bmatrix} 0.0036 \\ 0.0030 \\ 0.0044 \end{bmatrix}$

Table C.6: The training sets, for each variable, for each range using λ = 5.

Range	Training Design	
μ±3σ with λ = 5	$\mathbf{x}_1 = \begin{bmatrix} 5.044 \\ 5.122 \\ 5.200 \\ 5.278 \\ 5.356 \end{bmatrix}$ $\mathbf{x}_2 = \begin{bmatrix} 3.88 \\ 3.93 \\ 4.00 \\ 4.06 \\ 4.12 \end{bmatrix}$	$\mathbf{x}_3 = \begin{bmatrix} 0.00388 \\ 0.00393 \\ 0.004 \\ 0.00407 \\ 0.00412 \end{bmatrix}$
μ±4.5σ with λ = 5	$\mathbf{x}_1 = \begin{bmatrix} 4.966 \\ 5.083 \\ 5.200 \\ 5.317 \\ 5.434 \end{bmatrix}$ $\mathbf{x}_2 = \begin{bmatrix} 3.82 \\ 3.91 \\ 4.00 \\ 4.09 \\ 4.18 \end{bmatrix}$	$\mathbf{x}_3 = \begin{bmatrix} 0.00382 \\ 0.00391 \\ 0.00400 \\ 0.00409 \\ 0.00481 \end{bmatrix}$
μ±6σ with λ = 5	$\mathbf{x}_1 = \begin{bmatrix} 4.888 \\ 5.044 \\ 5.200 \\ 5.356 \\ 5.512 \end{bmatrix}$ $\mathbf{x}_2 = \begin{bmatrix} 3.76 \\ 3.88 \\ 4.00 \\ 4.12 \\ 4.24 \end{bmatrix}$	$\mathbf{x}_3 = \begin{bmatrix} 0.00376 \\ 0.00388 \\ 0.00400 \\ 0.00412 \\ 0.00424 \end{bmatrix}$
μ±10σ with λ = 5	$\mathbf{x}_1 = \begin{bmatrix} 4.68 \\ 4.94 \\ 5.200 \\ 5.46 \\ 5.72 \end{bmatrix}$ $\mathbf{x}_2 = \begin{bmatrix} 3.60 \\ 3.80 \\ 4.00 \\ 4.20 \\ 4.40 \end{bmatrix}$	$\mathbf{x}_3 = \begin{bmatrix} 0.00360 \\ 0.00380 \\ 0.00400 \\ 0.00420 \\ 0.00440 \end{bmatrix}$

Appendix D
The Rosenblatt Transformation

For arbitrary *pdf*`s of design variables, the Rosenblatt Transformation transforms the variable`s *pdf* from **v**-space to **u**-space resulting in a one-to-one mapping of the arbitrary *pdf* to the standard normal distribution. Suppose the $\mathbf{v} = \lfloor v_1, v_2, \ldots, v_\eta \rfloor$ is a vector of statistically dependent random variables. Let the joint probability and cumulative density function of **v** be defined as $f(\mathbf{v})$ and $F_\mathbf{V}(\mathbf{v})$ respectively. Define the marginal density functions and cumulative distribution functions as

$$f_i(v_1, v_2, \ldots, v_i) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} f_i(v_1, v_2, \ldots, v_i, s_{i+1}, \ldots, s_\eta) ds_{i+1} \ldots ds_\eta \tag{D.1}$$

$$H_i(v_i | v_1, v_2, \ldots, v_{i-1}) = \frac{1}{k_i} \int_{-\infty}^{v_i} f_i(v_1, v_2, \ldots, v_{i-1}, s_i) ds_i \tag{D.2}$$

Where $k_i = \int_{-\infty}^{\infty} f_i(v_1, v_2, \ldots, v_{i-1}, s_i) ds_i$ and $H_1(v_1) = F_1(v_1)$. The new set of independent standardized Gaussian random variables are then given by

$$\mathbf{u} = (u_1, u_2, \ldots, u_\eta) = \{\Phi^{-1}[H_1(v_1)], \Phi^{-1}[H_2(v_2|v_1)], \ldots, \Phi^{-1}\lfloor H_\eta(v_\eta | v_1, \ldots, v_{\eta-1})\rfloor\} \tag{D.3}$$

Example

From Robinson 1998, let V_1 be a Gaussian distributed random variable with mean μ_1 and standard deviation σ_1. Let V_2 be lognormally distributed random variable with *pdf*

$$f(v_2) = \frac{1}{v_2 \xi \sqrt{2\pi}} \exp\left[-\frac{1}{2} \left\{ \frac{\ln(v_2) - \lambda_{LN}}{\xi} \right\} \right] \tag{D.4}$$

Where, provided with the median, $\tilde{v}_2$, and coefficient of variation, κ_2

$$\begin{aligned}
\lambda_{LN} &= \ln(\tilde{v}_2) \\
\xi &= \sqrt{\ln(1+\kappa_2^2)} \\
\mu_2 &= \tilde{v}_2\sqrt{1+\kappa_2^2} \\
\sigma_2 &= \kappa_2\mu_2
\end{aligned}$$

Assume V_1, V_2 are correlated such that $\rho_{1,2} = 0.6$. Applying the Rosenblatt Transformation (D.3)

$$\begin{aligned}
\Phi(u_1) &= H_1(v_1) \\
\Phi(u_2) &= H_2(v_2|v_1)
\end{aligned} \tag{D.5}$$

Where

$$H_2(v_2|v_1) = \Phi\left[\frac{\ln(v_2) - \lambda_{LN} - \eta_H\left(\xi/\sigma_1\right)(v_1 - \mu_1)}{\xi\sqrt{1-\eta_H^2}}\right] \tag{D.6}$$

and

$$\eta_H = \frac{\rho}{\xi}\left(\frac{\sigma_2}{\mu_2}\right)$$

The new vector of reduced space random variables is therefore composed of

$$u_1 = \frac{v_1 - \mu_1}{\sigma_1} \tag{D.7}$$

$$u_2 = \frac{\ln(v_2) - \lambda_{LN} - \eta_H\left(\dfrac{\xi}{\sigma_1}\right)(v_1 - \mu_1)}{\xi\sqrt{1-\eta_H^2}} \tag{D.8}$$

Bibliography

Ahmed M. Y. M. and Qin N., (2009) “Comparison of Response Surface and Kriging Surrogates in Aerodynamic Design Optimization of Hypersonic Spiked Blunt Bodies”, 13th International Conference on Aerospace Sciences and Aviation Technology, May 26th – 28th , Military Technical College, Kobry Elkobbah, Cairo, Egypt.

Allen M., Raulli M., Maute K. and Frangopol D. M., (2004), “Reliability-Based Analysis and Design Optimization of Electrostatically Actuated MEMS”, *Computers and Structures*, Vol. 82, pp. 1007 – 1020.

Andina D. and Pham D. T., (2007), *Computational Intelligence for Engineering and Manufacturing*, Springer, The Netherlands.

Andrieu-Renaud C., Sudret, B. and Lemaire, M., (2004), “The PHI2 Method: A Way to Compute Time-Variant Probability”, *Reliability Engineering and System Safety*, Vol. 84, No. 1, pp. 75 – 86.

Bagchi T. and Templeton J. (1994), “Robust design engineering using variance transfer”, *Proceedings of ISSAT 2nd International Conference on Reliability and Quality in Design*, Seattle, March, pp. 203 – 207.

Barton, R. R. (1998), “Simulation Metamodels”, *Proceedings of the Winter 1998 Simulation Conference*, Washington, D. C., 13th – 16th December, pp. 167 – 174.

Barton, R. R., and Meckesheimer M.. (2006) “Metamodel-Based Simulation Optimization”, In *Handbooks in Operations Research and Management Science*, by S. G. Henderson and B. L. Nelson, 535-574. Elsevier.

Bogdanoff J. L. and Kozin F., (1985), *Probabilistic Models of Cumulative Damage*, John Wiley and Sons, New York.

Bucher, C. G., and Bourgund, U. (1990) “A Fast and Efficient Response Surface Approach for Structural Reliability Problems.” *Structural Safety* Vol. 7, no. 1, pp. 57 – 66.

Chiralaksanakul, A., and Mahadevan, S.. (2005) “First-Order Approximation Methods in Reliability-Based Design Optimization.” *Journal of Mechanical Design* Vol. 127, no. 5, pp. 851 – 857.

Clarke, S. M., Griebsch J. H. and Simpson T. W., (2005), “Analysis of Support Vector Regression for Approximation of Complex Engineering Analyses.” *Journal of Mechanical Design*, vol. 127 pp. 1077 – 1087.

Cochin, I., (1980), *Analysis and Design of Dynamic Systems*, Harper and Row, New York.

Deng, J. (2006) “Structural Reliability Analysis for Implicit Performance Function using Radial Basis Function Network.” ,*International Journal of Solids and Structures*, vol. 43, no. 11 – 12, pp. 3255 – 3291.

Der Kiureghain, A., and Dakessian T. (1998) “Multiple Design Points in First and Second-Order Reliability.” *Structural Safety* Vol. 20, no. 1, pp. 37 – 49.

Drezner Zvi and Wesolowsky G. O. (1990), “On the Computation of the Bivariate Normal Integral”, *Journal of Statistical Computation and Simulation*, Vol. 34 (1/2), pp. 101 – 107.

Eberhart R. C. and Shi Y., (2001), “Tracking and optimizing dynamic systems with particle swarms”, *Proceedings of the 2001 Congress on Evolutionary Computing*, Seoul, South Korea, 27th May – 30th May, pp. 94 – 100.

Esfandiari R. S. and Lu B., (2010), *Modeling and Analysis of Dynamic System*, CRC Press, USA.

Finkelstein M. and Cha J. H., (2010), “On Some Shock Models of Degradation”, *Advances in Degradation Modeling*, Part 2, pp. 117 – 124.

Gayton N. Bourinet J. M. And Lemaire M. (2003), “CQ2RS: A New Statistical Approach to the Response Surface Method for Reliability Analysis”, *Structural Safety*, Vol. 25, pp. 99 – 121.

Grandhi R. V. and Wang L., (1998) “Reliability-Based Structural Optimization using Improved Two-Point Adaptive Nonlinear Programming”, *Finite Elements in Analysis and Design*, Vol. 29, pp. 35 – 48.

Hassanpour, H., Mesbah M., and Boashash B.. (2004) “Time-Frequency Feature Extraction of Newborn EEG Seizure using SVD-Based Techniques.” *EURASIP Journal on Applied Signal Processing,* vol. 16, pp. 2544 – 2554.

Huang W. and Dietrich D. L. (2005), “An Alternative Degradation Reliability Modeling Approach using Maximum Likelihood Estimation”, *IEEE Transactions on Reliability*, Vol. 52, no. 2, pp. 310 – 317.

Hussain, M. K., Barton R. R. and Joshi S. B., (2002), “Metamodeling: Radial Basis Functions versus Polynomials.” *European Journal of Operational Research* Vol. 138, No. 1, pp. 142 – 154.

Jensen H. A. and Catalan M. A., (2007), “On the Effects of Non-Linear Elements in the Reliability-Based Optimal Design of Stochastical Dynamical Systems” *International Journal of Non-Linear Mechanics*, Vol. 42, pp. 802 – 816.

Jin, R., Chen W., Simpson T. W.. (2001) “Comparative Studies of Metamodelling Techniques under Multiple Modelling Criteria”, *Structural Multidisciplinary Optimization* Vol. 23, No. 1, pp. 1-13.

Ju B. H. and Lee B. C., (2008), “Reliability-Based Design Optimization using a Moment Method and a Kriging Metamodel”, *Engineering Optimization*, Vol. 40 (5), pp. 421 – 438.

Karray, F. O., and De Silver, C. (2004) *Soft Computing and Intelligent Systems Design*. Pearson.

Kaymaz I. (2005), “Application of Kriging Method to Structural Reliability Problems.” *Structural Safety* Vol. 27, pp. 133 – 151.

Kleijnen J. P. C. and Sargent R. G., (2000), “A Methodology for Fitting and Validating Metamodels in Simulation”, *European Journal of Operational Research*, Vol. 120, pp. 14 – 29.

Kleijnen J.P.C. and Van Beers W.C.M., (2003), “Kriging for Interpolation in Random Simulation”, *Journal of the Operational Research Society*, Vol. 54, pp. 255-262.

Lee T. H., Jung J. J., Hong S., Kim H. W. and Choi J., (2007), "Prediction for Motion of Tracked Vehicle Traveling on Soft Soil using Kriging Metamodel", *International Journal of Offshore and Polar Engineering*, Vol. 15, No. 2, pp. 132 – 138.

Madsen, H. O., Krenk, S. and Lind N. C. (1986) *Methods of Structural Safety*. Prentice-Hall Inc., USA.

Madsen, H. O. and Tvedt L. (1990), "Methods for Time-Dependent Reliability and Sensitivity Analysis", *Journal of Engineering Mechanics*, Vol. 116, No. 10, pp. 2118 – 2135.

Martin, J. D., and Simpson, T. W. (2003), "A Study on the Use of Kriging Models to Approximate Deterministic Computer Models." *ASME 2003 Design Engineering Technical Conference and Computers and Information in Engineering Conference*. Chicago: ASME, pp. 567 – 576.

Meckesheimer, M., Booker A. J., Barton R. R., and Simpson T. W., (2002), "Computationally Inexpensive Metamodel Assessment Strategies." *AIAA Journal* Vol. 40, No. 10, pp. 2053 – 2060.

Melchers, R. E. (1987), *Structural Reliability: Analysis and Prediction.* John Wiley and Sons.

Montgomery, D. C. (2005), *Design and Analysis of Experiments*. John Wiley and Sons Inc, USA.

Mourelatos, Z. P., and Wehrwein D. (2005) "Modelling and Optimization of Vehicle Drivetrain Dynamic Performance Considering Uncertainty". *SAE Noise and Vibration Conference.* Traverse City.

Ogata, K., (2004), *System Dynamics*, Pearson Prentice Hall, USA.

Ogata, K., (1990), *Modern Control Engineering*, Prentice Hall, USA.

Papadrakakis M. and Lagaros N. D. (2002) "Reliability-Based Structural Optimization using Neural Networks and Monte Carlo Simulation", *Computer Methods in Applied Mechanics and Engineering*, Vol. 191(32), pp. 3491 – 3507.

Rao, S. S. (1992), *Reliability-Based Design*, McGraw-Hill Inc., USA.

Rijpkema, J.J.M, Etman, L.F.P. and Schoofs, A.J.G., (2001), “Use of Design Sensitivity Information in Response Surface and Kriging Metamodels”, *Optimization and Engineering*, vol. 2, pp. 469 – 484.

Robinson D. G., (1998), *A Survey of Probabilistic Methods Used in Reliability, Risk and Uncertainty Analysis: Analytical Techniques 1*, Sandia National Laboratories, USA.

Rosenblatt, M., (1952), “Remarks on a Multivariate Transformation”, *The Annals of Mathematical Statistics*, Vol. 23, pp. 470 – 472.

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

Sakata S., Ashida F. and Zako M., (2003), “Structural Optimization using Kriging Approximation”, *Computer Methods in Applied Mechanics and Engineering*, Vol. 192, pp. 923-939.

Savage G. J. and Carr S. M., (2001), “Interrelating Quality and Reliability in Engineering Systems”, *Quality Engineering*, Vol. 14(1), pp. 137 – 152.

Savage G. J. and Son Y. K., (2011), “The set-theory method for systems reliability of structures with degrading components”, Reliability Engineering and System Safety, Vol. 96, pp. 108 – 116.

Seshadri R. (2002), “Integrated Robust Design using Probability of Conformance metrics”, MASc Thesis presented to the University of Waterloo, Waterloo.

Seshadri R. and Savage G. J. (2002), “Integrated robust design using probability of conformance metrics”, *International Journal of Materials and Product Technology*, Vol. 17, nos. 5/6, pp. 319 – 337.

Shapiro, S. S. and Alan J. G. (1981), *Statistical Modeling Techniques*, New York: Marcel Dekker Inc.

Simpson T. W., Mauery T. M., Korte J. J. and Mistree F. (2001), “Kriging Models for Global Approximation in Simulation-Based Multidisciplinary Design Optimization”, *AIAA Journal*, Vol. 39 no. 12, pp. 2233-2241.

Simpson T. W., Peplinski J. D., Koch P. N. And Allen J. K. (2001), “Metamodels for Computer-Based Engineering Design: Survey and Recommendations”, *Engineering with Computers*, VOl. 17, pp. 129 -150.

Stewart M.G., Rosowsky D.V. (1998), “Time-dependent Reliability of Deterioration Reinforced Concrete Bridge Decks”, *Structural Safety*, Vol. 20, No. 1, pp. 91 – 109.

Son Y. K. (2006), “Robust Design Methodology for Multi-Response Systems with Degrading Components”, *PhD Thesis*, The University of Waterloo, Waterloo, ON, Canada.

Srivastava, A., Hacker K., Lewis K., and Simpson T. W. (2004), “A Method for using Legacy Data for Metamodel-Based Design of Large-Scale Systems” *Structural Multidisciplinary Optimization* Vol. 28, pp. 146 – 155.

Tu, J., Choi, K. K. and Park, Y. H., (1999), “A new study on reliability-based design optimization”, *ASME Journal of Mechanical Design*, Vol. 121 (4), pp. 557 – 564.

Van Gigch, J. P. (1991), *System Design Modeling and Metamodeling*. New York: Plenum Press.

Van den Bogaard, J. A., Shreeram J. and Brombacher A. C. A. (2003), “Method for Reliability Optimization through Degradation Analysis and Robust Design”, *Proceedings of the Annual Reliability and Maintainability Symposium*, Tampa, Fl. USA, IEEE, pp. 55 – 62.

Wall, M. E., Rechtsteiner, A. and Rocha L. M. (2007), “Singular Value Decomposition and Principal Component Analysis.” In *A Practical Approach to Microarray Data Analysis*, by D. P. Berrar, W. Dubitzky and M. Granzow, 91 – 109. USA: Springer.

Wang, G. G. and Shan, S. (2007), “Review of Metamodelling Techniques in Support of Engineering Design Optimization”, *Journal of Mechanical Design*, Vol. 129, no. 4, pp. 370 – 380.

Wang, G. G., and Simpson T. W. (2004), “Fuzzy Clustering Based Hierarchical Metamodeling for Design Space Reduction and Optimization.” *Engineering Optimization* Vol. 36, no. 3, pp. 313 – 335.

Wehrwein, D., and Mourelatos Z. P. (2008), "Reliability-Based Optimization of Vehicle Drivetrain Dynamic Performance." *International Journal of Product Development* Vol. 5 (1-2), pp.: 54 – 75.

Wu, Y. T., Millwater H. R. and Cruse T. A., (1990), "Advanced Probabilistic Structural Analysis Method for Implicit Performance Functions", *AIAA*, Vol. 28 (9), pp. 1663 – 1669.

Youn, B. D., and Choi K. K., (2004) "Selecting Probabilistic Approaches for Reliability-Based Design Optimization", *AIAA Journal*, Vol. 42 no. 1, pp. 124 – 131.

The list of publications based on this thesis are as follows:

Savage G. J., Son Y. K. and Seecharan T. S., "Computing the CDF for Degrading Dynamic Systems", Accepted for inclusion in 18th ISSAT Conference on Reliability and Quality in Design.

Savage G. J., Seecharan T. S. and Son Y. K., "Probability-Based Prediction of Degrading Dynamic Systems", *ASME Journal of Mechanical Design*, Under Review.

Seecharan T. S. and Savage G. J., (2012), "Metamodel-Based Parameter Design of Static Systems", *The West Indian Journal of Engineering*, To Appear in July 2012 Issue.

Seecharan T. S. and Savage G. J., (2012), "Metamodel-Based Probabilistic Design of Static Systems with an extension to Dynamic Systems", *International Journal of Reliability, Quality and Safety Engineering*, Vol. 18(4), pp. 305 – 326 (To Appear).

Seecharan T. S. and Savage G. J. "Probability-Based Design Optimization of Dynamic Systems", International Journal of Reliability, Quality and Safety Engineering, Accepted for Publication December 31st 2011.

Seecharan T. S. and Savage G. J., (2011), "Probabilistic Design Methodology of Static Systems using Metamodels", *Conference Proceedings, 17th ISSAT Conference on Reliability and Quality in Design*, August 4th – 6th, Vancouver, B. C., Canada.

Seecharan T. S. and Savage G. J., (2010), "Comparison of the Radial Basis Function and Kriging Model in the Design of Dynamic Systems using the Monte Carlo Method", *Conference*

Proceedings, 16th ISSAT Conference on Reliability and Quality in Design, August 5th – 7th, Washington, D. C., USA.

Seecharan T. S. and Savage G. J. (2010) “Performability-Based Design Optimization of Dynamic Systems”, *International Journal of Performability Engineering*, Vol. 6 (2), pp. 123 – 136.

Seecharan T. S. and Savage G. J., (2009), “Performability Design of Dynamic Systems”, *Conference Proceedings, 15th ISSAT Conference on Reliability and Quality in Design,* August 6th – 8th, San Francisco, California, USA.