

2008

Metamodeling for the quantitative assessment of conceptual designs in an immersive virtual reality environment

Christian John Noon
Iowa State University

Follow this and additional works at: <http://lib.dr.iastate.edu/etd>

 Part of the [Mechanical Engineering Commons](#)

Recommended Citation

Noon, Christian John, "Metamodeling for the quantitative assessment of conceptual designs in an immersive virtual reality environment" (2008). *Graduate Theses and Dissertations*. 10922.
<http://lib.dr.iastate.edu/etd/10922>

This Thesis is brought to you for free and open access by the Graduate College at Iowa State University Digital Repository. It has been accepted for inclusion in Graduate Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.

**Metamodeling for the quantitative assessment of conceptual designs in an
immersive virtual reality environment**

by

Christian John Noon

A thesis submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE

Co-majors: Mechanical Engineering; Human Computer Interaction

Program of Study Committee:
Eliot Winer, Major Professor
James Oliver
Tom Shih

Iowa State University

Ames, Iowa

2008

Copyright © Christian John Noon, 2008. All rights reserved.

TABLE OF CONTENTS

1	INTRODUCTION	1
1.1	The Design Process	1
1.2	The Reality of Conceptual Design	5
1.3	Motivation	9
1.4	Thesis Organization	11
2	LITERATURE REVIEW	13
2.1	Introduction to Metamodeling	13
2.2	Overview of Metamodeling Techniques	16
2.2.1	Polynomial Response Surfaces	16
2.2.2	Kriging Approximations	17
2.2.3	Radial Basis Function Neural Networks	19
2.3	Development of Metamodeling techniques	23
2.4	Research Issues	26
3	METHODOLOGY	28
3.1	Dataset Development	28
3.1.1	Wheel Loading Dataset	29
3.1.2	Stress Analysis Dataset	33
3.2	Metamodel Development	38
3.2.1	Polynomial Response Surface Construction	39
3.2.2	Kriging Approximation Construction	42
3.2.3	Radial Basis Function Neural Network Construction	45
3.3	Process of Constructing a Metamodel	48
3.4	The Advanced Systems Design Suite	53
3.4.1	System Architecture	53
3.4.2	Interface Interactions	55
3.4.3	Assessment Tools	56
3.4.3.1	<i>Center of Gravity and Tipping Angle</i>	56
3.4.3.2	<i>Virtual Measuring and Wheel Loading</i>	58
4	RESULTS	60
4.1	Overall Summary Statistics	62
4.2	Sample Size Analysis	64
4.2.1	PRS Results - Sample Size - Dataset 1	65
4.2.2	Kriging Results - Sample Size - Dataset 1	66
4.2.3	RBFNN Results - Sample Size - Dataset 1	67
4.2.4	Metamodel Comparisons - Sample Size - Dataset 1	68
4.2.5	Metamodel Comparisons - Sample Size - Dataset 2	70
4.2.6	Metamodel Comparison with Ideal Values for RBFNN	72
4.3	Extrapolation vs. Interpolation Results	73
4.3.1	PRS Results - Standard Deviation - Dataset 1	75

4.3.2	Kriging Results - Standard Deviation - Dataset 1	76
4.3.3	RBFNN Results - Standard Deviation - Dataset 1	77
4.3.4	Metamodel Comparisons Based Upon Standard Deviation	78
5	CONCLUSIONS AND FUTURE WORK	81
5.1	Polynomial Response Surface Conclusions	81
5.2	Kriging Approximation Conclusions	82
5.3	Radial Basis Function Neural Network Conclusions	83
5.4	Future Work	84
5.5	Acknowledgements	85

LIST OF FIGURES

Figure 1:	Schematic of the engineering design process. (http://www.sallyridescience.com/toychallenge/design)	1
Figure 2:	A digital prototype created in the Solidworks virtual prototyping package. (http://www.deigner.com/design_news/solidworks_2008.html)	4
Figure 3:	A Solidworks finite-element analysis simulation of a digital prototype. (Left- http://www.dcwhite.co.uk/stress_fe.htm) A computational fluid dynamics simulation on a fighter jet digital prototype. (Right- http://iar-ira.nrc-cnrc.gc.ca/aero/aero_4.html)	6
Figure 4:	A screen capture of a digital prototype constructed using the "lightened" version of the CATIA CAD package-CATIA PLM Express.	8
Figure 5:	A chart demonstrating Microsoft Excel's Line of Best Fit functionality.	14
Figure 6:	Architecture of a radial basis function neural network.	20
Figure 7:	Systems of equations and the resulting metamodels using different metamodeling techniques.	24
Figure 8:	A top view of the loading rig for simulating a wheel loading legacy dataset. (Left) A bottom view of the loading rig. (Right) A close up image of the FEA simulation of the reaction forces at a wheel location. (Bottom)	31
Figure 9:	An aerial boom truck in a static position. (Left- http://www.machinerytrader.com/listings/detail.aspx?ohid=5324527) An aerial boom truck setup with the boom extended. (Right- http://www.machinerytrader.com/listings/detail.aspx?ohid=5324527)	34
Figure 10:	The sub-base of the boom truck with the FEA loading conditions attached to the outrigger plates and the pedestal.	35
Figure 11:	An image of the FEA simulation results for the sub-base displaying displacement properties.	37
Figure 12:	A sample Kriging metamodel report generated after constructing the metamodel.	50
Figure 13:	Schematic of the ASDS system architecture.	54
Figure 14:	The tipping angle tool displays the smallest angle the product will tip based upon the selected wheels.	57

Figure 15:	The wheel loading at each support position is graphically displayed to the user.	59
Figure 16:	A chart displaying the RMSE values fro PRS based upon sample size.	65
Figure 17:	A chart displaying the RMSE values for Kriging based upon sample size.	66
Figure 18:	A chart displaying the RMSE values for RBFNN based upon sample size for Dataset 1.	67
Figure 19:	A chart displaying the RMSE values for each metamodeling technique based upon sample size for Dataset 1.	68
Figure 20:	A chart displaying the RMSE values for each metamodeling technique based upon sample size for Dataset 2.	70
Figure 21:	A chart displaying the RMSE values for each metamodeling technique based upon sample size for Dataset 2 with the ideal radius values used for RBFNN.	72
Figure 22:	A chart displaying RMSE values for PRS based upon standard deviation.	75
Figure 23:	A chart displaying the RMSE values for Kriging based upon standard deviation.	76
Figure 24:	A chart displaying the RMSE values for RBFNN based upon standard deviation.	77
Figure 25:	A chart displaying the RMSE values for each metamodeling technique based upon standard deviation.	78

LIST OF TABLES

Table 1:	A subset of the full wheel loading dataset with a standard deviation of 0.5.	32
Table 2:	A subset of the full wheel loading dataset with a standard deviation of 1.0.	33
Table 3:	The mean and standard deviation of the design variables for the stress and displacement analysis dataset.	36
Table 4:	A subset of the full dataset of the results of the Solidworks FEA simulations for the boom truck stress and displacements analysis.	38
Table 5:	Number of articles on Google Scholar for each metamodeling technique.	39
Table 6:	Correlation parameters of Y1 for the wheel loading dataset with various global models.	44
Table 7:	Correlation parameters of Y1 for the stress analysis dataset with various global models.	44
Table 8:	The setup parameters for the Kriging metamodel built upon the wheel loading dataset of 50 sample points with a standard deviation of 1.0.	49
Table 9:	The predicted values for the output variable Y1 for the Kriging Approximation example.	51
Table 10:	A statistical summary of the performance of the Kriging approximation example for Y1.	52
Table 11:	An average of the performance characteristics for Y1-Y4 for the Kriging approximation example.	52
Table 12:	Shows the number of FEA and metamodel simulations required for the entire study.	61
Table 13:	An overall summary of statistical measurements for each of the three metamodeling techniques.	62

ABSTRACT

The engineering design process has undergone extensive research in the area of detailed design. Many computer aided design (CAD) software packages have been developed from this research to provide an integral analysis tool for companies in the detailed design phase. However with the development of more complex technologies and systems, decisions made earlier in the design process have been crucial to product success. To help provide valuable information to assist these earlier decisions, tools have also been developed for conceptual design such as lightened CAD packages, concept elimination methods, and image processing software. Unfortunately, these tools have been proven ineffective based on the inability to provide a lower fidelity real-time analysis of each and every concept. By providing real-time analysis, engineers could spend more time evaluating every concept mathematically and base decisions on factual information instead of personal opinion.

On a different note, companies continually undergo next generation development of their products. This continuous cycle of design iterations generates a stockpile of high fidelity analysis which we refer to as “legacy data.” Legacy data contains thousands of geometrical properties and analytical data used to assess the validity of previous designs. This data creates a vast amount of analytical engineering knowledge which can be harnessed to help evaluate the validity of future designs. Statistical approximations known as metamodels can be applied to summarize the general trends of the inputs and outputs of legacy dataset, and

eliminate the need for recreating CAD analysis models for each concept. Metamodeling techniques cannot produce 100% accuracy, but at the conceptual design stage, 100% accuracy is not a necessity. This thesis presents an implementation scheme for incorporating Polynomial Response Surface (PRS) methods, Kriging Approximations, and Radial Basis Function Neural Networks (RBFNN) into conceptual design. A conceptual design software application, the Advanced Systems Design Suite (ASDS), has also been developed to incorporate these metamodeling techniques into assessment tools to evaluate conceptual design concepts in both a desktop and immersive virtual reality (VR) environment.

The goal of the implementation scheme was to develop a strategy for constructing metamodels upon conceptual design datasets based upon their ability to perform under several conditions including various sample sizes, dataset linearity, interpolation within a domain, and extrapolation outside a domain. In order to develop the implementation scheme, two conceptual design datasets, wheel loading and stress analysis, were constructed due to a lack of available legacy data. The two datasets were setup using a design of experiments (DOE) to generate accurate sample points for the datasets. Once the DOE was formulated, digital prototypes were created in CAD software and the FEA test runs generated the responses of the DOE input parameters. The results of these FEA simulations generated the necessary conceptual design datasets required analyze the three metamodeling techniques.

The performance results revealed that each metamodeling technique outperformed the others when tested against the various parameters. For instance,

PRS metamodels performed very well when extrapolating outside its domain and with datasets consisting of more than 40 sample points. PRS metamodels require very setup and can be generated very quickly. If speed is the key consideration for metamodel construction, then PRS is the best option. Kriging metamodels showed the best performance with any non-linear dataset and large design space datasets exhibiting linear or non-linear behavior. Kriging metamodels are a very robust metamodeling technique especially when using a first-order global model on non-linear datasets. On the downside, Kriging metamodels require slightly more time to setup and construct than PRS metamodels. RBFNN metamodels performed well when interpolating within a large design space and on any sample size of linear datasets. However to reach performance levels of either PRS or Kriging, the ideal radius value must be determined prior to constructing the final model which took hours on small datasets. If the datasets consisted of thousands of design variables, constructing a RBFNN metamodel would take days to weeks to generate. However if construction time is not an issue, RBFNN metamodels outperform both PRS and Kriging techniques on linear datasets.

This implementation scheme for incorporating metamodels into conceptual design provides a method for generating rapid assessment capabilities as an alternative to high fidelity analysis. Future work includes evaluating additional conceptual design datasets to create a more robust implementation scheme. More research will also be done in implementing additional types and varying setup parameters of both Kriging Approximations and Radial Basis Function Neural Networks.

1 INTRODUCTION

1.1 The Design Process

The product design process has been explained and defined many different ways. For now and for the ease of understanding, let's divide it up into three base groups including “understanding the opportunities”—steps 1-2 of Figure 1, “conceptual design”—steps 3-4, and “detailed design”—steps 5-8. The first phase, “understanding the opportunities,” involves several different sub-categories, which combine to reach a common goal of creating a vision for the new product to be put into production. This is done through brainstorming and researching the problem at

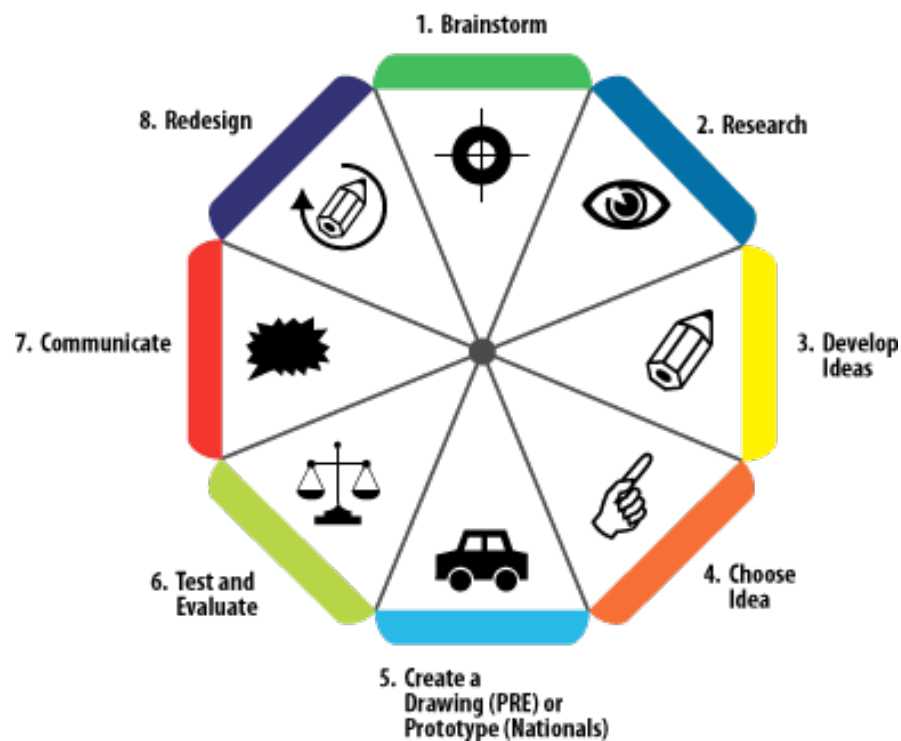


Figure 1: Schematic of the engineering design process. (<http://www.sallyridescience.com/toychallenge/design>)

hand. Companies often brainstorm to find answers to some of the following questions at this stage in design:

- What product do we wish were out there?
- What is difficult with our current product?
- Why does it not perform the way we want it to?
- What are our customer needs?

Once these questions have been researched and answered accordingly, the exact specifications and goals for the product such as performance, quality, and safety can be explicitly defined. The product is then ready to move forward to the second phase of design of developing a new concept.

Once the product moves into the conceptual phase of the design process, the next step is to create a functional model describing the inputs, outputs, and transformations that must happen for a product to work according to the market specifications and customer needs. This functional model is then used by engineers and design teams for generating many concepts to implement the functional specifications. After the concept pool is generated, the design team undergoes a selection process to determine the best idea. The result of this analysis is the concept to develop and move forward into the third phase of design.

The next phase of the design process, detailed design, involves first developing virtual models or physical prototypes to represent the final product for testing and assessment purposes. After the model is constructed, mathematical methods of analysis are constructed to evaluate the ability of the model to meet the needs of the customer as well as the functional model. At the end of this phase, a

working prototype or virtual model exists. Manufacturing, assembly, and maintenance process design of the product are then undertaken.¹

The entire design process is full of many difficult decisions based sometimes on very little hands-on information. Engineers are sometimes forced to make decisions based solely on instinct and experience instead of factual information. Since each decision made along the way is imperative to the success or failure of both the design and functionality of the final product, engineers need to have as much information as possible to make the correct decision. If the correct decisions can be made throughout the design process, the final product will benefit in a number of different ways. Some of these benefits include improving overall product quality, decreasing production time, and decreasing production costs.

It is estimated that up to 75% of product cost is spent during the product design phase including maintenance and manufacturing.² Therefore, the design process has undergone extensive research in both academia and industry to improve the cost effectiveness of the design process as a whole. This research generated many commercial software solutions to optimize the detailed design phase. Software packages such as Solidworks³ (see Figure 2), ProEngineer⁴, and Abaqus⁵, enable engineers to create digital prototypes of a product design. The digital prototype can then be subjected to various analytical tests which measure the validity of the design.

A large collection of today's product driven companies have integrated sophisticated prototyping and simulation software into the detailed design phase of their design processes. These software packages have become a necessity to the

engineering community. Physical prototyping is an enormously expensive, time-consuming process for companies producing extremely high price items such as airline jets, combine, or ocean-liner. Not only are the materials alone extremely expensive for such products, the costs of materials and construction labor can become enormous for such products. Additionally, the company needs to perform in-depth analysis to ensure the prototype was built within the tolerance specifications. With the time and money consumed by the prototyping process, physical prototyping is being continually diminished in many engineering companies because digital prototyping (see Figure 2) allows them to decrease the development time and cost of the final product in addition to avoiding significant physical prototyping costs.⁶

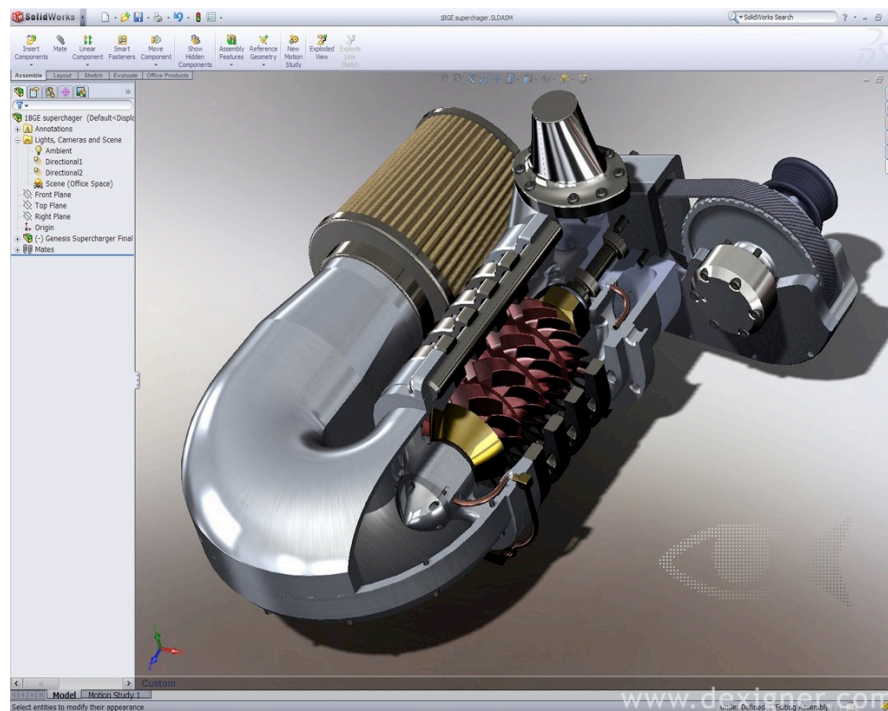


Figure 2: A digital prototype created in the Solidworks virtual prototyping package. (http://www.designer.com/design_news/solidworks_2008.html)

1.2 The Reality of Conceptual Design

With the development of more complex technologies and systems, decisions made in earlier stages of the design process become critical. Once a product reaches the detailed design phase, many specific properties such as dimensions and material properties need to be determined as the product will soon be moving into production. Otherwise, the company will incur significant delays and costs due to the redesign of the product.

For example, the detailed design phase consists of several stages including prototype creation, whether it be digital and/or physical, detailed analysis, performance evaluation, and possible redesign. Using a complex computer aided design (CAD) software package to create a digital prototype can take considerable amounts of time, especially with more intricate systems. Afterwards, complex assessment methods built into the CAD packages such as finite-element analysis (FEA) and dynamic simulations like computational fluid dynamics (CFD) are used to assess performance characteristics of a design. These performance criteria are then used to evaluate the validity of the design. If after all this time the design is found to be infeasible, the company has invested considerable time and money into the development of a product design which will never be put into production.

This example illustrates how important the early stages of the design process are to the success of the product design cycle. In order to optimize the engineering design process, it is imperative that correct decisions are made as early as possible. Companies cannot afford the time and money required to send invalid solutions

forward into detailed design. Therefore, early stages of the design process such as conceptual design are currently undergoing large amounts of research in both the academic and industrial communities to try to improve the quality of decisions and ideas made and produced in this phase of design. Additionally, conceptual design decisions can be made with minimal cost and time impact yet can produce significant impacts on the downstream design and manufacturing process.⁷ Since these initial decisions have such a large impact on downstream development, engineers need tools to make accurate decisions at the conceptual stage. Unfortunately, the number of conceptual design tools available is very limited.

The most prevalent tool used today in conceptual design is CAD software. However, these packages contain complex CAD functions such as precise dimensional mating, 3D meshing, finite-element analysis (FEA), and computational fluid dynamics (CFD) simulations (see Figure 3). These types of functions were not

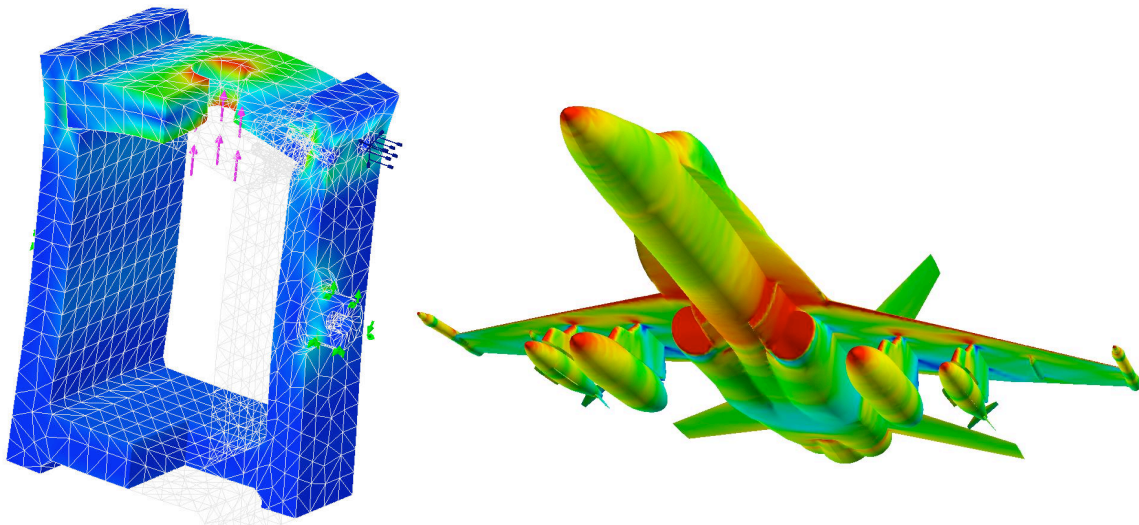


Figure 3: A Solidworks finite-element analysis simulation of a digital prototype. (Left-http://www.dcwhite.co.uk/stress_fe.htm) A computational fluid dynamics simulation on a fighter jet digital prototype (Right-http://iar-ira.nrc-cnrc.gc.ca/aero/aero_4.html)

developed for conceptual design but for detailed design. Conceptual design requires tools to estimate the look, feel, and performance of a particular concept quickly and with as much accuracy as possible. Assessment estimates of certain key conceptual design parameters are all that is necessary in this design stage. Full FEA and CFD simulations and analysis is just not possible. Running these analysis simulations are extremely difficult and time-consuming to setup and run even when the engineers are highly trained and experienced as many of the specific inputs needed have to be estimated for a concept design. To generate FEA or CFD analysis on many different concepts is unrealistic and simply cannot be accomplished in the limited amount of time available for engineers to establish the specifications of the concept to move into the next design phase. Therefore, by using CAD software packages for conceptual design, only a small subset of the concepts could be run through such simulations. This can lead to design teams dismissing possible solutions without data to support this decision. Certain design characteristics can be eliminated due to the inability to evaluate the validity of all the concepts using this method.

Commercial CAD developers have released “lightened” versions of their software such as PTC’s Pro/CONCEPT⁸ and CATIA PLM Express⁹ (see Figure 4) to try to reduce the limitations of 3D model creation. These packages reduce the amount of analytical functionality of the software, yet the complexity of 3D model creation such as precise mating and dimensioning still remains to be an unresolved issue. Requiring extensive training and a large learning curve, these lightened applications still do not meet the real-time creation and analysis requirements of digital prototyping at the conceptual design phase of the design process.

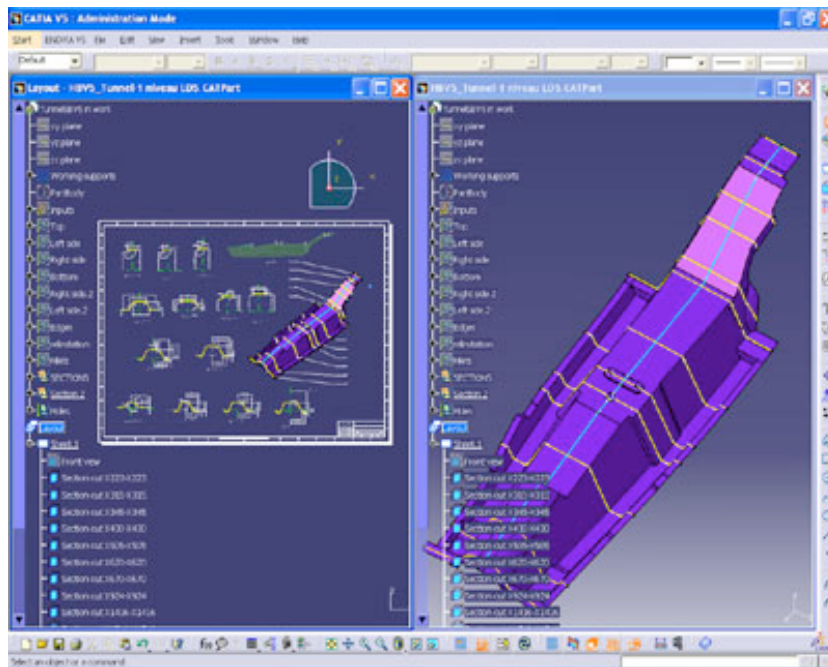


Figure 4: A screen capture of a digital prototype constructed using the “lightened” version of the CATIA CAD package—CATIA PLM Express.

Researchers have also developed other selection methods to rank the population of concepts. The goal of all these selection processes is to obtain as much information and concept details as cycle time and resources permit. Some of these selection processes include estimating technical difficulty and Pugh concept selection charts.¹ Estimating technical difficulty is a method for engineers to compare known measurements and a model with that of a new concept to estimate the concept’s performance based on the information from the pre-existing data. Pugh charts are a more common selection method, which use a minimal evaluation scale and three overall ranking metrics to perform concept selection when there is minimal information quality available. These methods have been proven effective but are

based on the opinions of engineers and not on hands-on factual data. In order to use these methods more effectively, more information needs to be provided to the engineers before using these methods to make concept selections.

In a traditional conceptual design exercise, engineers work with a variety of these tools and other experts to help visualize and convey their ideas. For example, the conceptual design team may turn to CAD packages, industrial design systems, image processing software such as PhotoShop, and sometimes even graphic artists to capture the overall shape and configuration of a particular concept. This is often a tedious and time-consuming process, but is still generally chosen over developing the entire 3D model in a “lightened” CAD package. The end result is generally several 2D images useful only as a visual reference to provide visual clarity amongst all the group members. There is not any factual information tied to analyze the validity of the concept nor the ability to make and assess changes in real-time.

1.3 Motivation

All four conceptual design methods—CAD packages, lightened CAD packages, concept selection methods, and traditional methods—have their advantages and purposes. However, none of these methods are able to achieve what is truly needed in conceptual design; fast geometry creation and lower fidelity real-time analyses of each concept to make accurate decisions early in the design process. Currently, CAD software provides high fidelity analysis to evaluate all the concepts, but this process is just too time consuming. Additionally, concept

elimination methods do not use design analysis tools to make concept selection and/or elimination decisions.

The need for high fidelity analysis in the detailed design phase is apparent. This is obvious since almost all companies have by now adopted digital prototyping into their detailed design process to cut various production costs. Digital prototyping creates vast amounts of high fidelity computational data to effectively assess the performance of a product. Once the product is completed and production begins, the company then begins the design process for the next generation of the product or a new product all together. This continuous cycle of design iterations builds a stockpile of high fidelity analysis data which is referred to as “legacy data.”

Legacy data contains thousands of geometrical properties and analytical data used to assess the validity of previous generations of product designs. This data creates a vast amount of analytical engineering knowledge which could be harnessed to help evaluate the validity of future designs. If general trends in the data exist, then statistical approximations known as metamodels can be applied to summarize this legacy data. For example, if twenty generations of a particular product exist and the same set of high fidelity analysis simulations were performed on each generation, then the input variables and results can be summarized mathematically using a metamodel.

By generating metamodels to interpolate and extrapolate high fidelity legacy data, the need for recreating CAD analysis models can possibly be eliminated. Metamodeling techniques cannot produce 100% accuracy for representing their

input data, but at the conceptual design stage, 100% accuracy is not a necessity at this early stage of design.

Metamodeling driven analysis tools can then facilitate a design by shopping paradigm.¹⁰ After the analysis of the population of concepts is concluded, the design team sorts through the analytical results shopping for an optimal design. If an optimal solution for the detailed design does not yet exist, the best concepts are selected and another iteration of concept development is undergone with the newly selected concepts. This process continues until one or several concepts are found to move forward into detailed design. This design by shopping paradigm based upon the analytical data provided from the metamodels can then be coupled with additional concept selection methods as mentioned previously such as estimating technical difficulty and Pugh selection charts. It is just important to note that none of these methods are effective without actual data.

1.4 Thesis Organization

This thesis focuses on the ability of different metamodeling techniques to best represent conceptual design data to create real-time assessment tools at the conceptual design phase of the design process. In Chapter 2 a literature review of Polynomial Response Surfaces (PRS), Kriging Approximations, and Radial Basis Functions (RBF), and Neural Networks is given. In Chapter 3 the methodology section begins with the construction of two conceptual design datasets to build the metamodels upon. The first is a legacy data wheel loading approximation, and the

second is a stress and displacement approximation. These datasets serve as evaluation tools to examine the performance between all three metamodeling techniques. Following the dataset construction is a section describing how each of the three metamodels were constructed and validated. In Chapter 4 the comparison results of the metamodel performance evaluation are presented. Chapter 5 contains a summary and discussion of the results in addition to future work.

2 LITERATURE REVIEW

2.1 Introduction to Metamodeling

Metamodeling is the general process of creating an approximation of a response or set of responses of a dataset over a certain domain. In more general terms, metamodels estimate the output(s) of a given set of inputs built upon the outputs of a similar dataset. This idea is best illustrated with a simple example of curve fitting using the line of best fit functionality built into Microsoft Excel. First, a set of data must be generated in order to have information to construct an approximation or estimation upon. Let's use the function seen in Eq. (1)

$$y = \sqrt{x} \quad (1)$$

To generate a dataset the function is evaluated at 200 equally distributed sample points in the domain or range of x from 0 to 20. The results or y -values of the evaluations can be seen below in Figure 5 as the orange line. This orange line represents a plot of Eq. (1) for the domain chosen.

When metamodeling is used, the exact function representing the data, Eq. (1) in this example, is unknown. The scattered dataset is generated from a real world scenario and not from an experimental equation. Therefore, engineers do not have any idea of what the mathematical representation of the results is. Additionally, the data tends to have noise and outlier data points instead of a nice smooth curve. In order to represent Eq. (1) in a more realistic setting for metamodel construction, random noise was introduced into Eq. (1) and the resulting data points can be seen

in Figure 5 as the blue noisy y-values points scattered about. This is a much more realistic dataset for which metamodels tend to be built upon.

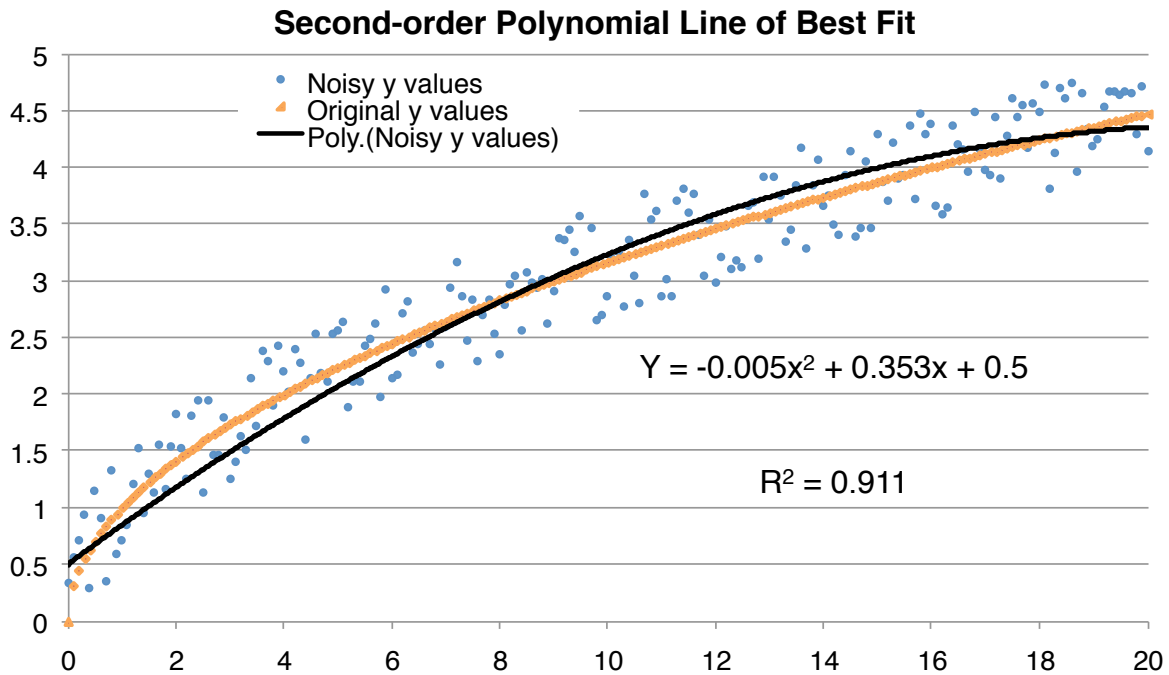


Figure 5: A chart demonstrating Microsoft Excel's Line of Best Fit functionality.

After the noisy dataset was built from Eq. (1), an approximation or metamodel of the scattered dataset can be built. The goal of the approximation is to create an equation as close to the actual trend of the data as possible. For this example, the general trend of the scattered dataset is Eq. (1). However, in most real metamodeling scenarios, the general trend is unknown. To construct a simple metamodel for this example, the line of best fit functionality of Microsoft Excel was used. The black line in Figure 5 is a second-order polynomial trend line built upon the scattered dataset. It uses the values of the scattered sample points to build an equation to estimate the y-value at a given x-value inside the domain. This equation and the R^2 value (coefficient of determination) are both displayed in Figure 5.

Anyone can then find the predicted y-value of the dataset by simply calculating the equation for any value of x inside the domain.

The R^2 value of Figure 5 is a measure of how well the metamodel fits the scattered dataset. A value of zero indicates that the metamodel does not fit the scattered dataset at all, and a value of one indicates a perfect fit. Therefore, when constructing metamodels, one useful measure of the metamodel's ability to fit the dataset is the R^2 value. This is useful but cannot predict how close the metamodel is to the actual trend. For example, the estimate line (black) and the actual line (orange) are not exactly the same and the R^2 does not accurately represent this difference.

The goal of metamodeling is to capture the general trend of the scattered dataset or make the black and orange lines as similar as possible. Therefore, an assumption of metamodeling is that the scattered dataset is a good representation of the general trend of the dataset. Therefore, if a metamodel is a highly accurate representation of the scattered dataset, then the metamodel is also a highly accurate representation of the general trend as well. To accurately model as many types of datasets as possible, statisticians and engineers have developed different metamodeling techniques. The next section provides an overview of each of the three different metamodeling techniques: 1) Polynomial Response Surface (PRS) Methodology, 2) Kriging Approximations, and 3) Radial Basis Function Neural Networks (RBFNN).

2.2 Overview of Metamodeling Techniques

2.2.1 Polynomial Response Surfaces

Polynomial Response Surface methodology is widely used throughout the engineering community¹¹ and was originally developed to analyze the results of physical experiments to create statistical models of the experimental results.¹² PRS models are designed to approximate datasets using polynomial expressions to fit the dataset and take the form

$$y(x) = f(x) + \epsilon \quad (2)$$

where $y(x)$ is the unknown function of interest, $f(x)$ is a known polynomial function of x , and ϵ is random error. The random error is assumed to be normally distributed about zero. The known polynomial function, $f(x)$, is generally a low-order polynomial. In order to satisfy more non-linear behavior, higher order polynomials can be used but require large numbers of sample points to satisfy the coefficients in the polynomial equation. In Eq. (3) the polynomial equation is linear. Eq. (4) shows a second-order or quadratic expansion of the polynomial equation.

$$\hat{y} = \beta_0 + \sum_{i=1}^k \beta_i x_i \quad (3)$$

$$\hat{y} = \beta_0 + \sum_{i=1}^k \beta_i x_i + \sum_{i=1}^k \beta_{ii} x_i^2 + \sum_{i=1}^k \sum_{j < i} \beta_{ij} x_i x_j \quad (4)$$

The coefficient parameters, β_0 , β_i , β_{ii} , and β_{ij} , of the polynomials in Eqs. (3) and (4) are determined through least-squares regression. Least-squares regression first calculates the partial derivatives of the coefficients, then minimizes the sum of the squares of the residuals of predicted values, $\hat{y}(x)$, from the actual values, $y(x)$. The coefficients of Eqs. (3) and (4) can be found using Eq. (5)

$$\beta = [X'X]^{-1}X'y \quad (5)$$

where X is the design matrix of sample data points, X' is its transpose, and y is a column vector that contains the values of the response at each sample point.¹³

2.2.2 Kriging Approximations

Kriging metamodels originated from mining and geostatistical applications for the Rand gold deposit.¹⁴ Kriging models are formulated through a combination of a global regression model and localized departures

$$y(x) = f(x) + z(x) \quad (6)$$

where $y(x)$ is the unknown function of interest of design variable x , $f(x)$ is a known approximation (usually polynomial) function of x , and $z(x)$ is the realization of a stochastic process with mean zero, variance σ^2 , and nonzero covariance. The $f(x)$ term in Eq. (6) is very similar to the polynomial approximation for polynomial response surface models. Each provide a global approximation of the design space.

In order to better fit the model to the design space, $z(x)$ created localized deviations so the Kriging model interpolates the n_s sampled data points more accurately in large residual areas. The covariance matrix of $z(x)$ is represented by

$$\text{Cov}[z(x^i), z(x^j)] = \sigma^2 \mathbf{R}[R(x^i, x^j)] \quad (7)$$

where \mathbf{R} is a correlation matrix of dimensions $(n_s \times n_s)$ with ones along the diagonal.

The correlation function $R(x^i, x^j)$ is specified by the user and lies between any two sampled data points x^i and x^j . A large variety of correlation functions exist including linear, exponential, general exponential, gaussian, spherical, and spline functions.

^{15,16} The following Gaussian correlation function was utilized

$$R(x^i, x^j) = \exp \left[- \sum_{k=1}^{n_{dv}} \theta_k |x_k^i - x_k^j|^2 \right] \quad (8)$$

where n_{dv} is the number of design variables, θ_k are the unknown correlation parameters used to fit the model, and x_k^i and x_k^j are the k th components of sample points x^i and x^j . Different values of θ for each design variable are used instead of using a single correlation parameter.¹⁷

Predicted values, $\hat{y}(x)$, of the response $y(x)$ at untried values of x can be estimated using Eq. (9)

$$\hat{y} = \hat{\beta} + r^T(x) \mathbf{R}^{-1} (y - f \hat{\beta}) \quad (9)$$

where y is a column vector of length n_s with responses of the sampled data point $\{x^1, \dots, x^{n_s}\}$, and f is a column vector of length n_s filled with ones when $f(x)$ is taken as a constant. In Eq. (9) the correlation vector $r^T(x)$ of length n_s between a new x and the sampled data points can be expressed as:

$$r^T(x) = [R(x, x^1), R(x, x^2), \dots, R(x, x^{n_s})]^T \quad (10)$$

In Eq. (9) the generalized least squares estimate, $\hat{\beta}$, is estimated using Eq.

(11)

$$\hat{\beta} = (f^T R^{-1} f)^{-1} f^T R^{-1} y \quad (11)$$

The estimate of the variance, $\hat{\sigma}^2$, in Eq. (7) between the underlying global model $\hat{\beta}$ and y is estimated using Eq. (12):

$$\hat{\sigma}^2 = \left[(y - f\hat{\beta})^T R^{-1} (y - f\hat{\beta}) \right] / n_s \quad (12)$$

where $f(x)$ is assumed to be the constant $\hat{\beta}$. The maximum likelihood estimates for the θ_k in Eq. (8) are determined using Eq. (13)

$$\max \phi(\theta_k) = [n_s \ln(\hat{\sigma}^2) + \ln |\mathbf{R}|] / 2 \quad (13)$$

where both $\hat{\sigma}^2$ and $|\mathbf{R}|$ are functions of θ_k . Any value of θ_k can be used to approximate the Kriging model, but to generate the best Kriging model, a simulated annealing algorithm¹⁸ is used to solve the k-dimensional unconstrained nonlinear optimization problem given by Eq. (13) to find the maximum likelihood estimates for the θ_k parameters.¹⁹

2.2.3 Radial Basis Function Neural Networks

Radial basis function (RBF) neural networks provide a powerful technique for generating multivariate, nonlinear mappings.²⁰ Training an RBF network consists of a single output in the design space being calculated from a linear space of single or multiple dimensions. RBF networks have a three layer architecture which can be seen in Figure 6. The first layer of the network consists of “input” units whose number is equivalent to the number of independent variables of the problem. The second layer is composed by nonlinear “hidden” units fully connected to the first layer. A single unit for each data point $x_i = (x_i, y_i, z_i, \dots)$ exists parametrized by its

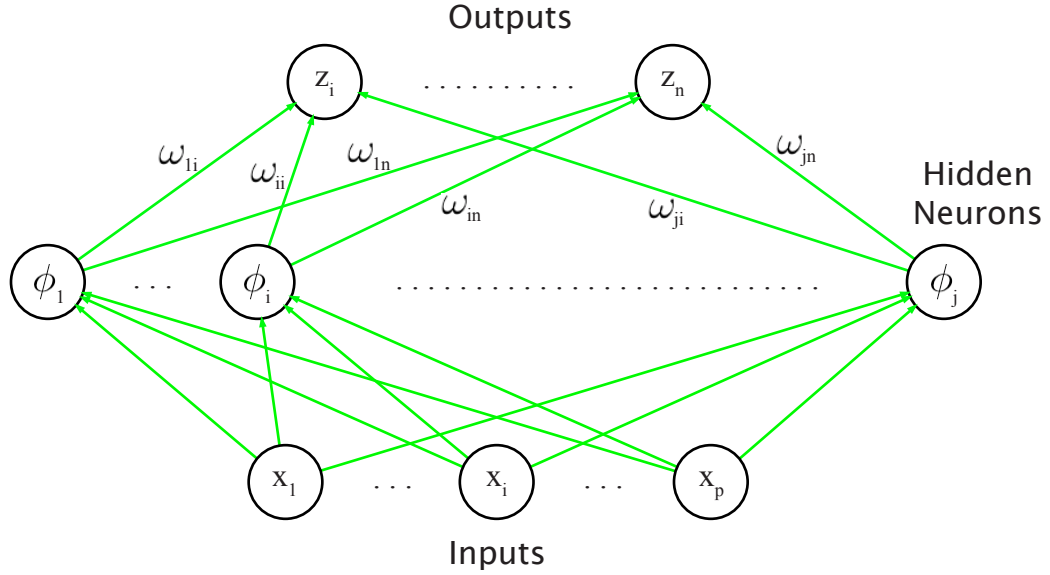


Figure 6: Architecture of a radial basis function neural network.

center, which has the coordinates (x_i, y_i, z_i, \dots) of the data point itself. The output layer, fully connected to the hidden layer, consists of one or more linear units, whose weights are the unknown coefficients of the RBF expansion.²¹

Vectors containing the input values are passed into a hidden layer of neurons which computes a hyperspherical function of x , the input vectors. The output of each of these i th hidden neurons can then be generalized as

$$\phi_i = \phi(||x - y_i||) \quad (14)$$

where y_i is the center of the RBF for neuron i , and $(||\dots||)$ denotes the Euclidean norm, magnitude, for the hypersphere created at neuron i . The nonlinear function ϕ can be chosen in many different ways and can be different for each hidden neuron.

There are many different radial basis functions including linear, multiquadric, and Gaussian functions in addition to thin-plate and Sobolev splines. For this thesis a

Gaussian was used as the radial basis function for the hidden neuron as is typically used and can be seen in Eq. (15)

$$\phi(x) = \exp\left(-\frac{(x - y_i)^2}{r^2}\right) \quad (15)$$

where y_i is the center point and r is the radius of the Gaussian RBF.²²

The outputs of the RBF neural network are then formed from the weighted sum of the outputs from each of the hidden neurons

$$z_i = \sum_j \omega_{ij} \phi_j + \theta_i \quad (16)$$

where the weights w_{ij} and the biases θ_i are adaptive variables set during the learning phase. In order to find the correct synaptic weights for the hidden neurons, training data are supplied to the network in the form of pairs x_p, t_p of input and target vectors. The learning algorithm then uses generalized least squares to minimize the sum-of-squares error for the training data defined by

$$E^S = \frac{1}{2} \sum_p \sum_i (z_{ip} - t_{ip})^2 \quad (17)$$

where $z_{ip} = z_i(x_p)$ denotes the output coming from neuron i when the network is given x_p for training input data. When a minimum sum-of-squares error is reached then Eq. (18) is valid

$$\frac{\partial E^S}{\partial \omega_{ij}} = 0 \quad (18)$$

Eq. (18) combined with RBF network outputs in Eq. (16) (after omitting the explicit bias terms) gives

$$\sum_p \left[\sum_k \omega_{ik} \phi_{kp} - t_{ip} \right] \phi_{jp} = 0 \quad (19)$$

Where $\phi_{jp} = \phi_j(x_p)$. Eq. (18) can then be simplified to

$$\sum_k \omega_{ik} M_{kj} = \sum_p \phi_{kp} \phi_{jp} \quad (20)$$

where the square matrix M is defined by

$$M_{kj} = \sum_p \phi_{kp} \phi_{jp} \quad (21)$$

M is the covariance matrix of the transformed data with a zero mean. In order to solve the weights for each neuron, M^{-1} is computed to solve Eq. (22)

$$\omega_{ij} = \sum_k (M^{-1})_{kj} \left[\sum_p \phi_{kp} t_{ip} \right] \quad (22)$$

Typically the number of hidden neurons equals the number of inputs with the centers located at the input location. This allows the RBF network to fit all the input data points into the model. Therefore, highly accurate nonlinear approximations are possible with RBF networks. Fitting the RBF network to each sample point generally creates an overfit of the dataset instead of observing a more generalized trend. Other implementations however exist for RBF to help account for overfitting. These techniques will be discussed in more detail in the following section.

2.3 Development of Metamodeling techniques

Metamodeling originated back in the early 1980s. Schmit et al.^{23,24} combined finite element analysis and mathematical programming algorithms to create structural synthesis, a method to reduce the structural weight of aerospace and automotive structures without compromising the structural integrity of the design.

Then in early 1990s, response surface methods first developed by Box and Wilson^{25,26} became the primarily focus of approximation methods research. Several different response surface methods were developed during this time. For instance, the Variable Complexity Response Surface Modeling (VCRSM) method^{27,28} used analyses datasets of varying fidelity to reduce the area of the design space to a more effective region of interest. Then from this smaller subspace, the response surfaces were built to see a higher accuracy due to the model being built in a more concise and accurate domain.

Robust Design Simulation²⁹ combined both response surface models and Monte Carlo simulation to generate robust design specifications. Robust Concept Exploration Method^{30,31} integrated response surface approximations into the early stages of complex systems design to explore the design space to determine a suitable range for specifications and to identify feasible starting points for design. This was called a robust concept exploration method because by using this method, quality concepts were identified before concept development began.

In the mid 1990s when response surfaces had undergone more research and were better understood, some limitations of this metamodeling technique became

apparent. Two of these shortcomings were the curse of dimensionality^{32,33} and the inability to generate accurate approximations on highly non-linear design space datasets¹⁹. These inabilities led to further development in both higher order response surface models³⁴ and mixed polynomial models³⁵ to try to better handle the highly non-linear datasets. Another approach was to find more efficient ways to sample the design space effectively with smaller datasets using computer analyses.^{36,37,38}

In the late 1990s, the focus shifted away from response surface models to alternative approximation methods such as radial basis functions³⁹. To better handle noisy datasets, curve-fitting procedures^{40,41} were developed to approximate only the smooth components of noisy and scattered data. The Extended Radial Basis Function (E-RBF) approach⁴² was also designed to impose smoothness constraints on the metamodel. This was accomplished by incorporating more flexibility in the metamodel by introducing additional degrees of freedom by adding coefficients into each dimension of the metamodel definition. A two dimensional example of Response Surface Methodology, Radial Basis Functions, and Extended Radial Basis Functions can be seen in Figure 7 below.

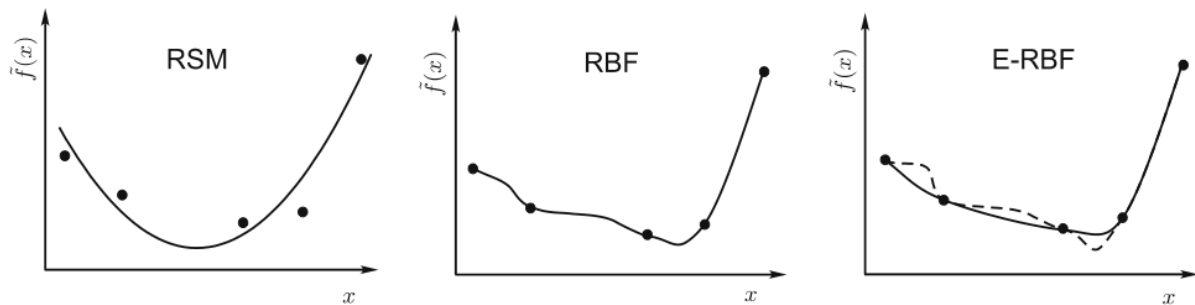


Figure 7: Systems of equations and the resulting metamodels using different metamodeling techniques.

While Radial Basis Functions were under development, so were Kriging Approximations⁴³. Other additional extensions of Kriging were implemented to try to create a more robust metamodel. Gano et al.⁴⁴ built Kriging based scaling functions alongside a trust-management scheme to get low fidelity models to converge to higher fidelity models. Gano et al.⁴⁵ later enhanced this method by introducing a metamodel update management system to reduce the cost of rebuilding a Kriging model. This was done by only updating the model parameters when producing a poor approximation. Another Kriging implementation is Co-kriging which adds additional gradient information to the unknown primary functions of interest at each specified sample location.⁴⁶ Forrester et al.⁴⁷ later combined Co-kriging with a Bayesian model update criterion based on an error estimate that reflects the amount of noise in the observed data.⁴⁸

Each of these methods have undergone significant research to improve performance and robustness. During these developments, engineers and statisticians used various types of detailed design production models as test datasets. Some of the engineering applications where metamodeling has already been applied include solid bar torsion and elongation⁴⁹, aircraft structural performance⁵⁰, aerodynamics⁵¹, shell structure buckling⁵², blade-stiffened composite panels⁵³, optimizing flow of a diffuser⁵⁴, and optimizing an aerospike nozzle in a rocket engine¹⁹. These are obviously not all the examples of metamodeling incorporated into design, but merely a few to demonstrate the integration of metamodeling into the design process.

2.4 Research Issues

Based on the literature of current research in various metamodeling techniques including Polynomial Response Methodology, Kriging Approximations, and Radial Basis Function Neural Networks for mathematical approximations in conceptual design, three research issues have been identified. They are:

- 1. To determine which metamodeling technique is best suited for large vehicle conceptual design based upon sample size performance.**

Building metamodels from conceptual design datasets can be challenging due to a small availability of sample points. It is crucial that metamodels possess the capability of generating high fidelity results from small sample datasets. This analysis would provide engineers with the knowledge to determine how many data points are necessary in order to construct an effective metamodel.

- 2. To determine if these metamodeling techniques can accurately interpolate and/or extrapolate responses inside and/or outside computational domains appropriate for large vehicle conceptual design.**

Most conceptual design exercises push the limits of preexisting designs. If metamodels were constructed upon the preexisting designs, how well could they then extrapolate responses outside their domain? Additionally, if metamodels are constructed on many different types of preexisting designs, can these

metamodels effectively interpolate for more specific design cases within their domain?

3. To develop an implementation scheme for integrating metamodels within a decision-based conceptual design framework.

When a conceptual design dataset becomes available to implement into a conceptual design tool, an implementation scheme for how to construct an accurate metamodel from the dataset would help ensure higher accuracy results. The implementation scheme would use the parameters of the dataset to draw conclusions about which metamodeling technique would be most suited to handle the properties of the given dataset.

3 METHODOLOGY

3.1 Dataset Development

Based on the development of the research issues identified, this chapter first describes the construction procedures for the conceptual design datasets. When the legacy datasets generated from a company's previous generations of detailed design is available, then the dataset development step in the metamodel generation can be bypassed and one of the methods described in Section 3.2 can be implemented. However, since legacy data is not available, alternative methods of gathering the datasets were implemented.

The choice of conceptual design datasets can range from measurements such as heat flow, engine power, tipping angle, wheel loading, stress and displacement, etc. The two conceptual design datasets chosen to evaluate the performances of the metamodeling techniques were wheel loading and stress analysis. Each of these measurements play an important role in the concept assessment of the conceptual design phase for a large vehicle design.

Due to the lack of legacy data, both the wheel loading and stress analysis datasets were simulated using a design of experiments (DOE).⁵⁵ A DOE was formulated to determine the specific parameters and values of all the design variables of the simulation. Digital prototypes were then created in CAD software to perform FEA test runs to calculate the responses of the input data. The results of these FEA simulations generated the conceptual design datasets required to begin

the analysis of the three metamodeling techniques. The following sections describe the process of constructing the wheel loading and stress analysis datasets.

3.1.1 Wheel Loading Dataset

The term wheel loading refers to the reaction force at each support point of a product. For example, the support points of a tractor are the tires while the support points of a table are the legs. Wheel loading is an important design criteria for all stages of design since it affects many physical design constraints of a product. In order to generate the wheel loading data for a generic large vehicle, a DOE was created to form a specific set of parameters to most accurately simulate the possible wheel loading combinations. These parameters included number of wheels, wheel positions, center of gravity (CG) position, and CG force.

The DOE consisted of four wheels with seven variable positions and a CG with a force of 1lb and a static center position. Each reaction force was then calculated as a percentage of the CG force, and the sum of the four reaction forces was the CG force. The entire design space using the parameters of the DOE consisted of over 2,400 FEA simulation design points. Since it was not possible to run 2,400 FEA simulations, an alternative method to simulate subsets of the design space was required. Several methods exist to generate these subsets of a design space including orthogonal (Taguchi) arrays, random sampling, and Latin hypercubes. Each of these methods provide a more uniformly distributed coverage of the design space creating concise datasets with fewer data points.

The parameters for the simulation included four design variables representing each of the four wheels. Each of the four design variables could vary between seven different wheel positions. The adequate number of simulations for the wheel loading dataset was determined to be 50 sample points. With this information, a method for sampling a subset of the design space could be implemented. Orthogonal arrays⁵⁶ were chosen initially for their easy setup. Many orthogonal arrays are available and implementing one is as simple as finding an array that fits the DOE parameters. An L-49 orthogonal array was then chosen due to the fact that it met the design criteria almost perfectly. The L-49 orthogonal array requires 49 trial runs and seven levels for each input variable while supporting between one to eight design variables.

However, orthogonal arrays were abandoned due to the inability to change the number of sample points within the dataset. By adding or removing a single sample point from the dataset, the L-49 array would lose its orthogonal properties, and thus rendering it an inappropriate way to sample the design space in a balanced manner. Therefore, orthogonal array sampling was replaced with a random sampling scheme. A random sampling scheme generates random values for each wheel position within a certain domain. Since a random sampling scheme chooses points at random throughout the design space, sample sizes of the dataset can be modified without affecting the equal distribution of the subset of the original design space.

Once the testing method was finalized, the Abaqus simulation model—the loading rig—was developed and can be seen in Figure 8. The loading rig consisted of several pieces including the base, legs, and supports. The base was located in the center of the loading rig and attached to each of the four legs. The CG force was

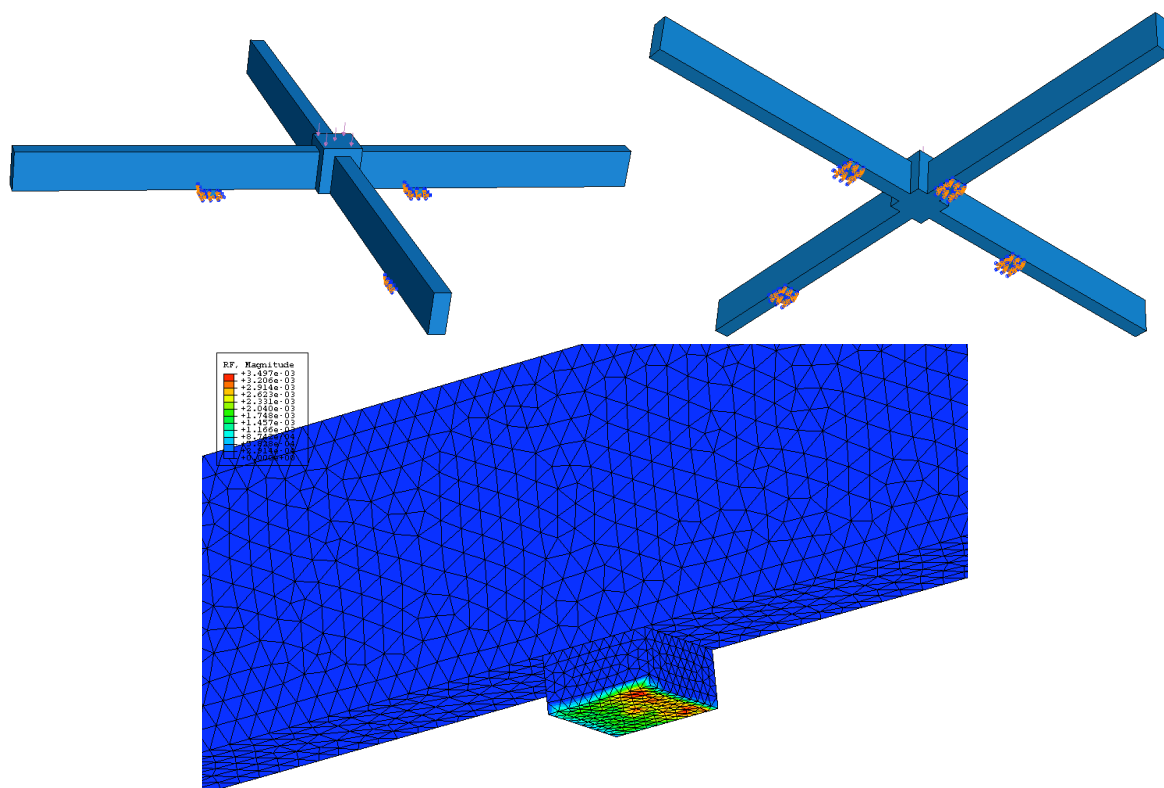


Figure 8: A top view of the loading rig for simulating a wheel loading legacy dataset (Left). A bottom view of the loading rig (Right). A close up image of the FEA simulation of the reaction forces at a wheel location (Bottom).

Upon completion of the loading rig model, the random sampling scheme was used to generate the values of the design variables for each sampling point within the design space. Two different wheel loading datasets were constructed. The first was generated with the random sampling scheme using a mean (average) of two and a standard deviation (range of the distribution above and below the mean) of 0.5. The second dataset also had a mean of two but a standard deviation of 1.0. Each dataset consisted of 80 sample points. These first 50 were used for constructing metamodels while the additional 30 were used as a cross-validation test dataset to evaluate the metamodel performance. Once the two datasets were populated, the 160 FEA simulations were run and the results recorded. The first 10 sample points for each dataset can be seen in Tables 1 and 2.

Table 1: A subset of the full wheel loading dataset with a standard deviation of 0.5.

Run No.	Input Design Variables				Output Design Variables			
	X1	X2	X3	X4	Y1	Y2	Y3	Y4
1	2.195	1.835	2.777	2.243	0.208	0.353	0.158	0.281
2	2.044	1.578	2.354	1.998	0.194	0.362	0.165	0.278
3	1.682	2.249	2.979	1.862	0.297	0.245	0.154	0.304
4	1.720	2.744	2.252	2.638	0.408	0.143	0.300	0.149
5	2.222	1.727	2.932	2.932	0.236	0.384	0.170	0.210
6	1.525	1.577	1.830	1.739	0.272	0.266	0.223	0.239
7	2.391	1.877	1.430	2.052	0.198	0.236	0.352	0.214
8	2.285	2.332	1.894	1.596	0.200	0.219	0.247	0.334
9	1.589	1.573	2.595	2.340	0.295	0.328	0.168	0.209
10	1.867	1.399	1.442	0.818	0.106	0.271	0.142	0.481
11	1.406	1.940	2.318	2.495	0.407	0.206	0.232	0.154
12	0.899	1.967	1.699	2.109	0.531	0.105	0.266	0.097
13	2.493	2.243	2.276	2.131	0.209	0.272	0.231	0.288
14	1.741	1.702	1.450	2.607	0.305	0.200	0.373	0.122
15	2.164	1.925	2.043	1.863	0.208	0.279	0.222	0.291

Table 2: A subset of the full wheel loading dataset with a standard deviation of 1.0.

Run No.	Input Design Variables				Output Design Variables			
	X1	X2	X3	X4	Y1	Y2	Y3	Y4
1	2.990	1.880	0.812	0.989	0.086	0.179	0.372	0.363
2	2.219	1.935	0.202	2.615	0.069	0.020	0.897	0.014
3	2.262	2.485	2.986	2.508	0.276	0.263	0.200	0.261
4	3.213	1.405	1.481	3.692	0.150	0.365	0.366	0.119
5	1.725	1.850	2.327	2.591	0.326	0.263	0.233	0.179
6	1.867	1.565	2.234	1.356	0.159	0.328	0.130	0.383
7	0.730	1.921	2.022	2.380	0.625	0.093	0.208	0.073
8	0.336	3.535	0.996	0.991	0.701	0.013	0.226	0.060
9	1.296	1.394	1.053	1.981	0.314	0.174	0.395	0.117
10	2.281	0.653	1.626	1.952	0.072	0.629	0.107	0.192
11	1.459	2.469	0.814	2.000	0.302	0.057	0.568	0.073
12	0.667	1.096	0.944	1.682	0.457	0.137	0.321	0.085
13	3.073	2.036	3.473	3.095	0.170	0.423	0.147	0.260
14	1.288	1.373	2.056	0.126	0.033	0.077	0.019	0.871
15	1.989	2.535	0.783	2.428	0.233	0.060	0.643	0.064

The input design variables X1, X2, X3, and X4 represent the distance between the wheel position and CG positions. The output variables Y1, Y2, Y3, and Y4 represent the reaction force for each wheel as a percentage of the CG force.

3.1.2 Stress Analysis Dataset

The second dataset was based upon FEA stress analysis simulations. The results of such tests in a conceptual design setting produce very effective assessment information to help make accurate judgements at this stage in the design process. The stress analysis dataset was based upon the stress and displacements incurred by boom trucks. These trucks (see Figure 9) have many



Figure 9: An aerial boom truck in a static position. (Left-<http://www.machinerytrader.com/listings/detail.aspx?ohid=5324527>) An aerial boom truck setup with the boom extended. (Right-<http://www.machinerytrader.com/listings/detail.aspx?ohid=5324527>)

different uses including electrical and telecommunications maintenance, tree care, and light and sign installation. In order to be best suited for these different types of jobs, boom trucks come in many different shapes and sizes. In addition to the truck ranging in length and weight, the boom can range anywhere from approximately 40 to 110 feet. The basket on the end of the boom can support loads up to 4000 pounds. To add a final parameter to the list, on most boom trucks, the boom is capable of rotating 360 degrees in all directions. All these varying parameters for these boom trucks require hundreds of various FEA simulations in order to ensure the exact specifications for a specific boom truck meet all factors of safety. These FEA simulations consume many resources of a boom truck manufacturing company. If metamodels could accurately capture the general trends of these FEA simulations of legacy data, then these metamodels could provide real-time FEA simulation data at the conceptual design phase.

Since the boom truck was chosen for the second dataset, a digital prototype of the reinforced center of the boom truck called the sub-base was created and can be seen below in Figure 10. The sub-base of the boom truck is a large welded platform constructed to handle the large moments created by the loading of the boom. The sub-base also disperses the vertical force and boom moment to the outriggers, the legs on the sides of the boom truck. Therefore, the large stresses incurred are handled completely by the sub-base and the outriggers, keeping the truck from being exposed to any structural damage, since the truck itself is not designed to handle such a load.

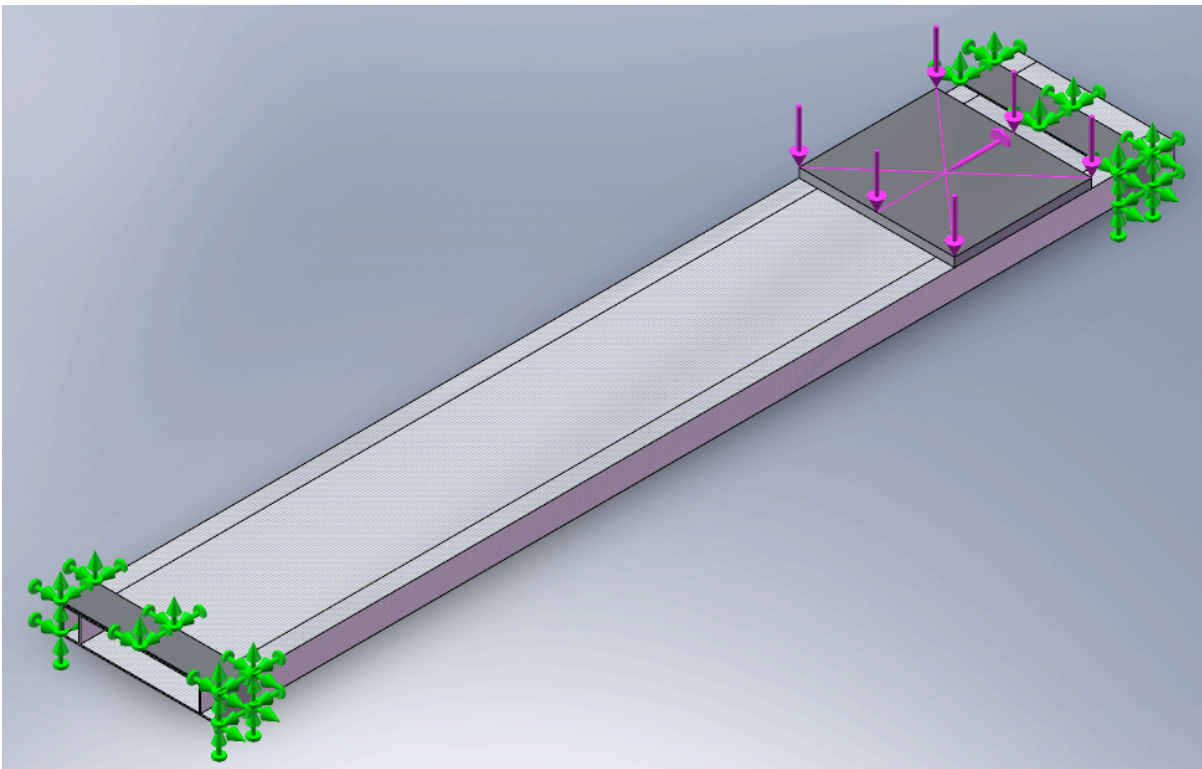


Figure 10: The sub-base of the boom truck with the FEA loading conditions attached to the outrigger plates and the pedestal.

The sub-base consisted of several different pieces including the outer beams, the inner beams, the large and small support plates, the outriggers, and the platform. The outer beams formed the outer structure of the sub-base and span the entire length of the bed of the truck. The inner beams were welded perpendicularly inside the outer beams for additional reinforcement. These inner beams form the interior sub-base support for the pedestal and boom assembly. Next, the large and small plates hold the entire structure together and help distribute the stresses throughout the entire sub-base assembly. The outrigger plates were welded directly to the outer plates where the outriggers attach to the sub-base. Finally, the pedestal is the means for attaching the boom assembly to the sub-base structure.

The varying parameters for the trial runs were then built around five different aspects of the boom truck including sub-base length, sub-base width, boom length, boom material weight, and basket weight corresponding to X1, X2, X3, X4, and X5. These five parameters formed the inputs of the dataset and were the basis for the random sampling array generated for the FEA simulations. The values for the random sampling generation can be seen below in Table 3.

Table 3: The mean and standard deviation of the design variables for the stress and displacement analysis dataset.

	Input Design Variables				
	Length (in)	Width (in)	Boom (in)	Boom (lbs)	Basket (lbs)
	X ₁	X ₂	X ₃	X ₄	X ₅
Mean	180	35	600	19000	500
Std. Dev.	24	2	175	5000	150

Since this dataset consisted of five input parameters, an additional 10 trial runs were required in order to effectively test the metamodel performance. Certain metamodels required additional data points to perform the linear regression analysis. Therefore, the random sampling array contained 90 trial runs. 60 of the trial runs were for building the metamodels while the additional 30 trial runs were created for evaluating the performance of each metamodel.

The output parameters of the FEA simulations were limited to five different parameters including maximum Von Mises stress, maximum displacement magnitude, maximum displacement position in the X-direction, maximum displacement in the Y-direction, and maximum displacement in the Z-direction corresponding to Y1, Y2, Y3, Y4, and Y5 can be seen in Figure 11. The XYZ positions of the maximum Von Mises stress were ignored for this study due to the

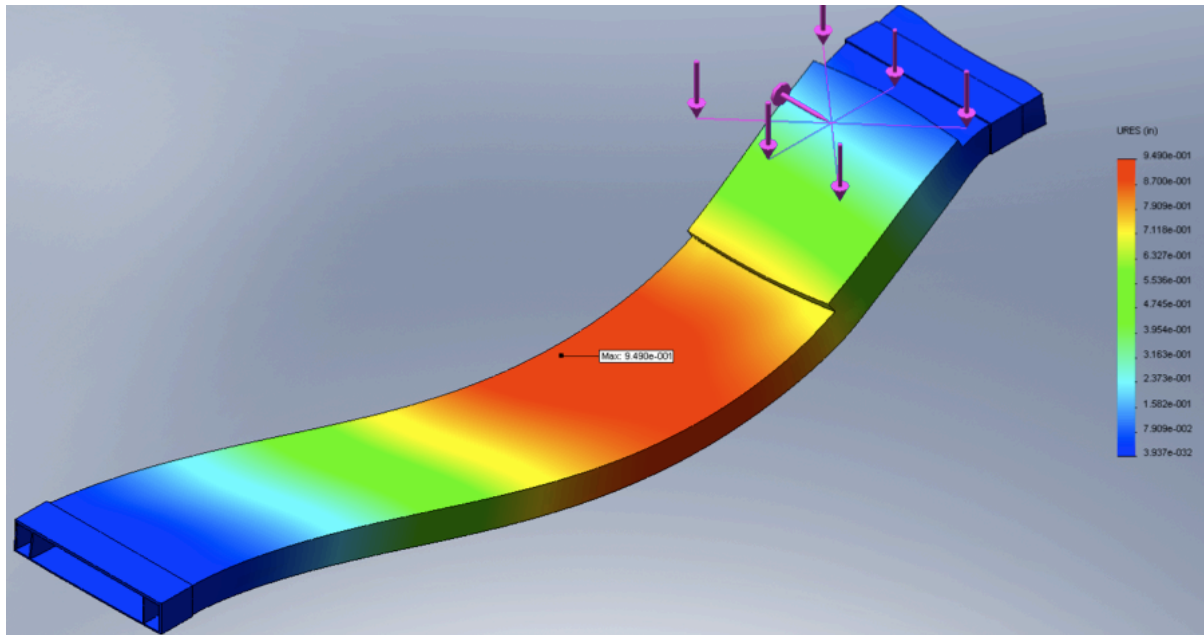


Figure 11: An image of the FEA simulation results for the sub-base displaying displacement properties.

fact that all 90 trial runs produced the same results for these positions. Therefore, the final dataset consisted of five design variables and five outputs. A small subset of the results of the FEA analysis for this dataset can be seen below in Table 4.

Table 4: A subset of the full dataset of the results of the Solidworks FEA simulations for the boom truck stress and displacement analysis.

Run No.	Input Design Variables					Output Design Variables				
	X1	X2	X3	X4	X5	Y1	Y2	Y3	Y4	Y5
1	198.2	37.2	697.9	21850	539.3	473600	0.82	18.58	6.00	69.95
2	163.4	39.7	683.7	14411	586.0	270800	0.45	19.87	6.00	57.99
3	196.3	35.5	481.1	27505	468.9	417400	0.71	17.73	6.00	71.97
4	154.3	34.5	649.9	18876	96.4	298900	0.49	17.23	6.00	56.48
5	201.6	36.4	366.7	15151	823.4	197300	0.33	18.20	6.00	73.34
6	129.0	34.0	473.3	14812	356.4	144600	0.24	17.01	6.00	53.96
7	186.8	38.7	483.9	18541	788.4	284600	0.47	19.37	5.84	63.20
8	162.4	37.2	634.7	22151	652.4	377800	0.62	18.10	6.00	57.99
9	161.4	32.5	285.9	20923	98.6	166000	0.28	16.27	6.00	53.00
10	183.6	33.7	357.2	15950	499.2	99850	0.16	16.83	6.00	66.87
11	171.9	37.7	634.6	16932	537.7	306400	0.51	18.84	6.00	60.94
12	203.3	35.8	857.1	24590	618.8	663100	1.17	17.89	6.00	74.96
13	177.4	35.8	664.1	15498	588.8	304300	0.51	17.90	6.00	61.49
14	204.3	31.6	522.5	17250	586.6	301800	0.55	31.59	3.50	79.26
15	168.6	35.5	591.5	28084	543.6	457100	0.75	17.73	5.84	59.19

3.2 Metamodel Development

After the two datasets were constructed, each of the metamodeling techniques was developed. Each technique had specific development strategies in order to provide the “best” model for that method. In order to determine the setup parameters for each of the three techniques, implementation from previous papers in each area was used. Finding the correct parameters was much more difficult for

Kriging Approximations and Radial Basis Function Neural Networks due to the vast amount of information available on Polynomial Response Surface Methodology. A Google Scholar search for each of the three techniques was done and can be seen in Table 5. This table shows the amount of research done in each of the three areas.

Table 5: Number of articles on Google Scholar for each metamodeling technique.

	Metamodel Articles		
	PRS	Kriging	RBFNN
Google Hits	3,950,000	76,700	59,700

In addition to the setup parameters for the metamodels, both dataset responses were normalized before constructing metamodels upon them. This was not as crucial for the wheel loading dataset since all the design variables were on the same order of magnitude. However, this was not the case for the stress analysis dataset as design variables having up to three orders of magnitude of difference existed. A detailed description of each of the three metamodeling development methods is described in the following sections.

3.2.1 Polynomial Response Surface Construction

When constructing PRS models, it is important to consider the general trends of the dataset before creating a metamodel to generalize the trends. For example, PRS models capable of generating highly accurate trend generalization due to excessive higher-order terms are not necessarily going to be the best approximation of the dataset. If the dataset consists of highly non-linear data points, then higher-order approximations are necessary to capture oscillations of the function. However,

if the dataset appears to have a more general response with several outlier points, a lower-order polynomial will be more apt to correctly capture the trend of the dataset. A final condition of the dataset can occur when additional noise parameters exist in the data. Again, high-order polynomials will overfit the data and provide a more specific representation of the individual data points rather than capture the general trends of the overall dataset. Often all that can be expected of a PRS model design is a steady performance in criteria important to the user.⁵⁷

The PRS models for the wheel loading and stress analysis datasets were generated in a similar manner. Ordinary least-squares regression was used to generate second and third-order approximations from the sample points of both datasets. The reason for using second and third-order approximations was to see if a higher order polynomial produced a better representation of either of the conceptual design datasets. Fitting the datasets with even higher order polynomial functions was neglected due to the fact that too many function coefficients can cause irregularity in the function producing an “overfit” of the dataset. Bucher et al.⁵⁸ used higher order polynomials without the interaction terms to create PRS models. However in this study, all the functions were produced from full polynomial expansions for both second and third-order models. Eqs. (23) and (24) are examples of a second and third-order PRS model built upon the wheel loading dataset.

$$\begin{aligned}
 y_1 = & 0.242 - 0.479x_1 + 0.292x_2 - 0.012x_3 + 0.281x_4 + 0.268x_1^2 \\
 & - 0.086x_1x_2 - 0.007x_1x_3 - 0.149x_1x_4 - 0.126x_2^2 + 0.022x_2x_3 \\
 & + 0.096x_2x_4 - 0.067x_3^2 + 0.091x_3x_4 - 0.100x_4^2
 \end{aligned} \tag{23}$$

$$\begin{aligned}
y_1 = & 0.252 - 0.549x_1 + 0.282x_2 + 0.0133x_3 + 0.288x_4 + 0.481x_1^2 \\
& -0.120x_1x_2 - 0.019x_1x_3 - 0.194x_1x_4 - 0.229x_2^2 + 0.108x_2x_3 \\
& +0.204x_2x_4 - 0.214x_3^2 + 0.194x_3x_4 - 0.185x_4^2 - 0.149x_1^3 \\
& -0.017x_1x_1x_2 + 0.079x_1x_1x_3 - 0.041x_1x_1x_4 + 0.116x_1x_2x_2 \\
& -0.313x_1x_2x_3 + 0.151x_1x_2x_4 + 0.092x_1x_3x_3 - 0.151x_1x_3x_4 \\
& +0.133x_1x_4x_4 + 0.058x_2^2 + 0.079x_2x_2x_3 - 0.113x_2x_2x_4 \\
& -0.071x_2x_3x_3 + 0.286x_2x_3x_4 - 0.235x_2x_4x_4 + 0.105x_3^3 \\
& -0.110x_3x_3x_4 - 0.009x_3x_4x_4 + 0.066x_4^3
\end{aligned} \tag{24}$$

Both PRS models shown in Eqs. (23) and (24) were built upon the wheel loading dataset with a sample size of 50 and a standard deviation of 0.5.

For the stress analysis dataset, second and third-order PRS models are given in Eqs. (25) and (26)

$$\begin{aligned}
y_1 = & -74896 - 40528x_1 - 2413x_2 + 520547x_3 + 343041x_4 + 115861x_5 \\
& +7175x_1^2 - 53790x_1x_2 + 182203x_1x_3 + 137901x_1x_4 + 132153x_1x_5 \\
& +25485x_2^2 - 38133x_2x_3 + 98306x_2x_4 - 26818x_2x_5 - 161971x_3^2 \\
& -73226x_3x_4 - 172772x_3x_5 - 27425x_4^2 - 141267x_4x_5 - 25210x_5^2
\end{aligned} \tag{25}$$

$$\begin{aligned}
y_1 = & -8031398 + 6238544x_1 + 13824641x_2 + 5979524x_3 + 10950058x_4 \\
& +2084877x_5 + 36035771x_1^2 - 7521947x_1x_2 - 31056752x_1x_3 \\
& -2885843x_1x_4 - 37901130x_1x_5 + 4679256x_2^2 - 7650587x_2x_3 \\
& -36009519x_2x_4 - 12201875x_2x_5 + 8437157x_3^2 - 3419250x_3x_4
\end{aligned} \tag{26}$$

$$\begin{aligned}
& +3077202x_3x_5 - 1671771x_4^2 + 8895627x_4x_5 + 15905511x_5^2 \\
& -9234840x_1^3 + 571171x_1x_1x_2 - 2629135x_1x_1x_3 - 27846500x_1x_1x_4 \\
& -12096297x_1x_1x_5 - 9990491x_1x_2x_2 + 18489963x_1x_2x_3 - 8551175x_1x_2x_4 \\
& +15461280x_1x_2x_5 + 7397554x_1x_3x_3 + 15236705x_1x_3x_4 + 15450204x_1x_3x_5 \\
& +6662270x_1x_4x_4 + 29227820x_1x_4x_5 + 14827519x_1x_5x_5 - 1404245x_2^3 \\
& -2447665x_2x_2x_3 + 7820834x_2x_2x_4 + 1534685x_2x_2x_5 - 4836414x_2x_3x_3 \\
& +24854008x_2x_3x_4 - 9238034x_2x_3x_5 + 13711159x_2x_4x_4 + 10186947x_2x_4x_5 \\
& +2224261x_2x_5x_5 - 4321955x_3^3 - 3886874x_3x_3x_4 - 845910x_3x_3x_5 \\
& -9369324x_3x_4x_4 - 8736211x_3x_4x_5 - 268887x_3x_5x_5 + 618079x_4^3 \\
& -8066094x_4x_4x_5 - 14204145x_4x_5x_5 - 10169637x_5^3
\end{aligned}$$

Both PRS models shown in Eqs. (25) and (26) were built upon the stress analysis dataset with a sample size of 60.

3.2.2 Kriging Approximation Construction

The method development for the Kriging Approximations was similar to the development of the PRS models. First of all the Kriging models were built upon the same datasets as PRS models, so any non-linearity characteristic considerations of the datasets for PRS models hold true with that of Kriging as well. With these dataset considerations in mind, it was determined that three different Kriging Approximations would be built for each dataset including constant, first-order, and

second-order global models. Examples of these global models for Y1 for the wheel loading dataset with a sample size of 50 are given in Eqs. (27-29)

$$f(x) = -0.108 \quad (27)$$

$$f(x) = -0.378 - 0.998x_1 + 0.486x_2 + 0.026x_3 + 0.463x_4 \quad (28)$$

$$f(x) = -0.007 - 0.903x_1 + 0.504x_2 - 0.070x_3 + 0.482x_4 + 0.187x_1^2 \quad (29)$$

$$\begin{aligned} & -0.055x_1x_2 - 0.007x_1x_3 - 0.091x_1x_4 - 0.067x_2^2 + 0.014x_2x_3 \\ & + 0.051x_2x_4 - 0.040x_3^2 + 0.053x_3x_4 - 0.051338x_4^2 \end{aligned}$$

Additionally the global models for the stress analysis dataset can be seen in Eqs. (30-32). These functions contain additional coefficients in the first and second-order global models due to the addition of a fifth design variable. These three global models were built upon y_1 of the stress analysis dataset with a sample size of 60

$$f(x) = -0.105 \quad (30)$$

$$f(x) = -0.0165 + 0.281x_1 + 0.019x_2 + 0.556x_3 + 0.570x_4 - 0.013x_5 \quad (31)$$

$$f(x) = 0.0647 + 0.279x_1 + 0.021x_2 + 0.570x_3 + 0.574x_4 - 0.025x_5 \quad (32)$$

$$\begin{aligned} & + 0.002x_1^2 - 0.020x_1x_2 + 0.071x_1x_3 + 0.050x_1x_4 + 0.046x_1x_5 \\ & + 0.011x_2^2 - 0.017x_2x_3 + 0.040x_2x_4 - 0.011x_2x_5 - 0.073x_3^2 \\ & - 0.031x_3x_4 - 0.070x_3x_5 - 0.011x_4^2 - 0.054x_4x_5 - 0.009x_5^2 \end{aligned}$$

Building an approximation for the global model was the first step of generating the Kriging Approximations. The second step used a Gaussian correlation function to account for the local departures of each data point. These departures were determined by the correlation matrix. To generate the correlation function, the θ

parameter of the correlation function must be calculated for each design variable. Initially a single value for θ for each design variable was used. However the initial trial runs proved a single θ value to be insufficient to accurately model the data. Each design variable required its own θ value. Therefore a simulated annealing algorithm was used to find the maximum likelihood estimates (MLEs) for the θ parameters for each design variable. A simulated annealing algorithm is a probabilistic method for locating a good approximation to the global optimum of a global optimization problem in a fixed amount of time. The parameters for the simulated annealing algorithm were an initial starting point of 15, a lower bound of 0.05, and an upper bound of 220. The correlation parameters for both the wheel loading and stress analysis datasets can be seen in Tables 6 and 7.

Table 6: Correlation parameters of Y1 for the wheel loading dataset with various global models.

Global Model	Correlation Parameters for Y1			
	θ_{x1}	θ_{x2}	θ_{x3}	θ_{x4}
Constant	1.99	0.29	0.05	0.05
First Order	0.15	0.06	0.06	0.07
Second Order	2.36	0.23	2.36	2.36

Table 7: Correlation parameters of Y1 for the stress analysis dataset with various global models.

Global Model	Correlation Parameters for Y1				
	θ_{x1}	θ_{x2}	θ_{x3}	θ_{x4}	θ_{x5}
Constant	0.94	5.05	0.10	0.29	0.08
First Order	0.26	0.05	1.54	0.17	220.00
Second Order	33.12	73.14	161.51	220.00	220.00

These MLEs created the most accurate representation of the local departures of the individual data points. Therefore the accuracy of the Kriging Approximations improved significantly by using the MLEs. Additionally, the three different global models help access whether a particular dataset benefits from a more non-linear representation of its data points. The combination of the MLEs for the Gaussian correlation function, the global model approximation, and the sample points fully specify the Kriging model. A new point is predicted using these values in Eqs. (9-11).

3.2.3 Radial Basis Function Neural Network Construction

To build the neural networks with Radial Basis Functions in the simplest way, a single-layer network of only three inter-connected layers were used. The three layers consisted of an input layer, a hidden layer, and an output layer. The input layer was composed of a layer of input neurons consisting of the input vector of design variables. The hidden layer was composed of radial basis neurons. The radial basis neurons received the input neuron values and calculated the Euclidean Norm between the input vector and center vector. This result was then multiplied by a scalar threshold value allowing the adjustment of each neuron's sensitivity. This product was then passed through a Radial Basis Function where a constant radius of the Gaussian function was used for each hidden neuron. The final output layer was composed of layers of linear activation functions and an error function. The output layer then minimized the root mean square error (RMSE) Eq. (33) for the model by adjusting the values for the center vectors, weights, and thresholds.

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{n}} \quad (33)$$

The process of building the network was iterative where the RMSE is minimized to reach a convergence limit called the goal. For the first iteration a single hidden neuron was placed where the largest residual of the input vector resided. After the neuron was placed, the RMSE value was minimized by shifting the parameters of the network accordingly. If the RMSE value was still larger than the goal value, another iteration of the network took place. This process continued until the RMSE value reached convergence with the goal value, or the network contained the same number of hidden neurons as value within the input vector.

An important consideration when building a RBF Neural Network is the number of hidden units and where their Radial Basis Function centers reside. Generally each data point in the training set corresponds to a RBF center. Additionally, a subset of the trial run data points could be used as the RBF centers for a less exact fit of the dataset. If the full dataset is used, then the number of degrees of freedom in the network equals the number of trial runs causing the network model to fit exactly through the dataset. This can either produce a positive or negative effect. On the positive side, if the dataset has a regular behavior, the model will represent the underlying trends of the data very well. On the other hand, if the dataset is contaminated by noise, then the network will learn all the trends of each individual data point and not capture the general trends of the dataset. This phenomenon is referred to as overfitting. The resulting function of an overfit network

does a poor job of representing the general properties of the dataset due to the rapid oscillations that occur within the network function⁵⁹.

In order to most accurately fit the dataset, the number of neurons for each model was equal to the number of trial runs within the dataset. For example, if the dataset consisted of 50 trial runs, then 50 hidden neurons were used to build the network. This ensures the model passes through every input neuron of the design space. Since it was known whether the dataset was noisy or contained outlier data points, this method was determined to be a sufficient initial performance measure for RBFNN metamodels. If however a poor performance was observed, then other methods such as less numbers of hidden neurons or curvature-driven smoothing would be implemented.

A final specified parameter of each network was the radius value for the Gaussian Radial Basis Functions. Generally a value between 0.25 and 2.0 for the radius captures the most accurate level of responsiveness for the functions. In order to determine the “best” radius value for a particular dataset, a large number of models, approximately 200, needs to be constructed upon varying radius values between 0.25 and 2.0. From the resulting analysis of each of the models, the best radius value for that dataset can be chosen through trial and error. Therefore for this study radius value of 0.5, 1.0, 1.5, and 2.0 were used.

The large number of RBF neural networks to be built for this study required a limitation of the various radius values sampled. Therefore four radius values of 0.5, 1.0, 1.5, and 2.0 were chosen to be modeled for each dataset in hopes that one of

the four values would approximately capture the models ability to accurately represent the trends of the dataset.

3.3 Process of Constructing a Metamodel

The actual step-by-step process of constructing various metamodels using different techniques was quite similar. Each metamodel involved a number of steps in order to setup, construct, and evaluate its performance. In order to better understand this process, the metamodel construction steps for a first-order Kriging Approximation for the wheel loading dataset is given.

The first step in generating any metamodel is to have a dataset consisting of both input and output parameters. The metamodel is constructed upon the input variable values to predict the output values. For the metamodels constructed in this study, metamodels were constructed upon a dataset, then a different testing dataset which contained different sample data points was used to evaluate the metamodel performance. Once the construction and testing datasets were constructed, the metamodel parameters were setup. The setup parameters for a Kriging model built upon the wheel loading dataset of 50 sample points with a standard deviation of 1.0 is given below in Table 8.

Each metamodel consists of several of inputs parameters, which must be set before constructing each metamodel. Every parameter listed in Table 8 was specified before constructing the metamodel. To better define each parameter, to begin, each simulation first required a model and test dataset. For this dataset, there

Table 8: The setup parameters for the Kriging metamodel built upon the wheel loading dataset of 50 sample points with a standard deviation of 1.0.

Setup Parameters for Kriging Metamodels	
Simulation Parameters	
Model Dataset	Model Size 50 - Std Dev 1.0.m
Test Dataset	TestData - Model Size 30 - Std Dev 0.5.m
No. Design Variables	4
Initial Theta Value	15
Theta Lower Bound	0
Theta Upper Bound	220
Global Model	First Order

were four design variables. Then the initial theta value and the upper and lower limits of the theta value were specified, followed finally by the order of the global model approximation for the Kriging Approximations.

Once the setup was completed, the simulation was run and a report was generated as seen in Figure 12. Each report generated many important pieces of information regarding the metamodel. This report gives general information about the setup parameters, then displays information regarding the dataset. Next the report displays detailed information about the actual metamodel, followed by cross-validation information and finally prediction information for the test dataset.

After the report was generated, the correlation coefficient and Yhat matrix from the report were copied into a spreadsheet as seen in Table 9 and 10. From these predicted output values, metamodel performance measures such as absolute error and root mean square error were additionally built into the spreadsheets. The results for these analysis for the Kriging approximation can be seen in Table 10.

```

SURROGATE TASK INFORMATION.
General information:
  Number of variables = 4
  Number of points    = 50
  Method              = { @srgtsKRG }

Kriging information:
  CorrelationModel     = @corrgauss
  RegressionModel      = @regpoly1
  Theta0               = [15 15 15 15 ]
  LowerBound           = [0.05      0.05      0.05      0.05 ]
  UpperBound           = [220 220 220 220 ]
=====
TRAINING SET INFORMATION.

Min of the response    = 0.019212
Max of the response    = 0.885098
Mean of the response   = 0.271848
StdDev of the response = 0.204646
=====
SURROGATE FITTING INFORMATION.

Kriging:
  Generalized least squares estimate = [-0.081327 ; -0.72973 ; 0.37192 ; ...
  Estimate of process variance       = 0.020388
  Correlation parameters (Theta)    = [0.29529  0.052973  0.37205 ...
  Correlation factors                = [0.891167  -1.30503 ...
  Scaled design sites               = [1.2742  -0.09896  -1.0841 ...
  Scaling factors for design arguments = [0.4359  0.50608  0.47807 ...
  Scaling factors for design ordinates = [0.27185 ; 0.20465 ]
  Cholesky factor of correlation matrix = [1      0      0 ...
  Decorrelated regression matrix     = [1      1.2742  -0.09896 ...
  From QR factorization: Ft = Q*G'   = [-2.6973      0      0 ...
  Number of function evaluations      = 49
  (q+2)*nv array                    = [15      16.8369  13.3635 ...
=====
CROSS-VALIDATION INFORMATION.
  PRESSRMS = 0.031201
  Cross Validation Errors = [0.013123 ; -0.066786 ; 0.038477 ; 0.019336 ; ...
=====
PREDICTION INFORMATION.
  Correlation Coefficient = 0.99684
  RMS Error               = 0.010041
  Maximum Absolute Error  = 0.026887
  Ytest                   = [0.326 ; 0.336 ; 0.124 ; 0.2548 ; 0.43114 ; 0.1416 ...
  Yhat                    = [0.32426 ; 0.34239 ; 0.099677 ; 0.25565 ; 0.4305 ...
  Error Matrix            = [0.001739 ; -0.006394 ; 0.024323 ; -0.00084783 ; ...
=====
SIMULATION OUTPUT INFORMATION.

Kriging:
  Estimated mean squared error = [2.8521e-05 ; 0.00027017 ; 0.00035098 ; ...
=====
END OF REPORT.

```

Figure 12: A sample Kriging metamodel report generated after constructing the metamodel.

Table 9: The predicted values for the output variable Y1 for the Kriging Approximation example.

Experiment #	Outputs for Y1		
	Kriging - Constant	Kriging - 1st-order	Kriging - 2nd-order
1	0.5947	0.5818	0.5885
2	0.5806	0.6215	0.6877
3	0.0761	0.0904	0.1447
4	0.4190	0.4317	0.4291
5	0.8986	0.8143	0.8184
6	0.1216	0.1616	0.1898
7	0.5265	0.5529	0.5048
8	0.8696	0.8197	0.8548
9	0.0677	0.1138	0.0377
10	0.1804	0.2499	0.2123
11	0.7553	0.7514	0.7183
12	0.4730	0.4771	0.4767
13	0.5790	0.5829	0.5953
14	0.4390	0.4737	0.5162
15	0.1438	0.1890	0.1926
16	0.4594	0.4618	0.4382
17	-0.0117	0.0305	-0.0755
18	0.5030	0.4756	0.5320
19	0.5222	0.5379	0.5383
20	0.2514	0.2432	0.2346
21	0.2617	0.2938	0.2575
22	0.0898	0.1014	0.0472
23	0.5306	0.5436	0.5629
24	0.1436	0.1679	0.2339
25	0.0943	0.1410	0.1487
26	1.0233	1.0588	1.0199
27	0.9737	0.9220	0.9395
28	0.5331	0.5367	0.5266
29	0.3892	0.4235	0.4090
30	0.4892	0.4756	0.4812

Table 10: A statistical summary of the performance of the Kriging approximation example for Y1.

	Kriging Summary Statistics for Y1		
	Kriging - Con.	Kriging - 1st	Kriging - 2nd
R	0.9951	0.9968	0.9944
R²	0.9902	0.9937	0.9889
Minimum Absolute Error	0.0022	0.0002	0.0006
Average Absolute Error	0.0321	0.0152	0.0235
Median Absolute Error	0.0223	0.0079	0.0173
Maximum Absolute Error	0.0829	0.0588	0.0802
Standard Deviation	0.0249	0.0162	0.0237
Root Mean Square Error	0.163%	0.048%	0.110%

After calculating the performance statistics for Y1, metamodels for Y2-Y4 were also constructed. Afterwards, a summary statistics table was constructed to get a better measure of the metamodel performance for all output variables. Table 14 below shows the results of averaging the performance characteristics of Y1-Y4 together. Since so many metamodels were constructed in this study, the results were combined to observe general trends. Additionally, there were too many metamodels to look at the results of each study on an individual basis.

Table 11: An average of the performance characteristics for Y1-Y4 for the Kriging approximation example.

	Kriging Summary Statistics for combined Y1-Y4		
	Kriging - Con.	Kriging - 1st	Kriging - 2nd
R	0.9704	0.9801	0.9924
R²	0.9421	0.9609	0.9849
Minimum Absolute Error	0.0008	0.0606	0.0010
Average Absolute Error	0.0467	0.1645	0.0222
Median Absolute Error	0.0344	0.1665	0.0159
Maximum Absolute Error	0.2071	0.3404	0.0804
Standard Deviation	0.0464	0.0864	0.0210
Root Mean Square Error	0.481%	9.349%	0.105%

3.4 The Advanced Systems Design Suite

In addition to developing these three different metamodels, a software package was created to implement these metamodels into real-time conceptual design tool called the Advanced Systems Design Suite^{60,61} (ASDS). The ASDS is a conceptual design tool created to quickly design 3D conceptual models, assess them with qualitative analysis and quantitative data, and visualize the results on a desktop and immersive virtual reality (VR) system. The interface enables fast geometry creation by simplifying the unnecessary inputs required by typical CAD systems. The ASDS application consists of a client interface on a tablet, laptop, or desktop PC and a VR viewer. VR Juggler^{62,63} and OpenSceneGraph⁶⁴ (OSG) serve as the VR platform and scenegraph manager for the ASDS. Both of these foundational software tools are available by open source licensing.

3.4.1 System Architecture

A brief outline of the software architecture for the developed application is illustrated in Figure 13. As shown, user interactions from the client application are transmitted over a network connection and replicated in the immersive viewer simultaneously. In addition both the client and immersive applications acquire the legacy models and data from the same data source.

Client Application—The client application operates under Windows XP and can run independently of the immersive application. In addition to OSG, the GIMP

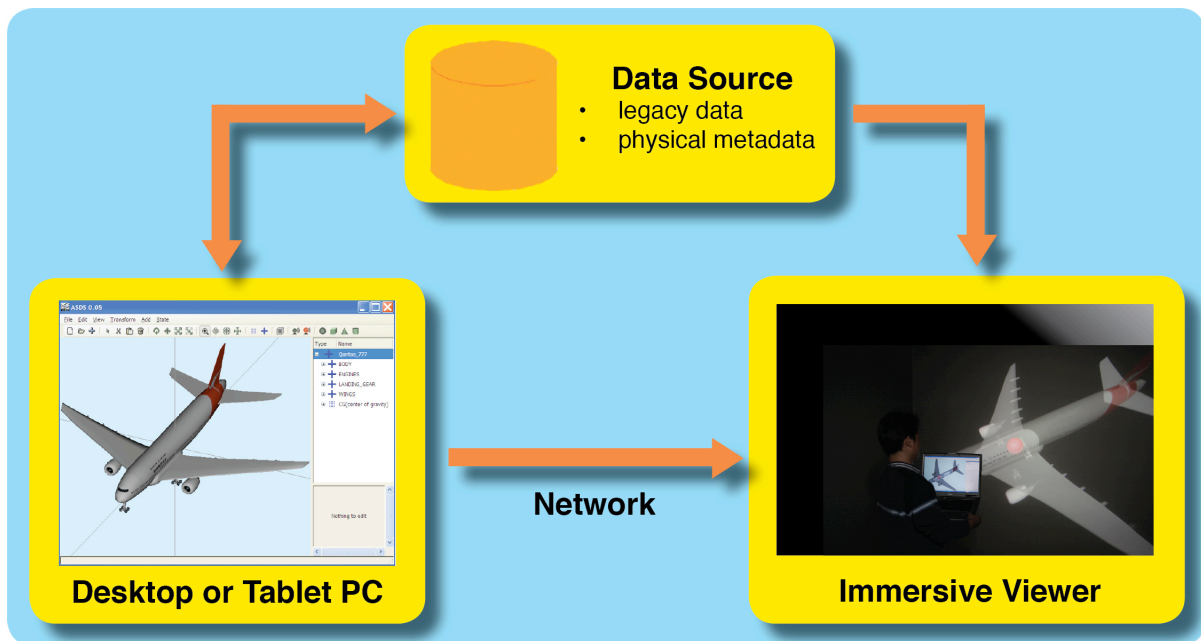


Figure 13: Schematic of the ASDS system architecture.

Toolkit⁶⁵ (GTK+) is used for graphical user interface (GUI) creation. Since both OSG and GTK+ are cross-platform development tools, the client interface could be ported to additional operating systems.

Network—Network communications between the ASDS client and immersive applications are transported using a UDP (User Datagram Protocol) socket program. UDP sockets were chosen over other communication protocols due to its speed of data transmission.

Immersive Application—The ASDS immersive application operates under a 64-bit Linux kernel calling OSG libraries and relying upon VR Juggler 2.2 for interface communication, stereoscopic viewing, and display-device abstraction. Additionally, the shared memory used in a cluster environment was redesigned to

maintain the software's performance between a single wall display or a large six-walled computer cluster.

Data Source—As a final product, it is envisioned that the ASDS part libraries will consist of legacy geometric models and physical data properties of previous generations of products as well as newly created parts from programs such as Google SketchUp. The physical data is then drawn out of the legacy data and used for assessment tool calculations and concept evaluation.

3.4.2 Interface Interactions

To begin designing concepts, engineers must first decide which existing models and legacy data will be useful. Then if necessary these models are converted to one of many different file formats the ASDS system can handle to form a modeling parts library. If legacy data does not fully represent all the necessary components to design a new concept, there are several options. First, a primitive shape can quickly be added into the scenegraph to represent the feature missing from the legacy data. If this is not sufficient, a more detailed part can be created quickly with an alternate model creation tool, such as Google SketchUp.

Once the parts library is created, any part or hierarchy of parts located in the parts library can be imported and edited inside the scenegraph by using either translation, rotation, or scaling manipulations. The hierarchy is similar to many CAD packages and can be edited on-the-fly. Objects can be grouped and ungrouped as the user chooses. This allows for a useful and sensible tree structure to be built for

each concept and additional assemblies and parts to be appended into the existing structure. By reusing existing hierarchies and enabling hierarchy manipulation, the ASDS can save the scenegraph information so a designer can pick up where they left off. These quick interaction methods enable engineers to quickly design multiple concepts on-the-fly and collaborate using the VR application to do real-time immersive conceptual design.

3.4.3 Assessment Tools

A 3D visual representation of a new concept design is extremely useful and already puts engineers ahead of the curve. However, besides a visual representation, assessment tools can give designers additional information to make educated decisions about concept integrity and viability. The ASDS has implemented metamodeling techniques to incorporate high fidelity FEA analysis into real-time assessment tools. These assessment tools can be used at any time on any concept thus providing immediate feedback and factual information to help make quick and informed decisions.

3.4.3.1 Center of Gravity and Tipping Angle

The first assessment tool dynamically computes the center of gravity (CG) of the entire model in both the client and immersive applications. Individual part weight and CG positions are stored as metadata within the scenegraph. With this information, by concatenating all the part transforms together, the CG location of any

individual part, subassembly, or entire scenegraph model can be computed. The CG location is represented by a red sphere inside the scenegraph while the scenegraph model turns transparent to view the exact CG location.

A second assessment tool computes the tipping angle of the entire model. The term tipping angle refers to the minimum angle a product can be subjected to before tipping over. To calculate the tipping angle, support points—wheels, legs, etc.—which keep the model in contact with the ground must be selected. The ASDS then uses the overall CG and contact positions of the support points to calculate the minimum tipping angle of the model and display it to the user as seen in Figure 14.

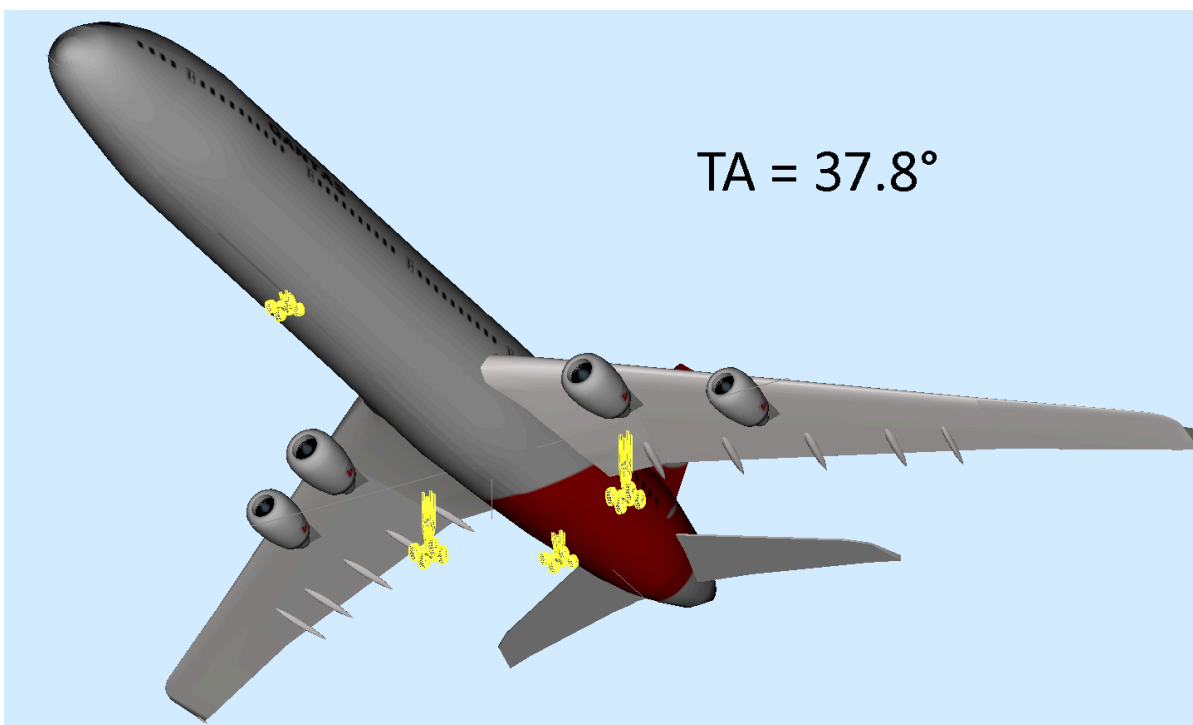


Figure 14: The tipping angle tool displays the smallest angle the product will tip based upon the selected wheels.

3.4.3.2 Virtual Measuring and Wheel Loading

Since several CAD features such as mating and collision detection have been eliminated in the ASDS interface for quicker assembly, a visual measuring tool was integrated into the ASDS client and immersive applications. When design teams collaborate within the immersive application, quick dimensional information is key to making rapid drag-and-drop part manipulations to verify whether different part library objects can be swapped to create a new dimensionally sound concept. The virtual ruler system allows the user to manipulate components and return physical characteristics from the scenegraph. Once the user selects a geometric boundary, both applications return physical characteristics of the selected boundary such as radius, length, width, etc.

Another assessment tool built into the ASDS system is wheel loading as seen in Figure 15. The term wheel loading comes from our target application to ground vehicles, but “wheel” simply refers to any support point. This tool first requires each support point be selected just like the tipping angle tool. Once completed, the ASDS system uses the contact positions of the support points and the overall CG position and weight to calculate the load distribution for each selected object. This information is then displayed graphically in both the client and immersive environments. This calculation is accomplished by either using statics equations by summing the force and moments about a point, or a using one of the three metamodel approximations to estimate the wheel load at each support point.

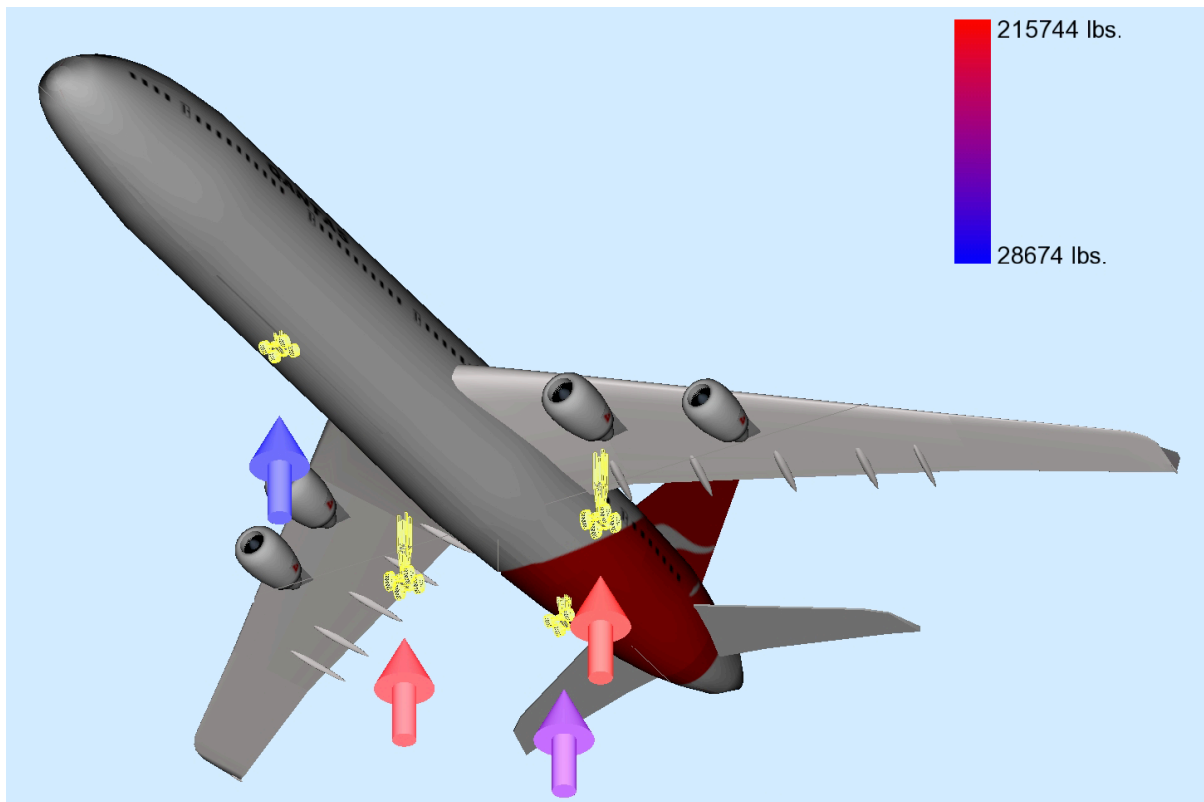


Figure 15: The wheel loading at each support position is graphically displayed to the user.

4 RESULTS

In order to obtain conclusive evidence about performances of certain metamodeling techniques for conceptual design, it is imperative to obtain enough data to observe the general trends of each metamodeling technique. If only a small amount of metamodels are built upon a very specific type of dataset, then the conclusions drawn about each technique would only apply to that specific dataset. Any general conclusions drawn in a more broad sense would be dismissed due to the small nature of the study. Several measures were taken to avoid drawing broad conclusions on a very specific dataset. First off, multiple datasets with very different characteristics and behaviors were generated from hundreds of FEA simulations.

Another measure taken was to generate as many metamodels as necessary to find conclusive evidence of specific metamodel performance. Therefore, thousands of metamodels were constructed on these datasets for each output variable of each metamodel type. For example, the Y1 output variable for the stress analysis dataset had many different metamodels constructed to predict its output. These metamodels include a second-order PRS model, a third-order PRS model, a constant Gaussian Kriging Approximation, a first-order Gaussian Kriging Approximation, a second-order Gaussian Kriging Approximation, a RBFNN - radius 0.5, a RBFNN - radius 1.0, a RBFNN - radius 1.5, and a RBFNN - radius 2.0. These models were also constructed upon datasets consisting of a model size ranging between 20 and 60 sample points. The total FEA simulations and constructed metamodels for this study can be seen below in Table 12.

Table 12: Shows the number of FEA and metamodel simulations required for the entire study.

	Total FEA Simulations				
	Abaqus		Solidworks		
FEA Simulations	160		90		
Metamodels	Total Metamodels Generated For Dataset 1				
	Size 20	Size 30	Size 40	Size 50	Size 60
PRS - 2nd-order	32	32	32	32	32
PRS - 3rd-order	32	32	32	32	32
Kriging - constant	48	48	48	48	48
Kriging - 1st-order	48	48	48	48	48
Kriging - 2nd-order	48	48	48	48	48
RBFNN - 0.5 rad.	64	64	64	64	64
RBFNN - 1.0 rad.	64	64	64	64	64
RBFNN - 1.5 rad.	64	64	64	64	64
RBFNN - 2.0 rad.	64	64	64	64	64
Metamodels	Total Metamodels Generated For Dataset 2				
	Size 20	Size 30	Size 40	Size 50	Size 60
PRS - 2nd-order	10	10	10	10	10
PRS - 3rd-order	10	10	10	10	10
Kriging - constant	15	15	15	15	15
Kriging - 1st-order	15	15	15	15	15
Kriging - 2nd-order	15	15	15	15	15
RBFNN - 0.5 rad.	20	20	20	20	20
RBFNN - 1.0 rad.	20	20	20	20	20
RBFNN - 1.5 rad.	20	20	20	20	20
RBFNN - 2.0 rad.	20	20	20	20	20
RBFNN - Best	1001	1001	1001	1001	1001
Total FEA Simulations = 250					
Total Metamodels Generated = 8070					

4.1 Overall Summary Statistics

With the vast amounts of data collected, looking at the overall performance of the three metamodeling techniques has little importance due to the large numbers of variable parameters for each method. The summary statistics for each of the three techniques can be seen below in Table 13. These statistics are the average of every metamodel built. For example, the data found under Kriging results from the average of the zero-order, first-order, and second-order performances on every dataset combined into a single average to show the overall performance.

Table 13: An overall summary of statistical measurements for each of the three metamodeling techniques.

Statistical Properties	Summary Statistics		
	PRS	Kriging	RBFINN
R	0.810	0.869	0.731
R²	0.739	0.777	0.626
Minimum Absolute Error	0.004	0.003	0.012
Average Absolute Error	0.165	0.084	0.324
Maximum Absolute Error	0.101	0.055	0.201
Absolute Error Standard Deviation	0.918	0.367	1.959
Root Mean Square Error	19.90%	9.22%	40.24%

This table alone shows Kriging Approximations to be a significant winner over the other two methods. However, to conclude from this table alone that Kriging is simply the best metamodel for conceptual design out of these three methods is simply untrue. Many additional considerations must be taken to adequately determine which metamodel performs the best under specific circumstances. In order to sift through the different levels of performance between the metamodels, the

results have been divided to look at specific circumstances to better understand exactly why certain types of metamodels perform better under specific cases.

First off, the performance of each metamodel is compared based upon sample size of both the first and second datasets each metamodel was constructed upon. Next, RBFNN parameters are redefined in the second dataset in order to find the ideal radius value before constructing the model. This helps to better understand the performance enhancements that are possible when ideal radius values for the Radial Basis Functions are determined prior to constructing the model itself.

The next section is based strictly on the first dataset and evaluates each models performance based upon the idea of interpolation and extrapolation. The two different ranged datasets having standard deviations of either 0.5 or 1.0 are compared against one another. The basis for these analysis was to determine which metamodel was best apt to handle extrapolating a small range of values outward across a much larger design space. Analysis was also done to determine which metamodel was most apt to interpolate data from a much larger dataset into a more concise region of the design space. These analysis breakdowns make it easier to draw accurate conclusions about each models performance.

Another important consideration was the normalization of the output variable design space. This was accomplished by using each output variable's minimum and maximum values and then normalizing these values between zero and one. Otherwise the RMSE values were up to five orders of magnitude different before the normalization. Therefore combining the results of the output variables to measure

performance characteristics would have been unreasonable since several output variables would have been insignificant when averaged with others.

The results of each specific case are presented as RMSE. This has been the dominate performance metric used throughout the cited work. Also zero conditioning of the dataset takes place before construction of the metamodels, so outlier data points can still exist in the dataset. These outliers can cause very large absolute errors on an individual basis. Even if the metamodel overall provides a good fit to the dataset, a single outlier data point can cause a large absolute error. If absolute error was used as the performance metric in this case, then the metamodel would appear to be a poor fit of the data, when in reality, the metamodel was actually a good representation. RMSE was used to provide a better understanding of not only the absolute error for the model, but also the models ability overall to fit the dataset.

4.2 Sample Size Analysis

Varying the sample size was analyzed due to the fact that these metamodels were being constructed for conceptual design based upon legacy data. Generally companies have massive amounts of legacy data retained from previous designs. Each new design undergoes hundreds a data simulations before being produced. Therefore, the goal was to determine how many sample points are necessary for each metamodeling type to adequately model the design space. The first dataset consisted of five different sample sizes ranging between 20 and 50 in increments of 10 while the second contained an additional 10 sample points.

4.2.1 PRS Results - Sample Size - Dataset 1

The results of the PRS performance for Dataset 1 can be seen below in Figure 16. The second-order PRS model performs slightly better with model sizes under 40 sample points. However, once the dataset consists of 40 or more sample points, a third-order PRS model fits the dataset more accurately. With a model size of 50, the third-order polynomial has almost 1.0% less error than the second-order PRS model. In summary for Dataset 1, second-order PRS models should be built up to 40 sample points. The extra coefficients of a third-order PRS model only show benefit with datasets containing 40 or more sample points.

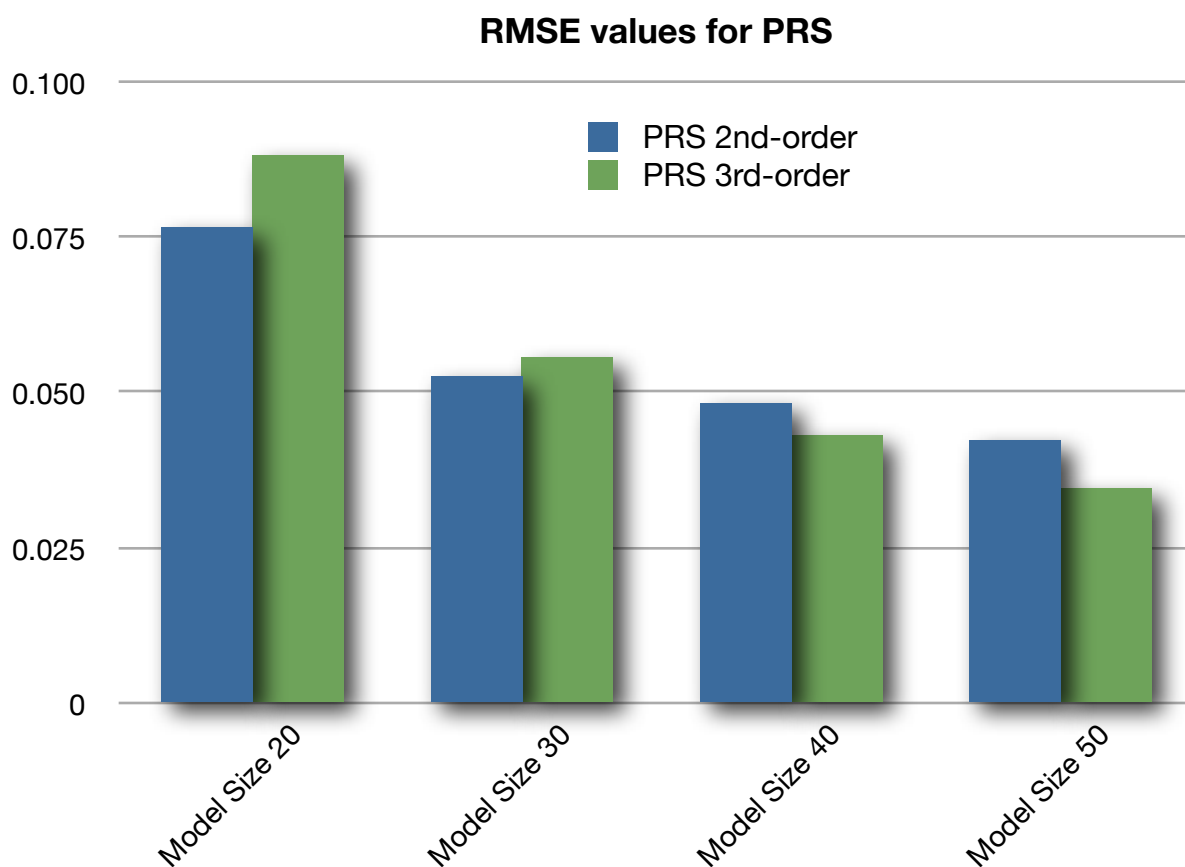


Figure 16: A chart displaying the RMSE values for PRS based upon sample size.

4.2.2 Kriging Results - Sample Size - Dataset 1

The results of the Kriging Approximations performance for Dataset 1 can be seen below in Figure 17. To begin the constant global model seems to be significantly outperformed with every model size. First and second-order Kriging Approximations show similar results for almost every model size except 50. Here the second-order model shows a significant improvement in performance. To summarize Kriging Approximations for Dataset 1, first-order approximations provide high accuracy until a model size of 50 where a second-order Kriging Approximation should be implemented due to higher accuracy.

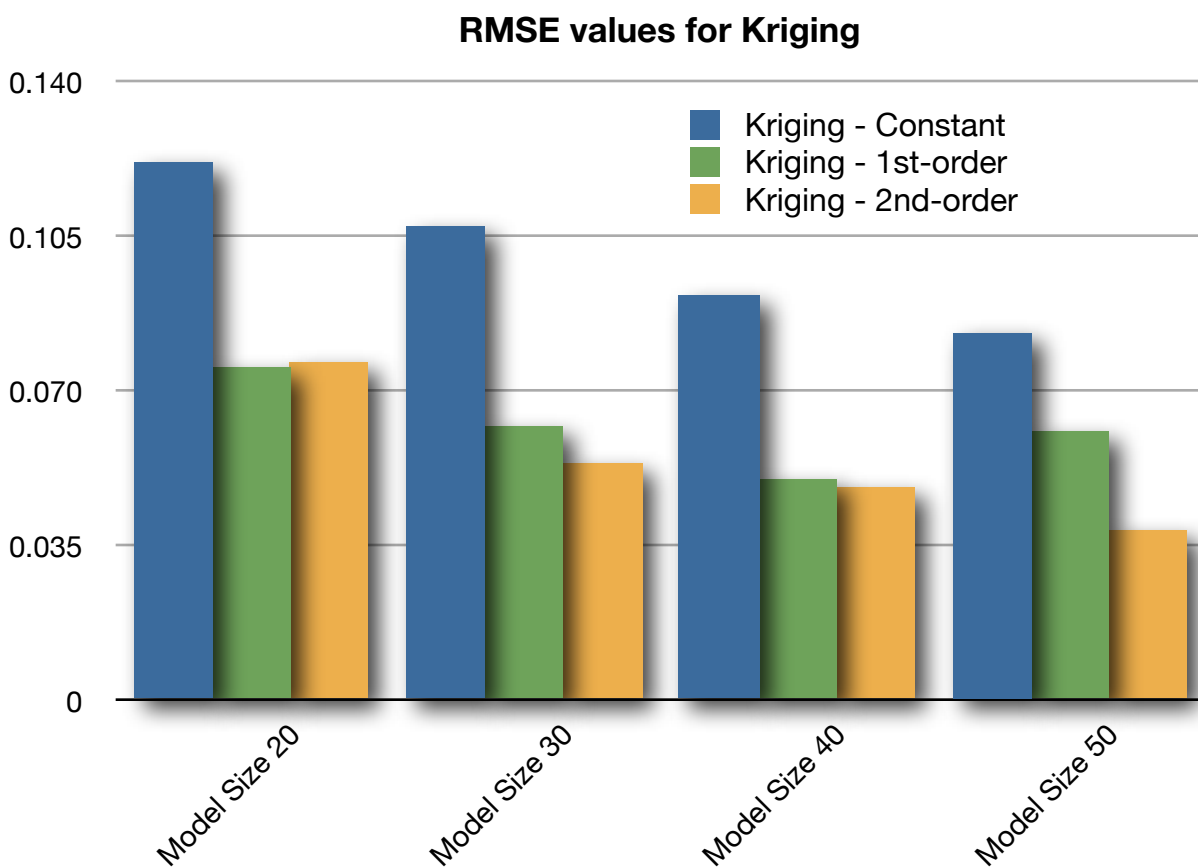


Figure 17: A chart displaying the RMSE values for Kriging based upon sample size.

4.2.3 RBFNN Results - Sample Size - Dataset 1

The results of the RBFNN models for Dataset 1 can be seen below in Figure 18. The RMSE values are completely scattered for each model size. There is no possible way to determine a best radius value based upon the four sampled radius values. Each model size has a different ideal radius value. In order to determine the performance of RBFNNs, ideal radius values must first be determined. Then the models should be constructed upon those ideal radius values. In summary, ideal radius values should be determined before constructing a RBFNN metamodel.

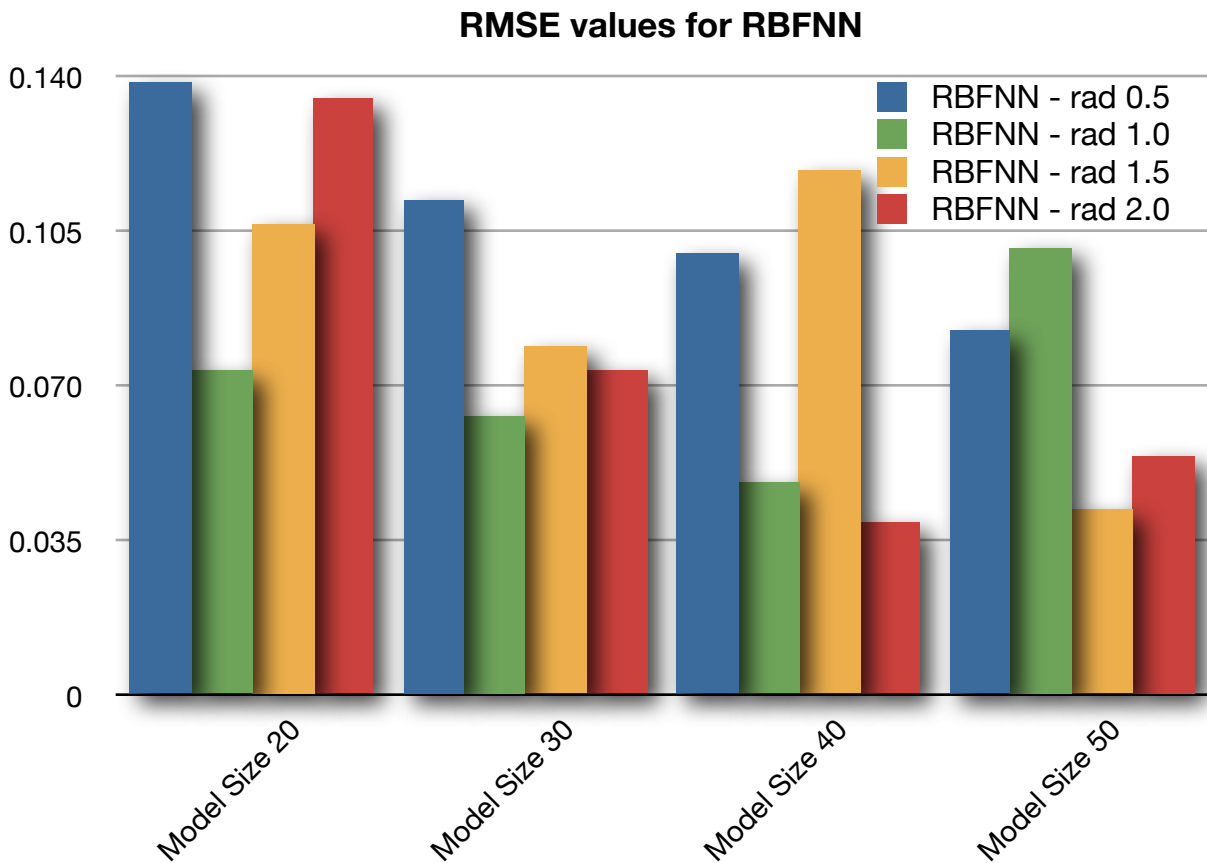


Figure 18: A chart displaying the RMSE values for RBFNN based upon sample size for Dataset 1.

4.2.4 Metamodel Comparisons - Sample Size - Dataset 1

Figure 19 below shows the best performance result for each metamodeling technique. The best performance simply refers to each type of metamodeling technique. For example, for a model size of 20, the second-order PRS model RMSE value was used since it was better than the third-order value. So these values represent the best possible solution for each of the three metamodeling techniques with the data sampled.

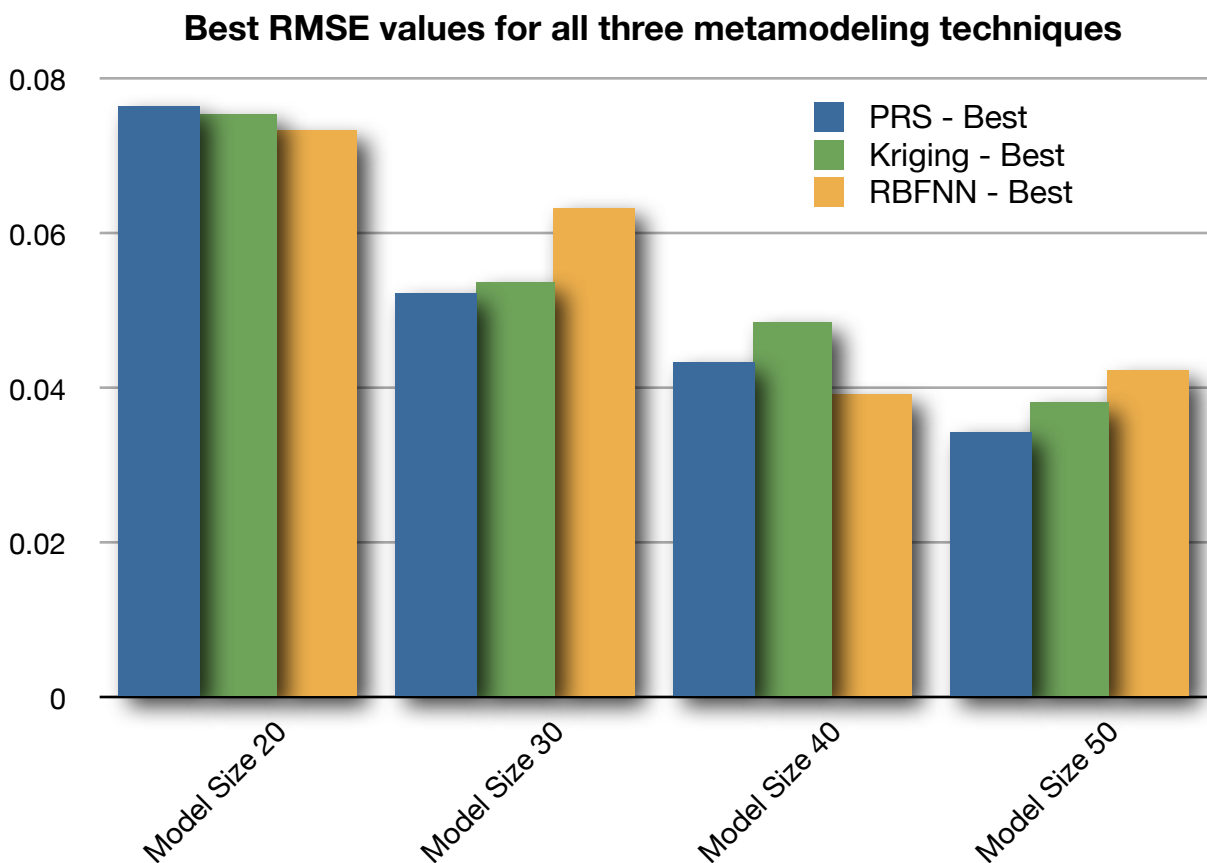


Figure 19: A chart displaying the RMSE values for each metamodeling technique based upon sample size for Dataset 1.

From Figure 19, it is difficult to draw conclusions about which metamodeling technique produced the best results for several reasons. First of all each result is within approximately 1.0% of the other RMSE values. So to draw any conclusions specifying an obvious winner is difficult with such small differences between them. With that said, PRS models showed the best performance in model sizes of 30 and 50 while RBFNN performed the worst. However, RBFNN outperformed the other two techniques with model sizes of 20 and 40. Kriging did not outperform either method with any model size which is quite surprising. Kriging uses a correlation function in addition to a global polynomial model to help pull the function towards outlier data points. Basically it combines the best parts of PRS and RBFNN into a single method. Yet it performs the worst on the first dataset.

These observations support a fairly linear dataset. In order for RBFNN model to accurately fit the test dataset with the current implementation of using the sample data points as centers for the hidden neurons, the dataset must be fairly linear to observe low RMSE values and a high R^2 value. Additionally, since PRS models tend to outperform Kriging models, this could possibly be due to the Kriging correlation function pulling the fit too close to each outlier data point causing additional oscillations in the fit of the design space. This overfitting the dataset with the correlation function does not occur in PRS. This could be a viable explanation for the results of Figure 19. On a final note, RBFNNs can also perform significantly better yet by finding the ideal radius values before constructing the metamodels. This idea was implemented in the following section.

4.2.5 Metamodel Comparisons - Sample Size - Dataset 2

Figure 20 below shows the best performance result for each metamodeling technique. Dataset 2 produces significantly different results than Dataset 1. The benefit of Kriging Approximations can easily be seen in the evaluation of this dataset. First-order Kriging Approximations outperform every other type of metamodel built upon this dataset in every model size. Kriging outperforms PRS by approximately 4.0% and RBFNN by approximately 10.5%. These differences are far greater than in Dataset 1.

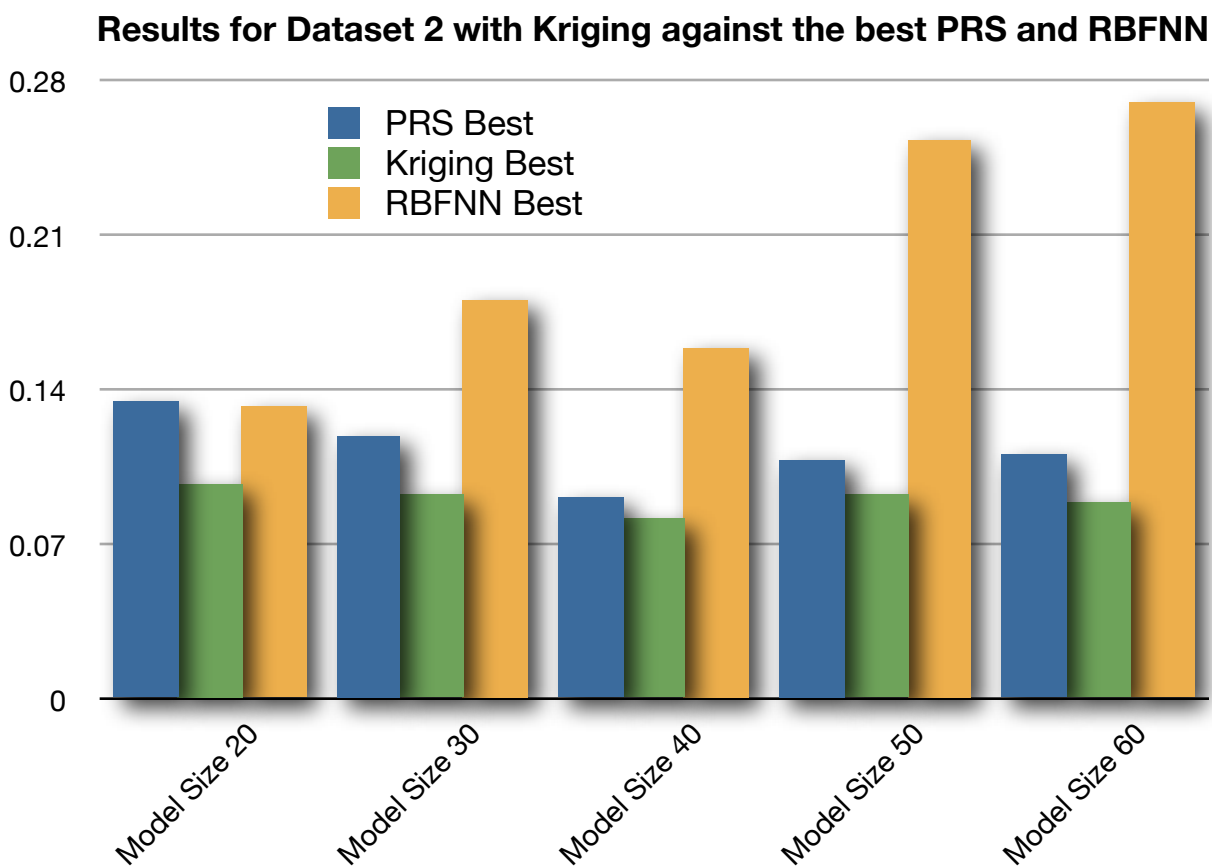


Figure 20: A chart displaying the RMSE values for each metamodeling technique based upon sample size for Dataset 2.

From these results it is clear Dataset 2 contains highly non-linear data points. The construction of the dataset parameters prior to the FEA simulations also support non-linear behavior. To explain exactly what is happening here with these results, first off, PRS does a descent job of fitting this dataset for conceptual design with roughly a 10% RMSE value. Kriging then uses a similar polynomial expression to generalize the dataset, then implements the correlation function to pull the final Kriging model closer to the outlier data points encapsulating more of the dataset within the model. This additional piece of Kriging can be directly observed in Figure 20. As for RBFNN, it is apparent that it is having a difficult time fitting the non-linear data points. The issue lies within the construction parameters of each RBFNN model. By fitting the input dataset almost exactly, the model is overfitting Dataset 2. Therefore even the more linear points within the test dataset are still producing large absolute error values due to the exact fitting of the outlier data points in the sample dataset.

In order to produce better results for RBFNN models, two methods can be implemented. The first method involves calculating the ideal radius value for the dataset prior to constructing the metamodel. This method is implemented in the following section. A second method would be to implement a curve smoothing function into the development of the RBFNN model after determining the ideal radius value. RBFNN models fit the construction dataset exactly, yet on a non-linear dataset cannot produce the same kind of accuracy values observed in Kriging Approximations.

4.2.6 Metamodel Comparison with Ideal Values for RBFNN

Figure 21 below shows almost the same information as Figure 20 with one exception, the RBFNN Best models have been constructed using the ideal radius value for each model size. To determine the ideal radius value, 1005 metamodels were built for each of the five model sizes. Simulations consisting of 201 radius values ranging between 0.25 and 2.0 with a constant step size were generated for each output variable. The lowest RMSE metamodel then became the new RBFNN Best containing the ideal radius value for that output variable at that model size.

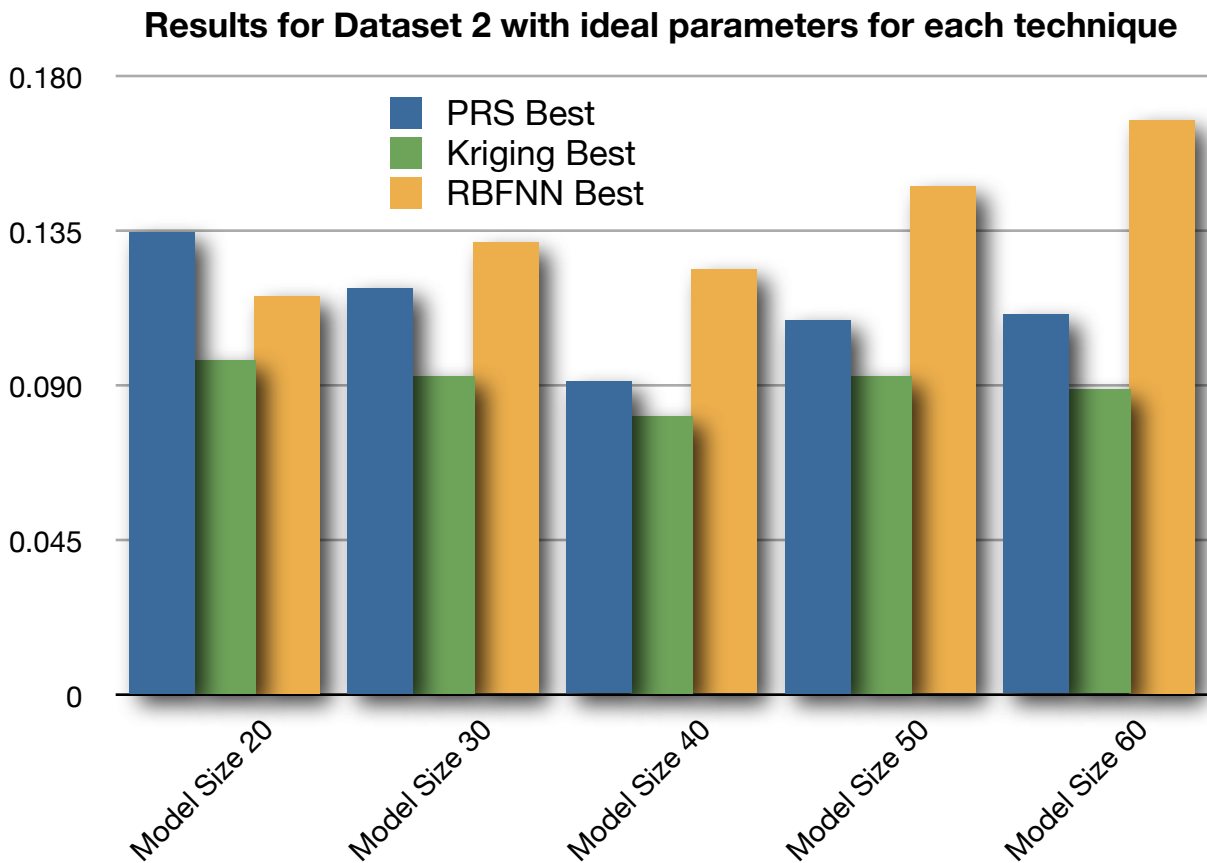


Figure 21: A chart displaying the RMSE values for each metamodeling technique based upon sample size for Dataset 2 with the ideal radius values used for RBFNN.

The RMSE difference now between Kriging and RBFNN has dropped from 10.5% to approximately 4.6%, almost a 6% decrease in RMSE due to building the RBFNN model with the ideal radius. The RBFNN models however still do not perform as well as the PRS metamodels. PRS models have a RMSE of approximately 2.2%.

To summarize the results of finding the ideal radius value for to find RBFNN Best, the time required to find the ideal radius value can only be justified when the dataset has a fairly linear behavior. Over five thousand metamodels had to be generated in order to produce the data to generate Figure 21. This still took hours to calculate with only five design variables and five output variables. If the dataset consisted of hundreds or even thousands of design variables, constructing RBFNN metamodels would be out of the question. Not to mention they still would not fit the data as well as a PRS metamodel or a Kriging Approximation.

4.3 Extrapolation vs. Interpolation Results

Conceptual design teams can have a number of different tasks ranging from designing a completely new product from scratch, designing another generation of a product already in production, or even sometimes taking several completely different products and combining them to harness new product capabilities into a single product. The question at hand is how well can metamodels handle these different types of conceptual design. If a metamodel is constructed upon legacy data in a very concise design space, is it capable of providing high enough accuracy at the

conceptual design phase to provide designers with useful information for concepts with parameters extremely outside the design space the metamodel was built on. This is referred to as extrapolation.

On the contrary, if a metamodel was constructed on a dataset comprised of a very wide range of sample points throughout a design space, can the metamodel accurately predict concept designs with parameters localized within a small region of the design space? This is referred to as interpolation. Throughout this section each metamodeling technique is tested on Dataset 1 to determine each technique's performance in terms of both extrapolation and interpolation. Additionally, the smaller and larger ranged datasets are evaluated against test datasets with the same range to test each metamodels ability to predict values within the same localized design space.

In order to test these cases, Dataset 1 was sampled with a standard deviation of both 0.5 and 1.0. Metamodels were constructed upon both ranges of datasets, and then tested against a test dataset of either the same range or the larger or smaller range. For example, PRS models were constructed upon the smaller range dataset—a standard deviation of 0.5—and then tested against both the smaller and large range test dataset. This then provides test results for a localized metamodel test against both a localized and interpolated dataset. Additionally, PRS models were constructed on the larger range dataset—a standard deviation of 1.0—and then tested against a test dataset of both smaller and larger ranges. Thus providing test results for a localized larger design space in addition to the results for interpolation of the large range model performance on a localized test dataset.

4.3.1 PRS Results - Standard Deviation - Dataset 1

Figure 22 shows the results of the PRS model performance. For interpolation, the second-order PRS model better captured the behavior for a localized test dataset. However, for extrapolation, the extra coefficients for the third-order PRS model created a more precise fit of the larger ranged test data. For the small ranged models tested against small range data, third-order PRS models perform better. However, for large ranged models tested against large range data, the opposite trend occurred where less coefficients did a better job of capturing the general trend of the data.

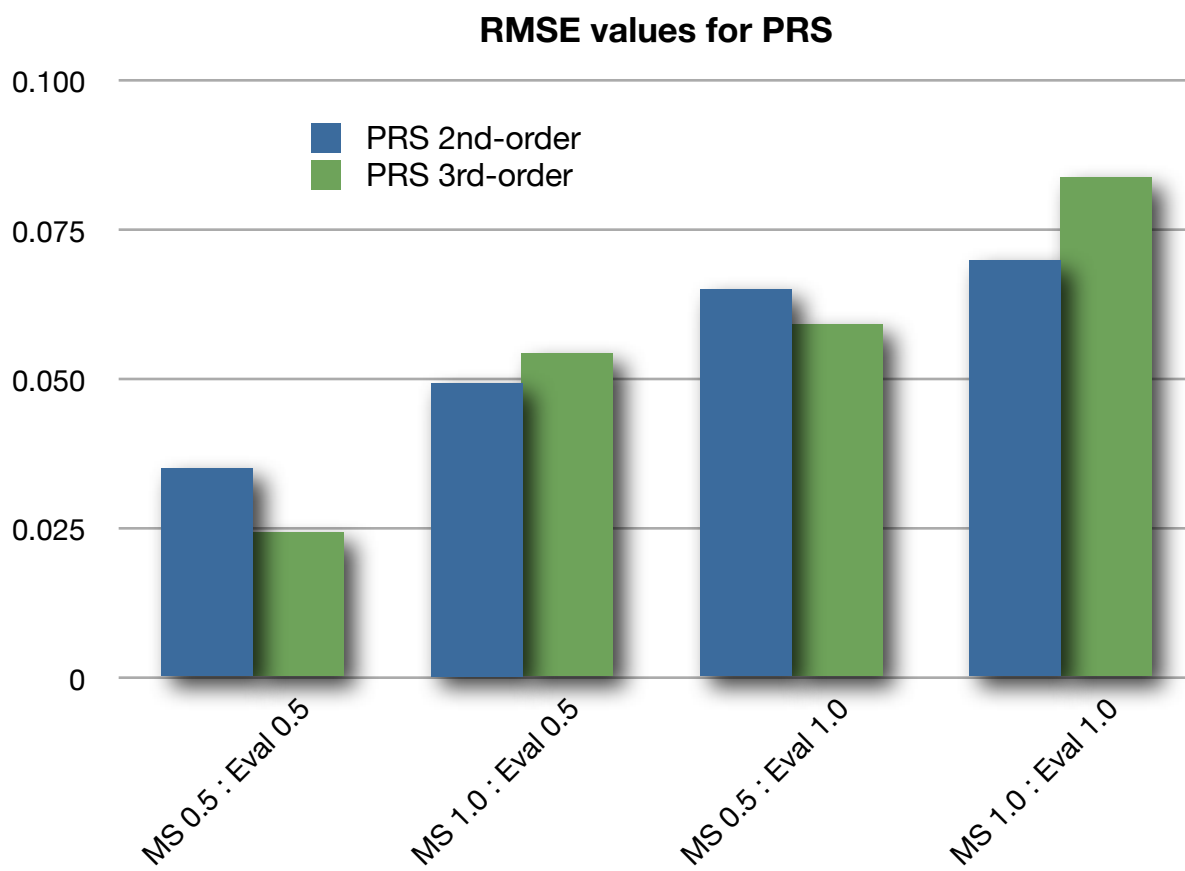


Figure 22: A chart displaying RMSE values for PRS based upon standard deviation.

4.3.2 Kriging Results - Standard Deviation - Dataset 1

Figure 23 shows the results of the Kriging model performance. For interpolation and extrapolation, the second-order Kriging model better captured the behavior for both the small and large range data. For the small ranged models tested against small range data, second-order models also more accurately predict the localized design space more accurately. However, for the large ranged models tested against large range data, the first-order model best fit the test data. PRS and Kriging perform better with less coefficients in a larger less concise design space.

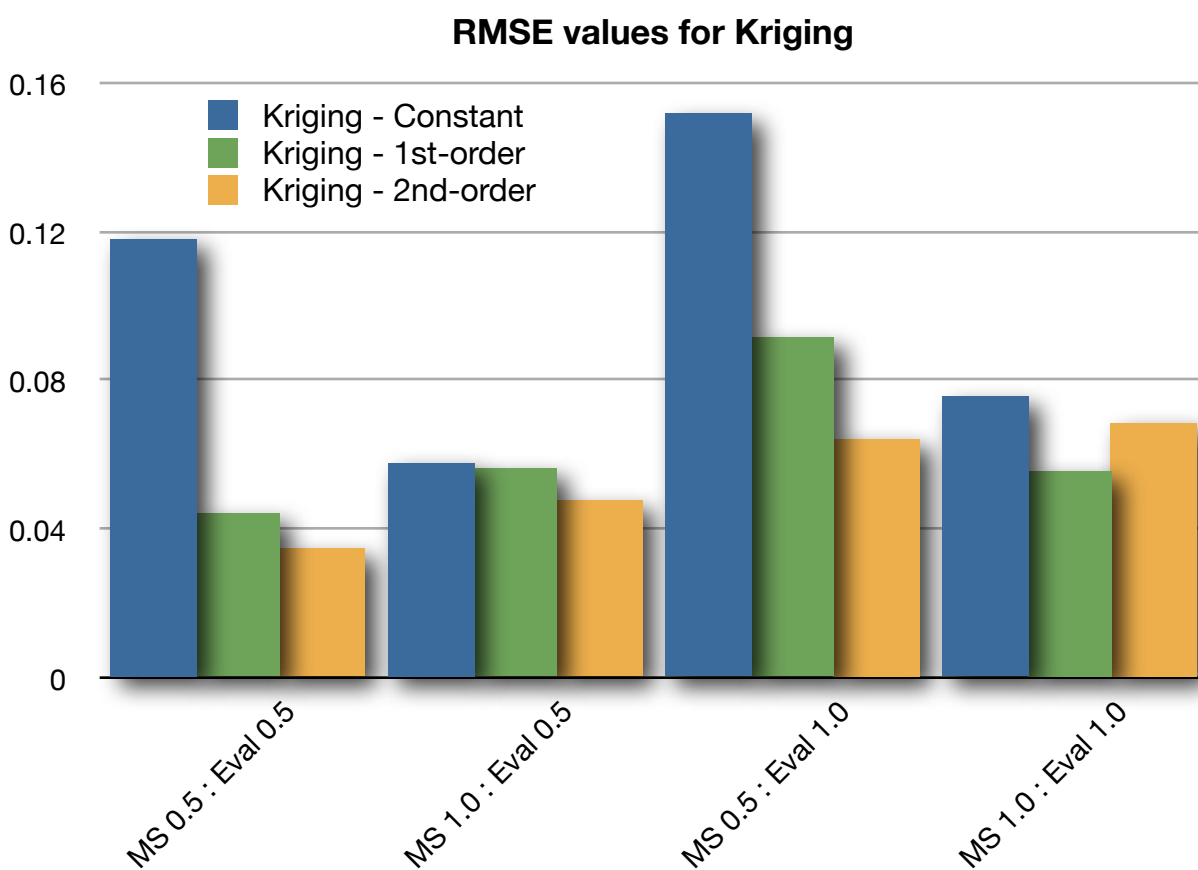


Figure 23: A chart displaying the RMSE values for Kriging based upon standard deviation.

4.3.3 RBFNN Results - Standard Deviation - Dataset 1

The results for RBFNN performances are shown below in Figure 24. These are not ideal radius values so the performance could greatly increase by finding these ideal radius values prior to constructing the models. RBFNN seems to work well with the first and second columns where the model was constructed to predict small localized areas. When extrapolating or predicting a large area of the design space, RBFNN breaks down. This shows the inability of RBFNN to handle more non-linear approximations due to overfitting the sample dataset.

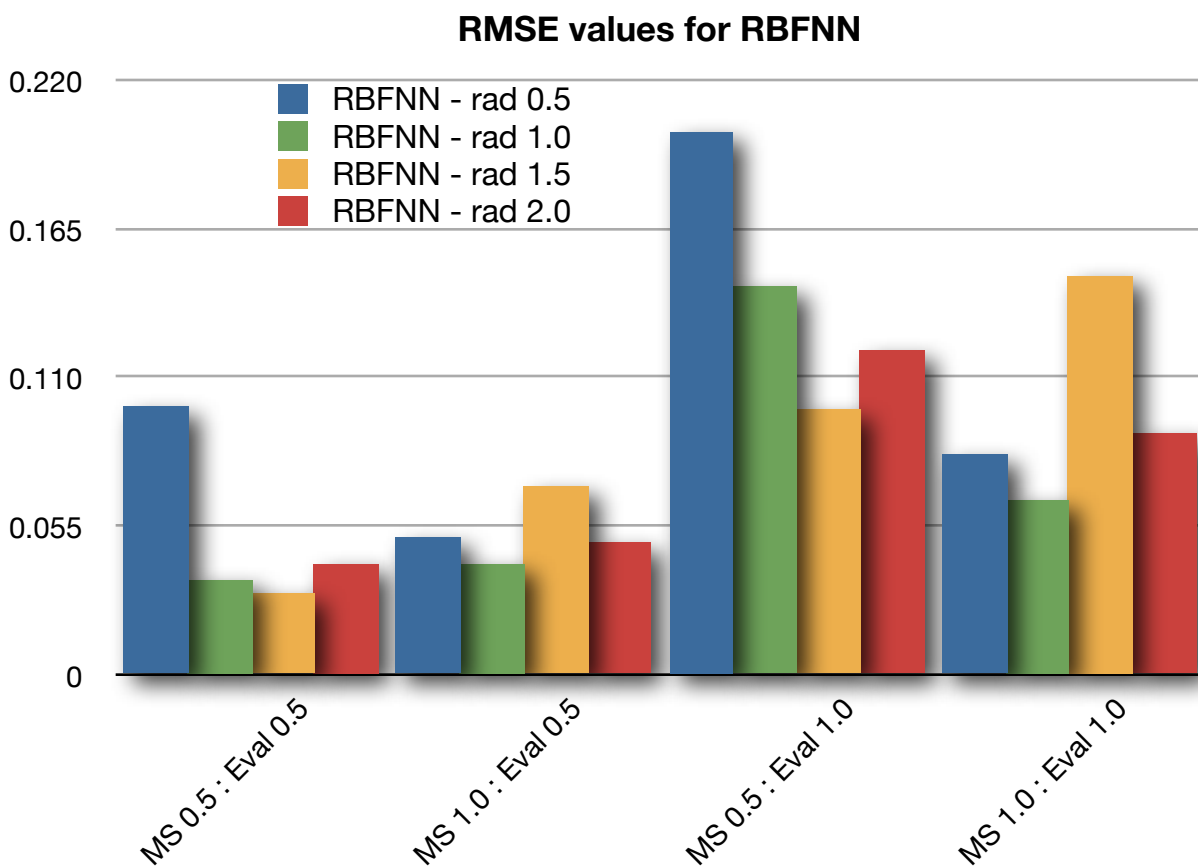


Figure 24: A chart displaying the RMSE values for RBFNN based upon standard deviation.

4.3.4 Metamodel Comparisons Based Upon Standard Deviation

The comparisons of each metamodeling techniques performance can be seen below in Figure 25. For interpolation, RBFNN models exhibit the best performance for interpolation and the worst for extrapolation. PRS showed the best performance in extrapolation and also in the first column with localized data due to the overfit caused by the correlation function for Kriging and the exact fit to the sample data for RBFNN. For the larger range data in the forth column, Kriging models outperform the other two techniques.

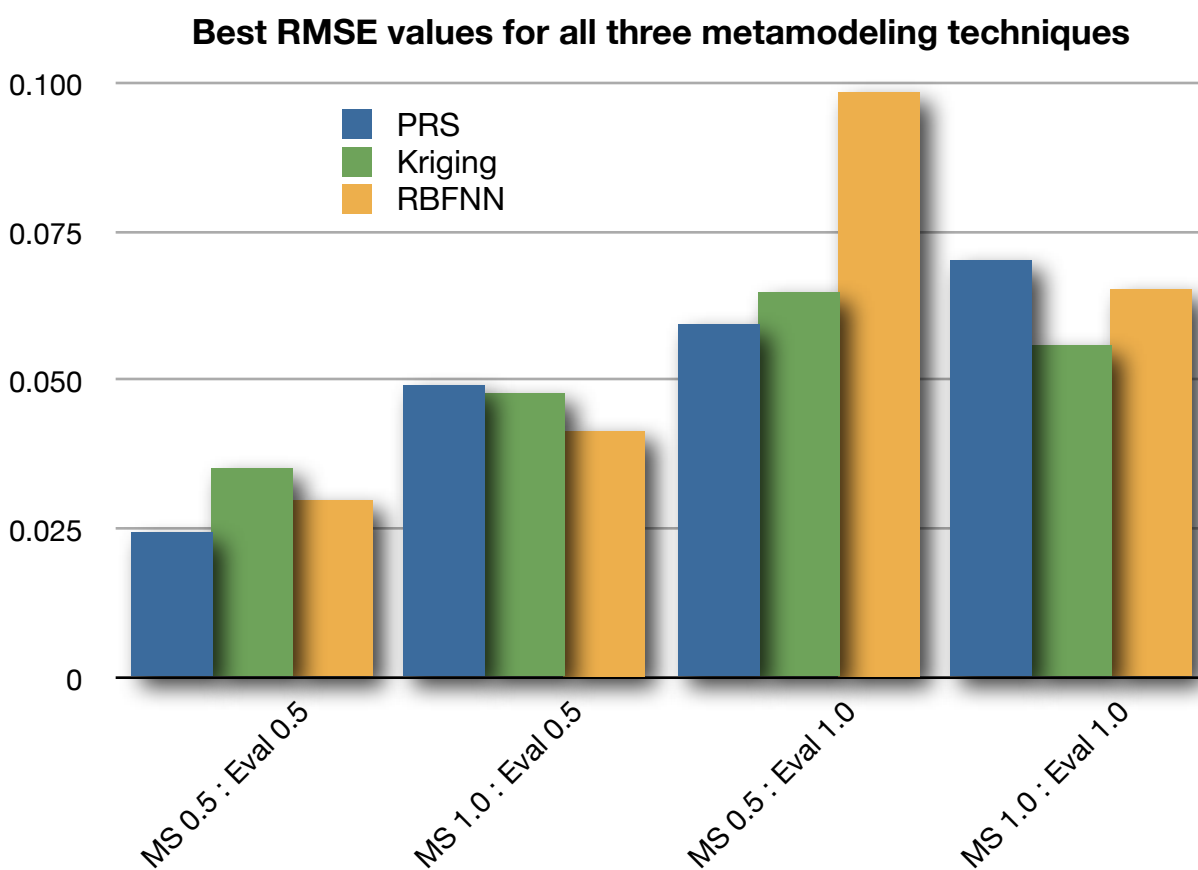


Figure 25: A chart displaying the RMSE values for each metamodeling technique based upon standard deviation.

The results for testing extrapolation and interpolation throughout this section follow the general performance trends observed throughout the entire results section. For the first column results where a smaller ranged model was tested against smaller range test data, the results show that higher order polynomials are better suited for small concise datasets. However, the correlation function for Kriging appears to hinder the method's performance compared to PRS. Additionally, RBFNN could possibly be a better solution here if time was taken to generate the ideal radius values before generating the metamodels.

For interpolation, RBFNN wins hands down, even with non-ideal radius values. The additional oscillations for both Kriging and RBFNN allow both methods to capture more of design space in localized regions. Then when predicting values in a localized area of the design space with these models, they are better suited due to the additional curvature of the models. If interpolation is the main focus of constructing an accurate metamodel, then RBFNN appears to be the best choice.

For extrapolation the exact opposite is true. The more accurate fitting of the localized sample data for RBFNN causes it to breakdown significantly when extrapolated out to larger areas of the design space. Third-order PRS models performed the best when testing for extrapolation capabilities with second-order Kriging models not far behind.

When large areas of the design space are sampled for both the metamodel and test dataset in the fourth column, Kriging Approximations appear to be the most robust solution. Although with RBFNN not far behind, it is difficult to conclude whether Kriging is the absolute best solution since non-ideal radius values were

used. Possibly with ideal radius values, RBFNN performance could match or exceed that of Kriging Approximations.

5 CONCLUSIONS AND FUTURE WORK

Rapid assessment capabilities need to be implemented into conceptual design as an alternative to high fidelity analysis. Building metamodels upon legacy data of previous high fidelity simulations has the potential reduce or possibly eliminate the need for high fidelity analysis in conceptual design, yet provide highly accurate information in real-time to engineers. With this structure in place, engineers could quickly evaluate a multitude of concepts in real-time with factual hands on information about each concept. Three different metamodeling techniques, Polynomial Response Surfaces (PRS), Kriging Approximations, and Radial Basis Function Neural Networks (RBFNN), were evaluated to determine which of the three best fit the general trends of several conceptual design datasets.

5.1 Polynomial Response Surface Conclusions

When to use PRS metamodels

1. Sample sizes greater than 40 data points for linear datasets
2. Extrapolation of concise datasets to a larger area of the design space
3. Linear datasets consisting of a small concise design space

When NOT to use PRS metamodels

1. Interpolation of large datasets in small regions of the design space
2. Any highly non-linear dataset
3. Datasets with a very large design space exhibiting non-linear behavior

Several other important notes regarding the implementation of PRS are to generally always use third-order polynomials to fit the data. In almost every situation, third-order polynomials outperformed second-order polynomials with the exclusion of highly non-linear datasets. Also, building a PRS metamodel requires very little effort and can be done very quickly. If speed is the key consideration for metamodel construction, then PRS is the best option. The only specification in the construction is the order of the polynomial which should almost always be third-order unless it is a highly non-linear dataset. In which case a PRS metamodel is a poor option.

5.2 Kriging Approximation Conclusions

When to use Kriging metamodels

1. Any non-linear dataset of any sample size
2. Large design space datasets exhibiting linear or non-linear behavior

When NOT to use Kriging metamodels

1. For cases where interpolation or extrapolation are required
2. Linear datasets with sample sizes larger than 40 data points

Kriging Approximations are a very robust metamodeling technique with very high performance using a first-order global model on non-linear datasets. In regards to more linear datasets, Kriging should be avoided. However, if Kriging must be used, second-order global models perform much better on linear datasets than first-order models. Always avoid using a constant global model for generating Kriging Approximations for the performance of these metamodels was significantly worse in

every single test case. On the downside, Kriging metamodels require slightly more time to setup and construct than PRS metamodels, but overall, the extra preparation results in a much more robust metamodel which can handle a larger number of different types of datasets.

5.3 Radial Basis Function Neural Network Conclusions

When to use RBFNN metamodels

1. Linear datasets consisting of any sample size
2. Interpolation of large area design spaces to concise localized areas
3. Linear datasets consisting of a small concise design space

When NOT to use RBFNN metamodels

1. Extrapolation of concise datasets to a larger area of the design space
2. Any highly non-linear dataset

Radial Basis Function Neural Networks require large amounts of preconditioning in order to reach the performance levels of either PRS or Kriging. In order to compete with the performance of the other techniques, the ideal radius value must be determined prior to constructing the final model. This process took hours for the datasets used. However, if the datasets consisted of hundreds of sample points and thousands of design variables, constructing a RBFNN would take days to week to generate. On another note, if construction time is not an issue, RBFNN metamodels outperform both PRS and Kriging techniques on datasets with linear behavior.

5.4 Future Work

For the future development of the use of metamodels for conceptual design, the focus will be towards evaluating additional datasets to continue to test the performance of metamodels on high fidelity analysis data. Using actual legacy datasets to test the performance of these metamodels is a high priority. If the same kind of accuracy results can be achieved using legacy data, then hopefully the implementation of metamodeling techniques into conceptual design can be adopted into conceptual design the same way CAD has been adopted into detailed design.

In addition to testing more types of conceptual design datasets, more research will be done to implementing additional types of both Kriging Approximations and Radial Basis Function Neural Networks. Kriging Approximations are already the dominating performer for non-linear datasets. However, many different alterations exist for Kriging including Co-Kriging, Collocated Kriging, and even Collocated-Co-Kriging Approximations. Possibly one of these may be able to approximate linear datasets better than standard Kriging Approximations.

RBNN metamodels also have potential. With all the possible construct methods for neural networks, any of these could prove to perform better for non-linear datasets or for extrapolation. There are many different possibilities when it comes to neural networks. First of all, the method in which to generate the neural network can be done several different ways. The population of neurons can come directly from the sample points like was done in this thesis, or a subset of the population can be used until the RMSE for the model drops below a certain value.

The population can be generated randomly throughout the entire design space. One final implementation for RBFNN will be to implement a curve smoothing function with ideal radius values to see if this technique can better handle a highly non-linear dataset.

5.5 Acknowledgements

I would like to take this opportunity to thank everyone involved with the work presented in this thesis as well as supported me throughout graduate school. First and foremost, I would like to thank my family for their love and support throughout my years of graduate school. It would not have been possible without you behind me. Next I would like to thank my advisor Dr. Eliot Winer. Not only would this work not have been possible without you, but thank you for your guidance throughout my entire educational career.

To the faculty and staff of the Virtual Reality Application Center (VRAC) at Iowa State University, thank you for your support and making my graduate school experience as meaningful and worthwhile as possible. It has truly been a wonderful experience to be a part of.

And finally my research colleagues Andrew Koehring and Brett Nekolny for their help with the large number of FEA simulations to construct the metamodels upon.

BIBLIOGRAPHY

¹Otto, K. N. and Wood, K. L., *Product Design: Techniques in Reverse Engineering and New Product Development*, Prentice-Hall, Inc., Upper Saddle River, NJ, 2001.

²Lotter, B., *Manufacturing Assembly Handbook*, Butterworths, Boston, MA, 1986.

³Solidworks, 3D Mechanical Design and 3D CAD Software, Software Package, Ver. Office Premium 2008, Solidworks Corporation, Concord, MA, 1993, <http://www.solidworks.com>, accessed August 2008.

⁴Pro/ENGINEER, 3D CAD Parametric Feature Solid Modeling Software, Software Package, Ver. Wildfire 4.0, Parametric Technology Corporation, Needham, MA, 1985, <http://www.ptc.com>, accessed August 2008.

⁵Abaqus FEA, Software Package, Ver. 6.8, Dassault Systèmes, Suresnes, France, 1981, <http://www.simulia.com>, accessed August 2008.

⁶Ulrich, K. T. And Eppinger, S. D., *Product Design and Development, Third Edition*, McGraw Hill, Boston, MA, 2004.

⁷Ullman, D. G., *The Mechanical Design Process, Third Edition*, McGraw Hill, Boston, MA, 2003.

⁸Pro/CONCEPT, A Conceptual Design 2D and 3D Digital Sketchbook, Software Package, Ver. 3.0, Parametric Technology Corporation, Needham, MA, 1985, <http://www.ptc.com>, accessed August 2008.

⁹CATIA PLM Express, Software Package, Ver. 5.0, Dassault Systèmes, Suresnes, France, 1981, <http://www.3ds.com>, accessed August 2008.

¹⁰Stump, G., Simpson, T. W., Yukish, M., and Bennett, L., "Multidimensional Visualization and Its Application to a Design by Shopping Paradigm," *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, AIAA, Atlanta, GA, September 4-6, 2002, paper no. AIAA-2002-5622.

¹¹Myers, R. H., and Montgomery, D. C., *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*, Wiley, New York, NY, 1995.

¹²Box, G. E. P., and Draper, N. R., *Empirical Model Building and Response Surfaces*, Wiley, New York, NY, 1987.

¹³Simpson, T. W., Peplinski, J., Koch, P. N., and Allen, J. K., "Metamodels for Computer-Based Engineering Design: Survey and Recommendations," *Engineering with Computers*, Vol. 17, 2001, pp. 129-150.

¹⁴Cressie, N. A. C., *Statistics for Spatial Data*, rev., Wiley, New York, 1993.

¹⁵Sacks, J., Shciller, S. B., and Welch, W. J., "Designs for Computer Experiments," *Technometrics*, Vol. 31, No. 1, 1989, pp. 41-47.

¹⁶Booker, A. J., Conn, A. R., Dennis, J. E., Frank, P. D., Trosset, M., and Torczon, V., "Global Modeling for Optimization: Boeing/IBM/Rice Collaborative Project," 1995 Final Rept., ISSTECH-95-032, The Boeing Co., Seattle, WA, 1995.

¹⁷Simpson, T. W., Mauery, T. M., Korte, J. J., and Mistree, F., "Comparison of Response Surface and Kriging Models for Multidisciplinary Design Optimization,"

Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis & Optimization, AIAA, Reston, VA, 1998, pp. 381-391.

¹⁸Goffe, W. L., Ferrier, G. D., and Rogers, J., “Global Optimization of Statistical Functions with Simulated Annealing,” *Journal of Econometrics*, Vol. 60, No. 1-2, 1994, pp. 65-100.

¹⁹Simpson, T. W., Mauery, T. M., Korte, J. J., and Mistree, F., “Kriging Models for Global Approximation in Simulation-Based Multidisciplinary Design Optimization,” *AIAA Journal*, Vol. 39, No. 12, December 2001, pp. 2233-2241.

²⁰Broomhead, D. S., and Lowe, D., “Multi-variable Functional Interpolation and Adaptive Networks,” *Complex Systems*, Vol. 2, 1988, pp. 321-355.

²¹Poggio, T., and Girosi, F., “A Theory of Networks for Approximation and Learning,” *Proceedings of the IEEE*, IEEE, Vol. 78, Issue 9, September 1990, pp. 1481-1497.

²²Orr, M. J. L., “Introduction to Radial Basis Function Neural Networks,” Technical Report, Centre for Neural Systems, Edinburgh University, 1996.

²³Schmit, L. A., “Structural Synthesis—Its Genesis and Development,” *AIAA Journal*, Vol. 19, No. 10, 1981, pp. 1249-1263.

²⁴Schmit, L. A., Jr. and Farshi, B., “Some Approximation Concepts for Structural Synthesis,” *AIAA Journal*, Vol. 12, No. 5, 1974, pp. 692-699.

²⁵Box, G. E. P. and Wilson, K. B., “On the Experimental Attainment of Optimal Conditions,” *Journal of the Royal Statistical Society*, Vol. Series B, 13, 1951, pp. 1-38 (with Discussion).

²⁶Myers, R. H. and Montgomery, D. C., *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*, New York, John Wiley & Sons, 1995.

²⁷Giunta, A. A., Balabanov, V., Haim, D., Grossman, B., Mason, W. H. and Watson, L. T., "Wing Design for a High-Speed Civil Transport Using a Design of Experiments Methodology," *6th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Bellevue, WA, AIAA, 1996, pp. 168-183, AIAA-96-4001-CP.

²⁸Kaufman, M., Balabanov, V., Burgee, S. L., Giunta, A. A., Grossman, B., Mason, W. H. and Watson, L. T., "Variable-Complexity Response Surface Approximations for Wing Structural Weight in HSCT Design," *34th Aerospace Sciences Meeting and Exhibit*, Reno, NV, 1996, AIAA-96-0089.

²⁹Mavris, D. N., Bandte, O. and DeLaurentis, D. A., "Robust Design Simulation: A Probabilistic Approach to Multidisciplinary Design," *Special Multidisciplinary Design Optimization Issue of Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 298-307.

³⁰Chen, W., Allen, J. K., Mavris, D. and Mistree, F., "A Concept Exploration Method for Determining Robust Top-Level Specifications," *Engineering Optimization*, Vol. 26, No. 2, 1996, pp. 137-158.

³¹Koch, P. N., Allen, J. K., Mistree, F. and Barlow, A., "Facilitating Concept Exploration for Configuring Turbine Propulsion Systems," *ASME Journal of Mechanical Design*, Vol. 120, No. 4, 1998, pp. 702-706.

³²Balabanov, V., Kaufman, M., Knill, D. L., Golovidov, O., Giunta, A. A., Haftka, R. T., Grossman, B., Mason, W. H. and Watson, L. T., "Dependence of Optimal Structural Weight on Aerodynamic Shape for a High Speed Civil Transport", *6th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Bellevue, WA, AIAA, 1996, pp. 599-612, AIAA-96-4046-CP.

³³Koch, P. N., Simpson, T. W., Allen, J. K. and Mistree, F., "Statistical Approximations for Multidisciplinary Optimization: The Problem of Size," *Special Multidisciplinary Design Optimization Issue of Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 275-286.

³⁴Venter, G., Haftka, R. T. and Starnes, J. H., Jr., "Construction of Response Surfaces for Design Optimization Applications," *6th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Bellevue, WA, AIAA, 1996, pp. 548-564, AIAA-96-4040-CP.

³⁵Roux, W. J., Stander, N. and Haftka, R. T., "Response Surface Approximations for Structural Optimization," *6th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Bellevue, WA, AIAA, 1996, pp. 565-578, AIAA-96-4042-CP.

³⁶Koehler, J. R. and Owen, A. B., "Computer Experiments," *Handbook of Statistics*, Ghosh, S. and Rao, C. R., Eds., New York, Elsevier Science, 13, 1996, pp. 261-308.

³⁷Simpson, T. W., Lin, D. K. J. and Chen, W., "Sampling Strategies for Computer Experiments: Design and Analysis," *International Journal of Reliability and Applications*, Vol. 2, No. 3, 2001, pp. 209-240.

³⁸Giunta, A. A., Wojtkiewicz, S. F., Jr. and Eldred, M. S., "Overview of Modern Design of Experiments Methods for Computational Simulations," *41st AIAA Aerospace Sciences Meeting and Exhibit*, Reno, NV, AIAA, 2003, AIAA-2003-0649.

³⁹Hardy, R. L., "Multiquadratic Equations of Topography and Other Irregular Surfaces," *Journal of Geophysical Research*, Vol. 76, 1971, pp. 1905-1915.

⁴⁰Dyn, N., Levin, D. and Rippa, S., "Numerical Procedures for Surface Fitting of Scattered Data by Radial Basis Functions," *SIAM Journal of Scientific and Statistical Computing*, Vol. 7, No. 2, 1986, pp. 639-659.

⁴¹Bishop, C., "Improving the Generalization Properties of Radial Basis Function Neural Networks," *MIT Press Journal*, Vol. 3, No. 4, 1991, pp. 579-588.

⁴²Mullur, A. A. and Messac, A., "Metamodeling using Extended Radial Basis Functions: A Comparative Approach," *Engineering with Computers*, Vol. 21, No. 3, 2006, pp. 203-217.

⁴³Cressie, N. A. C., *Statistics for Spatial Data*, New York, John Wiley & Sons, 1993.

⁴⁴Gano, S. E., Renaud, J. E. and Sanders, B., "Hybrid Variable Fidelity Optimization Using a Kriging-based Scaling Function," *AIAA Journal*, Vol. 43, No. 11, 2005, pp. 2422-2430.

⁴⁵Gano, S. E., Renaud, J. E., Martin, J. D. and Simpson, T. W., "Update Strategies for Kriging Models for Using in Variable Fidelity Optimization," *Structural and Multidisciplinary Optimization*, Vol. 32, No. 4, 2006, pp. 287-298.

⁴⁶Won, K. S. And Ray, T., "Performance of Kriging and Cokriging Based Surrogate Models within the Unified Framework for Surrogate Assisted Optimization," *Evolutionary Computation*, Vol. 2, 2004, pp. 1577-1585.

⁴⁷Forrester, A. I. J., Sobester, A. and Keane, A. J., "Multi-fidelity Optimization via Surrogate Modelling," *Proceedings of the Royal Society A*, Vol. 463, No. 2088, 2007, pp. 3251-3269.

⁴⁸Simpson, T. W., Toropov, V., Balabanov, V., and Viana, F. A. C., "Design and Analysis of Computer Experiments in Multidisciplinary Design Optimization: A Review of How Far We Have Come—or Not," *12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, AIAA, Victoria, British Columbia Canada, September 10-12, 2008, paper no. AIAA-2008-5802.

⁴⁹Toropov, V. V. and van der Giessen, E., "Parameter Identification for Nonlinear Constitutive Models: Finite Element Simulation – Optimization – Nontrivial Experiments", *Optimal Design with Advanced Materials*, Pedersen, P., Ed., The Frithiof Niordson Volume, Proceedings of IUTAM Symposium, Elsevier Scientific Publishers, 1993, pp. 113-130.

⁵⁰Chang, K. J., Haftka, R. T., Giles, G. L. and Kao, P.-J., "Sensitivity-based Scaling for Approximating Structural Response," *Journal of Aircraft*, Vol. 30, No. 2, 1993, pp. 283-287.

⁵¹Hutchison, M. G., Unger, E. R., Mason, W. M., Grossman, B. and Haftka, R. T., "Variable Complexity Aerodynamic Optimization of a High-Speed Civil Transport Wing," *Journal of Aircraft*, Vol. 31, No. 1, 1994, pp. 110-120.

⁵²Venkataraman, S. and Haftka, R. T., "Design of Shell Structures for Buckling Using Correction Response Surface Approximations", *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis & Optimization*, St. Louis, MO, AIAA, 1998, pp. 1131-1144, AIAA-98-4855.

⁵³Vitali, R., Haftka, R. T. and Sankar, B. V., "Correction Response Approximation for Stress Intensity Factors for Composite Stiffened Plates", *39th AIAA/ASME/ASCE/ AHS/ASC Structures, Structural Dynamics and Material Conference*, Long Beach, CA, AIAA, 1998, pp. 2917-2922.

⁵⁴Madsen, J. I. and Langthjem, M., "Multifidelity Response Surface Approximations for the Optimum Design of Diffuser Flows," *Optimization and Engineering*, Vol. 2, No. 4, 2001, pp. 453-468.

⁵⁵Kuehl, R. O., *Design of Experiments: Statistical Principles of Research Design and Analysis*, Duxbury Press, Belmont, CA, 1994.

⁵⁶Otto, K. N., and Antonsson, E. K., "Extensions to the Taguchi Method of Product Design," *ASME Journal of Mechanical Design*, Vol. 115, No. 1, March 1991, pp. 5-13.

⁵⁷Myers, R. H., Khuri, A. I., and Carter Jr., W. H., "Response Surface Methodology: 1966-1988," *Technometrics*, Vol. 31, No. 2, May 1989, pp. 137-157.

⁵⁸Bucher, C. G., and Bourgund, U., "A Fast and Efficient Response Surface Approach for Structural Reliability Problems," *Structural Safety*, Vol. 7, 1990, pp. 57-66.

⁵⁹Bishop, C., "Improving the Generalization Properties of Radial Basis Function Neural Networks," *Neural Computation*, Vol. 3, No. 4, Winter 1991, pp. 579-588.

⁶⁰Zhang, R., Noon, C. J., Winer, E., Oliver, J. H., Gilmore, B. J., and Duncan, J. R., "Development of a Software Framework for Conceptual Design of Complex Systems," *3rd Annual AIAA Multidisciplinary Design Optimization Specialists Conference*, AIAA, Waikiki, HI, April 23-26, 2007, paper no. AIAA-2007-1931.

⁶¹Zhang, R., Noon, C. J., Winer, E., Oliver, J. H., Gilmore, B. J., and Duncan, J. R., "Immersive Product Configurator for Conceptual Design," *ASME International Design Engineering Technical Conferences 33rd Design Automation Conference*, ASME, Las Vegas, NV, September 4-7, 2007, paper no. DETC2007-35390.

⁶²VR Juggler, Software Package, Ver. 2.2.1, Infiscape Corporation, Ames, IA, 2003, <http://www.vrjuggler.org>, accessed August 2008.

⁶³Bierbaum, A., Just, C., Hartling, P., Meinert, K., Baker, A., and Cruz-Neira, C., "VR Juggler: A Virtual Platform for Virtual Reality Application Development," *IEEE Virtual Reality Conference 2001 (VR 2001)*, IEEE, Yokohama, Japan, March 13-17, 2001, pp. 89-96.

⁶⁴OpenSceneGraph, An Open Source High Performance 3D Graphics Toolkit, Software Package, Ver. 2.6.0, OpenSceneGraph Professional Services, Prestwick, Scotland, 2001, <http://www.openscenegraph.org/projects/osg>, accessed August 2008.

⁶⁵GTK, An Open Source Graphical User Interface, Software Package, Ver. 2.13.7, GNOME Foundation, Cambridge, MA, 2000, <http://www.gtk.org>, accessed August 2008.