Test Planning and Validation through an Optimized Kriging Interpolation Process in a Sequential Sampling Adaptive Computer Learning Environment

by

Jeremy Lamar Barnes

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
August 4, 2012

Keywords: Kriging, Validation, Design of Experiments, Budget,
Space Filling Designs, Variogram

Approved by

Saeed Maghsoodloo, Chair, Professor Emeritus of Industrial and System Engineering
Alice Smith, Professor of Industrial and System Engineering
Mark Carpenter, Professor of Statistics

Abstract

This dissertation explores Kriging in an adaptive computer learning environment with sequential sampling. The idea of this Design for Kriging (DFK) process was first mentioned in [1] titled "Kriging for Interpolation in Random Simulation". The idea presented by [1], paved the way for continued research in not only applications of this new methodology, but for many additional opportunities to optimize and expand research efforts.

The author proposes several advancements to the above process by introducing a novel method of interpolation through an advanced Design for Kriging (DFK) process with cost considerations, advanced initial sample size and position determination, search techniques for the pilot design, and standardized variogram calculations. We use the terminology variogram over semivariogram as described by [2]. The resulting applications in this research are two-fold. One is to use this process in the upfront experimental design stage in order to optimize sample size and Factor Level Combination (FLC) determination while considering the overall budget. The other application is the use of sampled empirical and interpolated data to form a representative response dataset in order to perform statistical analyses for validation of Monte Carlo simulation models. The DFK process is defined as:

1) Define factor space, boundaries, and dimensions
2) Determine initial sample size through cost considerations and estimation variance analysis. Determine FLCs by a space filling design performed by an augmented simulated annealing algorithm

3) Observe responses, with replication if required, at the initial sample size and FLC selection
4) After sample responses have been observed, perform Kriging interpolation at n_K^U where n_K^U is some number of unobserved FLCs
5) Calculate the estimation variance
6) Based on the results from steps 3-5, identify the next x^c candidate input combination set, $\{x_i^c, x_{i+1}^c, \ldots, x_{n_K^U}^c\}$, based on budget considerations and variance reduction, and repeat steps 3-5 using $\{x_i^c, x_{i+1}^c, \ldots, x_{n_K^U}^c\}$
7) After an acceptable prescribed accuracy measurement level is achieved or budget is exhausted, Krige the n_K^A observations to achieve a representation of the underlying response function

After DFK process is completed, statistically compare a verified Monte Carlo estimated response dataset (Y^{MC}) with the combination of the Kriging metamodel response dataset (Y^K) and the actual response data Y^S, to assess the model against the combined response dataset Y.

Acknowledgements

The author is grateful to Dr. Saeed Maghsoodloo for his guidance, meticulous attention to detail, insight, and agreement to oversee this work. The author owes this very dissertation effort and Ph.D. completion to Professor Emeritus Saeed Maghsoodloo's generosity and wisdom. The author must also recognize Dr. Alice Smith for her demanding, but continuous support for this effort. Dr. Alice Smith has produced a major contribution in the academic growth of the author through the efforts, assignments, and opportunities that she provided to the author. The author also wants to thank Dr. Mark Carpenter for his agreement and cooperation on this research. In addition, the author must recognize Dr. David Umphress for his positive support and guidance.

This work is dedicated to the following: Carol Furman (mother), Rickie Barnes (father), Hannah Harris (sister), Mary Alice Barnes (fraternal grandmother), Pauline Thrower (maternal grandmother) and wife to be, Dilcu Helvaci. Without each of these people, this work is not possible. The author shares his deepest gratitude to all of these people.

The author also wants to appreciate the MITRE Corporation and the Missile Defense Agency (MDA) for supplying the author with supplemental support that was critical for the completion of this degree. The author wants to add a debt of gratitude for all the friends and colleagues that encouraged the continuation of this research, despite all odds and setbacks.

Finally, the author must appreciate Dr. Wim van Beers and Dr. Jack Kleijnen. Their work of Kriging in simulation in conjunction with the author's personal interests inspired this research effort.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

BoK	Body of Knowledge
CI	Confidence Interval
CPU	Central Processing Unit
DoD	Department of Defense
DFK	Design for Kriging
DFKS	Design for Kriging Sampling
DOX	Design of Experiments
EA	Evolutionary Algorithm
EI	Expected Improvement
FLC	Factor Level Combination (x)
GoF	Goodness of Fit
GRG	Generalized Reduced Gradient
GUI	Graphical User Interface
IID	Independent and Identically Distributed
I/O	Input/Output
K-S	Kolmogorov-Smirnov
LC	Linear Combination
LHS	Latin Hypercube Sampling
LOS	Level of Significance

M&S	Modeling and Simulation
MC	Monte Carlo
MDA	Missile Defense Agency
MIL-STD	Military Standard
MSE	Mean Square Error
OK	Ordinary Kriging
SLREG	Simple Linear Regression
V&V	Verification and Validation
VBA	Visual Basic for Applications

Nomenclature

$\mathcal{B}$	Borel Set
$\mathcal{C}$	Correlogram
$\mathcal{N}(n \times n)$	Square neighborhood structure
$\mathbb{R}^d$	The n dimensional set of real numbers
Ψ	Linear cost scalar
$(1-\alpha)$	Confidence coefficient
ε	Random error term
θ	A parameter
μ	Population mean
ν	Degrees of freedom
σ	Population standard deviation
${\hat{\sigma}_K}^2$	Ordinary Kriging estimated variance
$\hat{\sigma}_{SK}^2$	Simple Kriging estimated variance
γ	Variogram function
$\boldsymbol{\Lambda}$	Kriging weight vector
λ_i	The ith Kriging weight
φ	Ratio of lag over range: $\frac{h}{a}$

ω	Lagrangian multiplier
$\gamma(h)_i^f$	Fitted variogram γ value
Σ	Covariance matrix
$\boldsymbol{c}$	Variogram matrix for a single unobserved FLC
$\boldsymbol{C}$	Variogram matrix for the selected FLCs
$\boldsymbol{p}$	Position vector p
p_i	ith position in $\boldsymbol{p}$
$\boldsymbol{q}$	Position vector q
q_i	ith position in $\boldsymbol{q}$
$\boldsymbol{X}$	Euclidean distance (lag) matrix
$\boldsymbol{x}$	Vector of Euclidean distances between observed FLCs and a single unobserved FLC
$\boldsymbol{Z}$	Response vector
c_0	Nugget
c_1	Partial sill
d_j	Distance per variogram function
$d(\boldsymbol{p}, \boldsymbol{q})$	Euclidean distance
h	Lag
lb	Lower bound
N	Total number of FLCs in the factor space
$N(h)$	Number of unique lags in n_{ir}
n_{ir}	A subset of the N possible FLCs

n_K^A	$N - n_K^U$ FLCs at which Kriging interpolation is performed
n_K^U	The number of unobserved Factor Level Combinations (FLCs) at which Kriging interpolation is performed
n_{MC}	Sample size for Monte Carlo model
n_{REP}	Number of replications
S	The square root of the unbiased estimator of population variance σ^2
$\{s: s \in D\}$	The variable s is defined as an element of set D
ub	Upper bound
U	Factor space
d	Dimension number
x_{min}	The FLC all of whose elements are at minimum
x^c	Next observational candidate point
$\{x_i, x_{i+1}, \dots, x_n\}$	Inputs for the initial observational set
$\{x_i^c, x_{i+1}^c, \dots, x_n^c\}$	Inputs for the next observational candidate set
$\max\{\hat{V}(x^c)\}$	Next observational candidate point with highest Kriging variance
Y^{MC}	Response dataset generated from Monte Carlo models
Y^K	Interpolation dataset generated from DFK
Y^S	Observed sample response dataset
Y	Combined response dataset Y^{MC} and Y^K
$\bar{z}$	Point estimate of mean response
$\hat{z}$	Point estimate of the response at a specific unobserved FLC
$Z(\mathrm{s})$	Random function at location $\{s \in D\}$

$Z(\bullet)$	The response of a random process
$\hat{Z}(\bullet)$	A Predicted response of a random process

CHAPTER 1

Introduction

This chapter introduces the research background and motivation for the dissertation, outlines the objectives that the research intends to achieve, and then gives the research methods adopted. The introduction provides the framework for the research that follows. The chapter concludes by describing the layout of the dissertation.

1.1 Background and Motivation

Kriging has traditionally been used in the realm of geostatistics since its inception in the 1960s and it has continued to gain importance in the deterministic and stochastic simulation community [3] and in the machine learning community [5]. The iterative nature of learning a predictive model is also known as active learning [6]. Kriging was developed by Georges Matheron and named in honor of Danie Krige, a South African mining engineer. Kriging is used extensively in geostatistics which is the mapping of surfaces from limited sample data and estimation of values at the unsampled locations [7],[8],[9]. There are several methods of Kriging, all with the intent to estimate through interpolation a continuous, spatial attribute at an unsampled site. Kriging is a form of generalized linear regression that forms an optimal (or best) linear estimator in a minimum mean square error sense [9] through partial differentiation. Kriging is formally defined as a random process described by $\{Z(s): s \in D\}$ where D is a fixed subset of $\mathbb{R}^d$ with a positive d-dimension [10] and $Z(s)$ is a random function at location $\{s \in D\}$.

This dissertation introduces a novel advanced DFK process and utilizes that process for test planning and simulation model validation. The DFK process is a general approach that can be used in many areas where data interpolation is needed assuming, as in any case, that the experiment meets the criteria and assumptions needed for the DFK algorithms to work. The application of Kriging can range from unexploded ordnance location prediction through the use of the logistic probability density function, topography resolution enhancement, image sharpening, computer graphical enhancements, and statistical validation and response data estimations. This advanced DFK process has applications in experiments that have a need to optimize interpolation accuracy while minimizing cost.

The advanced DFK process starts with the identification of an experiment/system. The experiment obviously contains a sample space, boundaries, inputs, outputs, and other relevant characteristics. After experiment identification and the establishment of factor space, the boundaries and dimensions are formalized. Utilizing a unique space filling design, along with cost considerations and estimation variance (${\hat{\sigma}_K}^2$), the initial sample size and FLCs are formalized. Next, empirical observed responses with the initial sample size and at the FLCs are collected from the experiment. Inputs that are independent cannot be used in Kriging since the Kriging coefficient matrix will become singular resulting in a zero determinant with no feasible solution for Kriging. After the responses from the sample observations with replication, if required, have been established, a fit of the covariance matrix is performed. This results in the function which is used to create the variogram (γ) where $\gamma^{-1} = \mathcal{C}$, where $\mathcal{C}$ is defined as a correlogram. Next, Kriging is performed at n_K^U unobserved points. The estimation variance is again calculated with the use of the variogram to determine if the current interpolations meet a prescribed accuracy level to determine where the next worst interpolation occurs. Kriging is performed again, within budget constraints, for another sample set n_K^U until an acceptable accuracy level is reached. Kriging is performed a final time to achieve the remaining unobserved responses n_K^A. With the use of Y^K and Y^S where $Y^S + Y^K = Y$, a surface map of the representative underlying function can be constructed. Next, a verified Monte Carlo model is run a sufficient number of samples each of size n_{MC}, with replication as required, to generate Y^{MC}. The datasets Y and Y^{MC} are statistically compared using the non-parametric Kolmogorov-Smirnov (K-S) Goodness-of-Fit (GoF) Test to decide if the cumulative distribution function of the empirical sample differs from the cumulative distribution function produced from the Monte Carlo Model. These methods are presented in Chapter 6.

1.2 Research Objectives

The research objectives mainly include three aspects. The research objectives are a unique combination of mathematical methods along with distinct applications for the justification of the establishment of such an interpolation method process.

The first objective explores the level of accuracy that can be attained between a simulation model and empirical limited data by using sequential Kriging in an adaptive computer learning environment with associated cost considerations. The response dataset generated through the interpolation capabilities of Kriging are compared with that of an associated Monte Carlo simulation model to infer validation statements of the Monte Carlo model.

The second objective determines techniques in augmented space filling designs, estimation variance calculations, and cost constraints to identify an initial sample size and FLCs for this application. This contribution allows one to determine not only the number of data points to sample or tests to be conducted, but also what FLCs of inputs should be selected to maximize the application of Kriging interpolation model. The second objective addresses using interpolation methods for test planning prior to executing any tests or collecting sample data. The reader should note that the calculation methods are presented in Chapter 5.

The third objective, although seeming to be multiple objectives, all have the intent of optimizing the DFK process. The context of optimization has two meanings herein. The first meaning is to optimize the estimation variance by selecting FLCs with the highest estimation variance and the second is to optimize the software in terms of Central Processing Unit (CPU) time. The third objective is the determination of an augmented simulated annealing process for sequential sampling providing an adaptive computer learning concept for the process by selecting a sample size weighted to where the interpolation estimates are the worst, i.e. the unknown response is the noisiest. In addition, a single method is evaluated for a covariance function for use when developing the variogram that will allow for increased accuracy of the variogram calculations. This is accomplished through the combination of iterative regression analysis combined with Generalized Reduced Gradient (GRG) or Evolutionary Algorithm (EA) as required. Additionally, the gamma (γ) values of the traditional variograms currently have to be recalculated for each additional input. We optimize the traditional variogram calculation process during the test planning phase through dynamic array slicing that eliminates the need for recalculations of the variogram for previous inputs.

1.3 Research Methods

The dissertation presented here improves the utility of Kriging and establishes an advanced DFK process with cost constraint considerations. The research develops and enhances an overarching process based on multiple mathematical theories that will improve interpolation accuracy, define minimal required observational samples with a goal of minimizing cost, and provide an advanced statistical measure to perform validation between computer model datasets and limited empirical datasets.

As is inherent in Kriging, the models are fully and clearly expressed and are differentiable, which will allow sensitivity analyses in the responses through partial differentiation to assist in selection of n_K^U. This approach combined with FLC selection aims to reduce the number of total iterations required.

The empirical research in this dissertation is important. The dissertation separates out each step of the process and uses several cases to investigate the improvement of the proposed step in the process. The dissertation then puts together the findings from DFK optimization and performs validation case studies based on physical systems (or representations thereof). The final results of the DFK process are presented in an effective manner.

Finally, comparison and contrast is widely used throughout the research. For the presented test cases, models or methods are used to implement, analyze and compare their advantages and disadvantages.

1.4 Dissertation Layout

The dissertation is divided into eight chapters including this first chapter entitled "Introduction". The layout and organization of remaining chapters are as follows.

Chapter 2 presents the literature review on Kriging. This includes the origins, applications, and its early development. The application review of Kriging pertains to Kriging in simulation, empirical sampling, and sequential sampling. Chapter 2 continues by expanding on the current Body of Knowledge (BoK) for validation. The purpose of this portion of the chapter is to express explicitly where this advanced interpretation process benefits validation research. The chapter continues with some review of simulation modeling including definitions for important and relevant statistical measures. The chapter finalizes with general information regarding the Monte Carlo method.

In Chapter 3, the Kriging methodology is presented. The principles, mathematical development, and structure are identified and introduced. Chapter 3 continues with discussion of the variogram or covariance functions. Next, an introduction to the terminology and ideology and assumptions behind the lag and the neighborhood structure when used in spatial interpolations is presented. Chapter 3 concludes with a brief summary.

Chapter 4 describes in detail the current DFK process. The layout for describing this process is through a detailed discussion of each step while utilizing an empirical example for clarity and demonstration of concept. The included demonstration pertains to an experiment in soil sampling for a certain area around Toomer's corner in Auburn, AL to determine the amount of soil contamination by the use of Spike 80DF. This chapter is critical for understanding the current approach. The reading of this chapter will provide the reader with the information needed in understanding the remaining chapters.

Chapter 5 describes specific advancements that pertain to the associated steps in the advanced DFK process. Chapter 5 will empirically investigate the effectiveness of each of the following:

1) Determination of n_{ir} when cost constraints are present along with estimation variability
2) Space filling designs to maximize the information observed from each input combination
3) Standard variogram fitted function
4) Stopping criteria based on budget considerations are investigated

Chapter 6 reassembles the seven step process into a comprehensive and cohesive interpolation method for generating Y^K. Iterative Kriging is performed through software to demonstrate augmentation of Y. Chapter 6 empirically analyzes how the process aides in the overall test planning and validation efforts. The systems (or representations) used were independently developed and given as black boxes in which to gather response data. This actual data (or the terminology "truth" data used by DoD) are augmented through the advanced DFK process to provide data to compare with the corresponding Monte Carlo simulations data. Chapter 6 concludes by evaluation of Y against Y^{MC}. This assessment yields an overall effectiveness of the proposed process.

Chapter 7 introduces the DFK software in order to satisfy the application needs. The software provides a user-friendly graphical user interface (GUI), contains built-in help, and is presented as an add-in to Microsoft Excel®. The detailed description about usage and manipulation of this functional application software will be discussed with graphical aides throughout. In addition, chapter 7 provides concise information in order to obtain and install the software. Chapter 8 concludes the dissertation and lays the foundation for future research directions.

CHAPTER 2

Literature Review

Simulation is the experimenting with or exercising a representation of an entity, a system or an idea under a multitude of objectives including, but far from limited to, acquisition and analysis [11]. Models begin in the conceptual stage which supports the development of the underlying theories and assumptions such that the structure, logic, and mathematical and casual relationships match a system that the model is set to represent [12].

2.1 Metamodels

Originally metamodels were constructed using regression analysis as discussed in [13]. Regression metamodels are categorized by order. First order metamodels have a response variable Y and are modeled as $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ ... + \ \varepsilon$ where β_i are the model parameters, each X_i $\{i: i = 1, 2, ... k\}$ is a model input, and ε is residual error. Second order metamodels include both pure quadratic and two-variable interaction terms and are modeled as follows $Y = \beta_0 + \sum_i \beta_i X_i + \sum_{i,j} \beta_{ij} X_i X_j + \varepsilon$ with similar variable representations as the first order metamodel.

Metamodeling techniques have evolved to include such methods as neural networks, Kriging, experimental designs, along with the traditional regression methods [14]. Metamodels allow improved understanding, a quicker turnaround time for data generation used in sensitivity analysis, optimization, and decision analysis. Metamodels provide lower fidelity of the full simulation model with a simpler model that can usually be constructed within a spreadsheet [15]. A metamodel approximates the Input/Output (I/O) transformation of simulation such that the metamodel relates its outputs to that of a system. Therefore, metamodel output approximations can help in determining a simulation model's validity.

In summary, metamodels will continue to be utilized as simulations become increasingly complex as they provide rapid information about the underlying simulation model. This information can be used in decision making, validation, or optimization of simulation models. Optimization of simulation models through the use of metamodels is not discussed as it is left for future research.

2.2 Origins of Kriging

Kriging, originally named "krigeage" by Pierre Carlier [10], was developed by Georges Matheron [16] and named in honor of Danie Krige, a South African mining engineer. The approach was developed to ascertain gold mining valuation problems. Initially, through exploration research, Kriging can be tracked back to the research of Wold [17], Kolmogorov [18],[19], and Wiener [20]. The process of Kriging is used extensively in geostatistics which is the mapping of surfaces from limited sample data and estimation of values at the unsampled locations [7],[8],[9]. Matheron [16] describes Kriging as a weighted average of available samples that can be used for predictions and with suitable weights the variance should be minimized [21]. Daniel Krige defined the term as a multiple regression which is the best linear weighted moving average of the grade of a type of rock that contains mineral for a block (or area) of any size by assigning an optimum set of weights to all the available and relevant data inside and outside the block [22]. Ord [23] states that Kriging is an interpolation method for random spatial process, and similarly Hemyari and Nofziger [24] define Kriging as a weighted average, where the weights are dependent upon location and structure of covariance of observed points [24]. The author argues against the later definition and instead states that the weighted average may depend upon location and structure of covariance of observed or unobserved points as long as the problem has a clearly defined bounded region and the normality assumption holds. The above statement is clarified in Chapter 5.

Originally, Kriging was a linear predictor. Later developments in geostatistics, Kriging was extended to nonlinear spatial prediction called indicator Kriging. As stated in [21], the origins of Kriging given by Cressie [25]. Cressie addresses Kriging from different disciplines and states the conclusion that Kriging is equal to spatial optimal linear prediction. There are several forms of Kriging, all initially formulated for the estimation of a continuous, spatial attribute at an unsampled point. Figure 1 shown below is a pictorial of a block for a two-dimensional sample with the open circle being the estimation location [9]. Extrapolations outside the block are possible, but are unreliable. The designs presented in this research are limited to blocks with some positive $\mathbb{R}^d$.

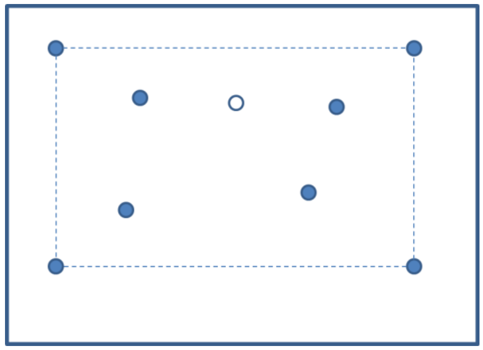


Figure 1: Two-Dimensional Block with Estimation Location

Kriging uses a weighted average of sample values to yield a linear estimate of the unknown value at a given location. This estimator is in the form of $\hat{Z} = \lambda_1 z_1 + \lambda_2 z_2 + \ldots + \lambda_n z_n$ where the weights λ sum to 1 (in Ordinary/Punctual Kriging) [26], [27] and z_i $(i = 1, 2, \ldots n)$ is the response data. The results or interpolations are unbiased with an estimation variance. Typically, weights are optimized using the variogram model describing the location of the samples and all the relevant inter-relationships between known and unknown values.

A variogram describes the covariance structure of the difference between two observations and is the backbone in ordinary Kriging [28]. A sample variogram is shown below in Figure 2.

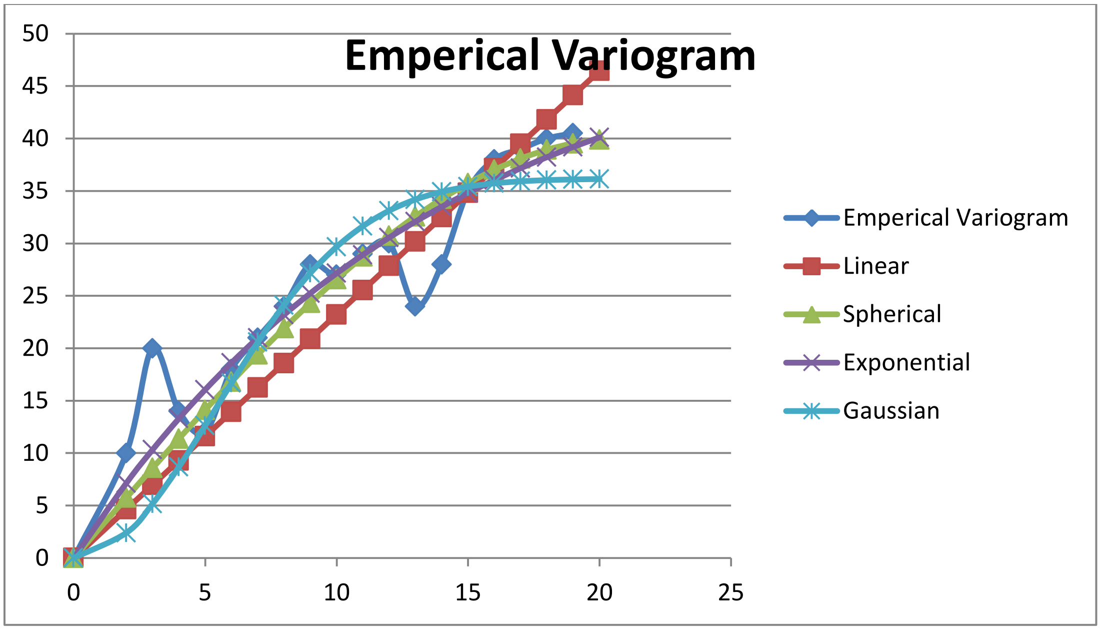


Figure 2: Sample Variogram

This section provides a brief introduction and history about Kriging which is formally introduced and discussed in Chapter 3.

2.2 Kriging Applications

Kriging now covers many research areas and disciplines. The original applications were that of ore and gold mining and quickly expanded to soil sciences [29],[30], geology [31],[32],[33], meteorology [34], and hydrology [35],[36]. Today Kriging applications continue to grow and can be seen in wireless network analysis, cost estimation, engineering design, simulation studies, and optimization.

The literature demonstrates the growth of Kriging applications in many new areas. These areas include biomechanical engineering [37],[38], wireless wave propagation [39],[40], material sciences [41],[42], engineering design [43],[44], economic analysis [45],[46], simulation interpolation [47],[48][3], and optimization [49],[50],[51]. The use of Kriging in simulation interpolation has given rise to the term Design for Kriging (DFK) as mentioned by Klein [52]. DFK was first introduced by Beers [3] and, arguably, was originally titled "Customized Sequential Designs for Random Simulation Experiments: Kriging Metamodeling and Bootstrapping". The introduction of DFK further increases the opportunity for application in design, planning, optimization, and simulation.

2.3 Kriging in Simulation

Random simulations are typically run with different combinations of simulation inputs while responses are observed, generally with replication. These I/O data are analyzed through statistical analysis such as low-order regression metamodels. Since a metamodel is an approximation of the I/O transformation implied by the underlying simulation model, the metamodel can be applied to the output of all types of simulation models. Even though Kriging metamodels have been applied extensively in discrete event simulation, Kriging has hardly been applied to random simulation. As with regression, in deterministic simulation, Kriging has been applied frequently and is attractive since this technique ensures that the metamodel's prediction has exactly the same value as the observed simulation output. When used in random simulation, Kriging produces an unbiased mean estimator of the simulation responses at unsampled locations [28].

For the introduction of Kriging in simulation, the classic references by Sacks et al. [53],[54] utilize Kriging interpolation as an inexpensive but efficient predictor for cost reduction in computer experiments that are computationally expensive to run. Mitchell and Morris [55] suggested that modifications can be made to handle simulation interpolation with random inputs while investigating the use of Kriging to evaluate the importance of input parameters in the deterministic simulation of a groundwater flow model. Barton [47] acknowledges the application of Kriging in random simulation, but points out that there is only a small set of samples available at this time. Barton [47] states that the current availability of computer code is a limiting factor. Kriging metamodels have been shown to yield more accurate predictions than low-order polynomial regression models [28] and also have been shown through the male gypsy moth flight phenology study that Kriging accuracy is nearly as precise as a complicated 6^{th} order polynomial regression model [56].

Kriging in random simulation environments is perhaps most represented by the work of van Beers and Kleijnen [57],[1],[4],[3]. They explored Kriging in three main areas. The first area investigated was Kriging in random simulation when variances of simulation outputs are not constant. The second investigation was the introduction of DFK in a deterministic simulation environment. That led to the third area of investigation and the basis behind this dissertation: DFK in random simulations. The DFK in random simulation study led to an alternative and more accurate variance measure as discussed by Kleijnen [58].

Physical simulation, like computationally expensive simulations, has similar difficulties in obtaining data with an adequate sample size because of cost and schedule constraints. Exploring Kriging interpolation for physical systems has been very few in the literature [59],[60]. This research provides more work in this area as it is warranted.

2.4 The Use of Kriging in Optimization

The literature describes that Kriging has recently been applied to optimization in a two areas. The first area is the assistance of Kriging in evolutionary optimization. The second area is in sequential Kriging optimization. Evolutionary algorithms are used to solve optimization problems where exact fitness functions may not exist. Kriging is used as a temporal or progressive fitness function to assist in evolutionary optimization. Research in this area has been conducted by Ratle [61]. Sequential Kriging optimization uses Kriging to approximate the objective function with a Kriging model, and then uses the Kriging model to determine points for sequential sampling [21]. Sequential Kriging optimization is similar to DFK in that a Kriging model is used to determine points to be sequentially sampled. Additionally, Biles et al introduced Kriging for constrained simulation optimization. Constrained simulations are simulations that impose the additional constraint of being related by some relation [62]. The research in [51], modeled an inventory system with the objective of finding the optimal values of reorder point and maximum inventory. Optimal solutions were found through the results of their experiments. Their results indicate that Kriging offers opportunities for solving constrained optimization problems in stochastic simulation.

Although Kriging offers potential improvements in optimization, little literature exists on improving the DFK process in a sequential sampling adaptive computer learning environment. This research addresses optimization in areas of initial sample size selection, variogram modeling, and reductions in the number of iterations in the sequential sampling environment.

2.5 Monte Carlo Simulation

Monte Carlo Simulation involves the use of pseudo random numbers to model systems where time plays no substantive role (i.e., static models). "The Monte Carlo method provides approximate solutions to a variety of mathematical problems by performing statistical sampling experiments on a computer," [63]. Monte Carlo allows generation of artificial data through the use of a random number generator and utilizing the underlying probability law of interest. To generate Monte Carlo inputs the cumulative distribution function (cdf) of the input(s) must be inverted. After inverting the cdf, uniform random numbers between 0 and 1 are generated and put onto a one-to-one correspondence with the inverted cdf, thus resulting in random inputs that can be used in Monte Carlo simulation. This procedure is on sound statistical ground because it can be proven that the cdf of all continuous variates have the $U(0,1)$ distribution.

Primary components of a Monte Carlo simulation are as follows:

1. pdf – the density function describing the system to be modeled
2. Random number generator – source of "random" numbers $U(0, 1)$
3. Sampling rule(s) – method for generating random samples from the specified $pdfs$ (generally based on the $U(0, 1)$ random numbers but can be transformed)
4. Scoring – recording the "outcome" of each sample

A secondary ("optional") component that is sometimes found in Monte Carlo models is variance reduction techniques – methods for reducing variance in the estimated solution (to reduce the required computer time).

2.6 Validation

Many industries ranging from commercial to military require the use of multifaceted representations to closely imitate complex real-world processes, functions, systems, and entities. These representations are more commonly known as models with the process known as modeling. While models can range from mathematical approximations, physical representations, diagrams, etc., the models in this dissertation are developed and utilized in a controlled test case environment with a limited initial empirical dataset [64]. Simulation models, in general, are extensively used to gain insight into advanced configurations of systems and sub-systems that interact together with some common goal.

The users and stakeholders of these simulation models are rightfully concerned whether the information derived from these models are valid, which means a viable application that satisfies the metrics of simulation models' objectives or intended use [13]. Also note that the term confidence here does not coincide with the statistical definition of confidence but instead is defined as self-assurance. Benefits from valid simulation models and simulation models in general are far-reaching and range from decision making [65], predicting (forecasting) through interpolation and extrapolation, response surface mapping, operating procedure policies, to evaluating/defining performance characteristics.

Although there are great benefits from valid simulation models, the reader should be aware that there are many difficulties in gaining the confidence level in these models to claim an agreed upon statement of validity. A major issue that complicates validation of all simulation models is the wide variety of quantitative and qualitative approaches that contain uncertainty, risk, and the typical high cost involved in gaining a required confidence level from the stakeholder(s) that the model(s) at hand are valid. This is reiterated by the quote [12] "there is no set of specific tests that can be easily applied to determine the validity of the model." Validation costs not only involve life-cycle model development, testing, capital resources including human capital, and other cost factors, but also the price of a commonly accepted and regarded standard [66] known as independent (or third party) validation. Independent validation is not only desirable but offers benefits, such as the lack of bias, when used as an integral part of the overall validation effort [67].

The validation issue compounds when models are accompanied by very limited amounts of response data that are available, or can be realistically obtained. For clarity, response data also called truth or empirical data is regarded as any actual system data, performance characteristics, and/or model parameters that have been collected and is in an acceptable state to analyze the simulation model against. It is extremely important to notice the words "acceptable state". Response data need to be rigorously analyzed to ensure that it represents an accurate representation when estimating parameters.

All simulation models are validated to some level depending on many factors such as cost, schedule, criticality and data availability. Simulation models can be categorized into three types of validation modeling environments: models with no real data, models with limited available data, and models with a multitude of available data. Each of these situations calls for specific validation methods. The categorized type of validation for this dissertation focuses on simulation models using limited data sets. Since model validation is specific to the intended purposes of the models, all boundaries, accuracies, parameters, and conditions will be clearly identified as described in [12] prior to model development. Validation is not to be performed at any specific stage of the model development life-cycle but at increments in between and throughout. Generally accepted life-cycle descriptions are found in MIL-STD-499B [68] and can be reviewed for clarification at the reader's digression. As stated, validation of a simulated model should be completed at increments during model development. Validation therefore should be an integral part of a simulation model's development. Different methodologies and techniques apply depending on where the model is in the development cycle, the availability of data, and the type of model at hand. Even with different methodologies and techniques available, programmatically five primary areas are involved for simulation model validation: requirements validation, conceptual model validation, design verification, implementation verification/validation, and operational validation [69].

This dissertation from a simulation model point of view focuses on operational validity with the assumption that the representative system or experiment exists in which to draw limited response data through sequential sampling. Sequential sampling limited response data yields points that allow for estimations of the underlying response function. As response data are obtained throughout the sequence, the response functions become more accurate and precise. The precision and accuracy is not due to sequential Kriging, but due to increasing sample size associated from sequential sampling. Using this given response function (set of weights for Kriging) information, interpolation can be used to estimate the unsampled response points using algorithms such as regression and Kriging.

Validation requires three steps. The first is the process of validation. The second is the methodologies of validation that are available and can be applied throughout the life-cycle of a model. The final step is the validation techniques. Many techniques (prescriptive and descriptive) can be used that overlap the different methodologies while overarching validation process remaining relatively the same. Validation is depicted by Figure 3.

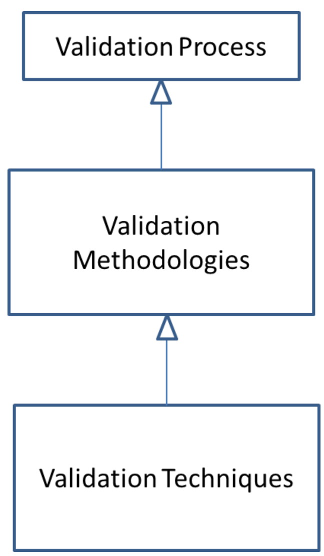


Figure 3: Overall Validation

The overarching approach to validation appears to be straightforward, but the details involved are very dynamic and complex depending on the application.

2.7 Validation Process

The validation process begins by obtaining pre-test prediction data through methods such as MC. Next, test data are gathered and compared with the MC data, and results are determined to either be consistent or different. If the results are consistent, they should be formally recorded and therefore should be deliverable. This information is categorized as the current state of validation. The states will vary from initial, to ongoing, to final as the life-cycle of a model matures. If the results are different when compared, an analysis should be conducted to determine if the discrepancy is in the test condition, the model, human error, and/or the interface/component performance. If it is determined that it is in the model, then the model should be altered and the process should be repeated. If the discrepancy is in the test condition then one has to determine if the test condition met all the objectives. If all objectives have been met, then formally record the results. If the objectives were not met, then replan the test. Finally if the discrepancy was caused due to system interface/component performance (or human error) then the governing Validation and Verification (V&V) authority has to decide whether the error was due to an anomaly, reevaluate the system and document the results or to modify the Modeling and Simulation (M&S) and start the process over. The general validation process described by [69] is shown below in Figure 4.

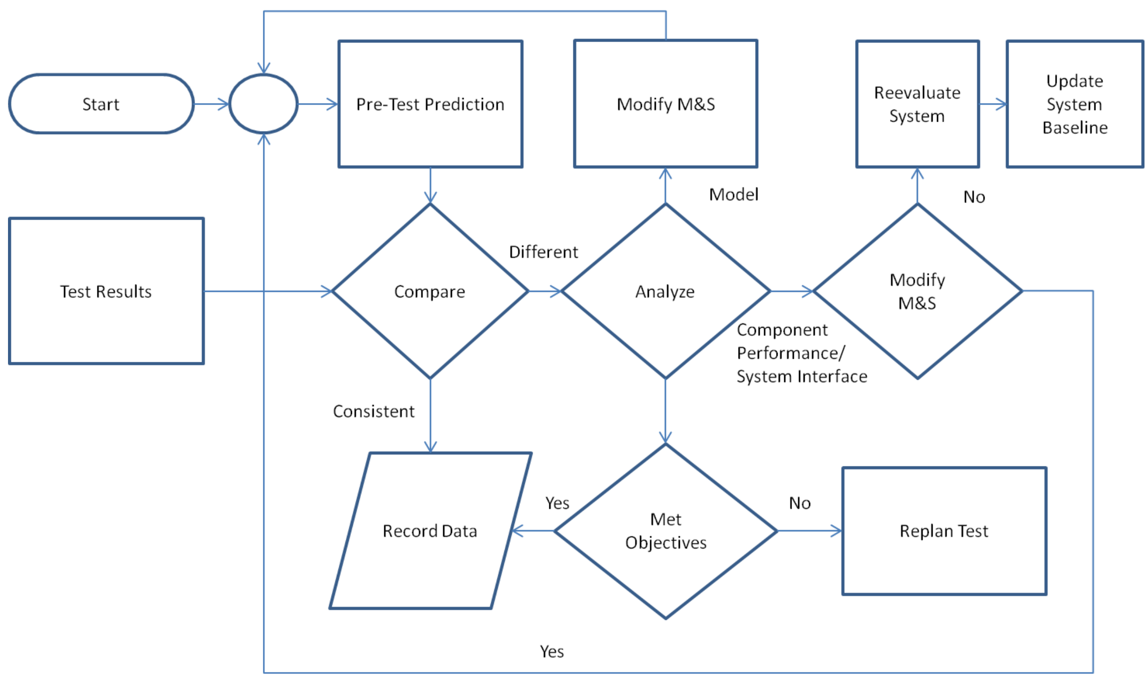


Figure 4: Observational Results Validation

This process obviously assumes that the model has been developed and partially undergone verification tasks.

2.8 Validation throughout the Model Life-Cycle

The life-cycle of a simulation model generally goes through five stages. These stages are the pre-concept and concept definition, defining mathematical relationships, model development including metamodels, operational stage, and the support stage. These stages do not have to occur linearly as in a waterfall development and may contain feedback loops between them. Verification is considered complete after the models have been developed and are at a level of maturity in order to proceed to validation.

2.9 Validation Methodologies

Validation occurs through various types of methodologies and varies throughout the life-cycle development of a simulation model [66] which was previously described. The overall validation process should be completed by using one or more of the methods described in [12] and summarized in Table 1: Validation Methodology.

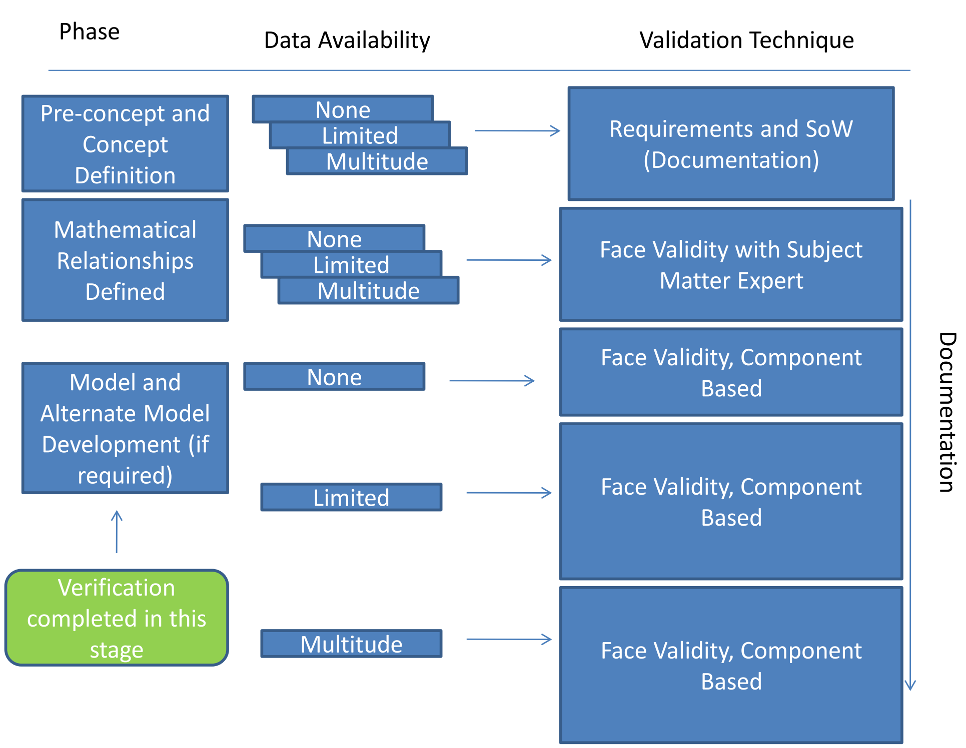
Table 1: Validation Methodology

Validation Methodology	**Description**
Comparison to Other Models	Comparing the result of one model to that of a valid model
Degenerate Tests	Removing portions of the model and observing the behavior
Event Validity	The events of occurrences of the simulation model are compared to the real system
Extreme-Conditional Tests	Testing a model at its boundaries and observing the behavior
Face Validity	Asking experts to view the I/O stream
Fixed Values	Allows the developer to check the output against calculated values
Historical Data Validation	Allows usage of data to build and test a model
Multistage Validation	Develop assumptions, empirically test where plausible, and compare input output relationship
Operational Graphics	Use graphical images or renderings to demonstrate the model
Sensitivity Analysis	Change a value such as the input of the model and determine the effect of the output
Predictive Validation	Run the model in advance of the actual system to be tested and review results after the system has completed the same cycle
Traces	Tracing an entity through a system
Turing Tests	Ask experts to discriminate the difference between the model and the actual system

Various techniques can be used in each of the methodologies described.

2.10 Data Availability

The nature of data are; unavailable, limited available, and a large amount of available data. Data originate from either the model or the referent test. Despite where data originate, it will always come in one of the above three forms. Validation methodologies and techniques of models differ greatly depending on the nature of available data. Statistical theory shows that as the amount of available data increases, the confidence level in validity will increase. This assumption is based on parameters such as system complexity, but should be considered a reasonable assumption under most nominal conditions. Figure 5: Validation through the Life-cycle as shown below is a summary of the previous topics discussed.



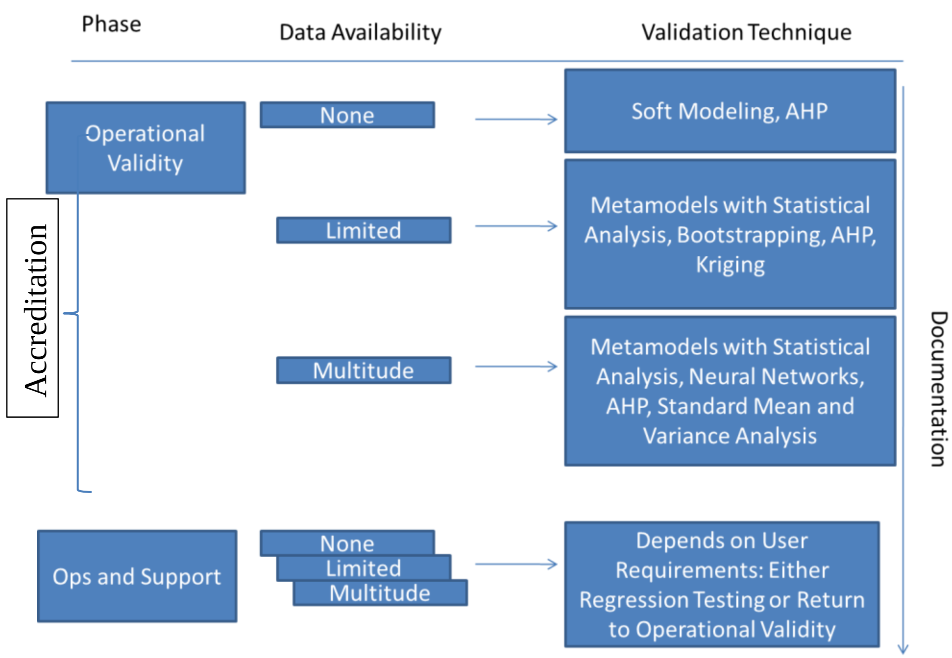


Figure 5: Validation through the Life-cycle

Validation can be incrementally performed or performed in one comprehensive undertaking. Despite the approach, the overall process of validation will remain the same. For achieving successful levels of validity, a combination of different validation methodologies and techniques, depending on data availability and model maturity, should be considered carefully. Expertise in applying the validation process along with a well-balanced and managed schedule and budget will ensure successful validation on the M&S of interest.

2.11 Validation Techniques

Various techniques are used in validation of M&S. These validation techniques are separated into two categories: descriptive and prescriptive. Descriptive techniques are considered "what to do" approaches for validation and the prescriptive techniques are considered "how to do" approaches for validation processes [70]. Typically, the techniques described as prescriptive are focused more on mathematical analysis, typically statistical in nature. The descriptive techniques focus more on the programmatic portion of validation. This research introduces an advancement of the prescriptive interpolation technique of Kriging through a novel DFK with cost constraint process. This new process will be compared to other common statistical measurement used in validation to provide an empirical measure of improvement of this advanced technique over standard methods.

2.12 Literature Review Summary

The literature describes a multitude of research in areas of Kriging and validation. Kriging has been used extensively in traditional applications such as geology and soil sampling. The literature supports increasing applications of Kriging in simulation, design, and optimization in today's modern technological environments. In lieu of the literature, research is needed in the DFK process and utilization of Kriging models for physical experimentation. Improving the DFK process will allow for test planning, decreased computational intensity in sequential sampling, and decreased cost in physical sampling. The improvements also set the stage for even further advancements and applications for this unique process in future.

CHAPTER 3

Methodology

This chapter describes the mathematical concepts that are presented throughout Chapters 4, 5, and 6. The other mathematical methodologies that accompany the advanced DFK process are presented in Chapter 5.

3.1 Kriging Methodology

First, it is important to establish the definition of Euclidean distance. Given points p and q, the length of the line segment connecting them is $\overrightarrow{pq}$. The distance in any $n - dimensional$ space from p to q can be stated as:

$$d(\boldsymbol{p}, \boldsymbol{q}) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

where $d(\boldsymbol{p}, \boldsymbol{q}) \geq 0$ is the separation distance between two points.

There are several forms of Kriging, all with the intent to estimate a continuous, spatial attribute at an unsampled site. According to [9], "Kriging is a form of generalized linear regression for the formulation of an optimal estimator in the minimum mean square error sense". Simple Kriging provides a gateway into more detailed methods of Kriging. Simple Kriging is limited due to its simplicity and embedded assumptions. Ordinary Kriging, the method used in this research, is the most widely used Kriging method and is derived from the principles founded in simple Kriging. The acronym B.L.U.E is associated with ordinary Kriging. The acronym stands for Best Linear Unbiased Estimator [8]. Ordinary Kriging is linear since it estimates weighted linear combinations of data. It is unbiased in the sense that it attempts to have zero mean residual. Finally, ordinary Kriging is considered "best" since it tries to minimize the variance of errors. Practically speaking, the goal of ordinary Kriging is unattainable as the true mean error and variance are almost always unknown. This implies that it cannot guarantee zero mean error or that variance is minimized. The best attempt is to build a model from available data and work with the average error and the error variance. In ordinary Kriging, a probability model is used such that the bias and error variance can both be calculated. Weights should be chosen such that the average error for the model is exactly zero and that the error variance is minimized.

To develop the Kriging methodology, the following must be defined in order to begin the mathematical development. Let $Z(\cdot)$ be a random function consisting of the collection of random variables $Z(\boldsymbol{x}, \omega) : \{x : x \in D\}$ where D is a fixed subset of $\mathbb{R}^d$ in the positive dimension d and $\{\omega : \omega \in U\}$ where U is the sample space. U makes up a Borel set $\mathcal{B}$, the collection of all subsets U including $\emptyset\ and\ A_1, A_2, \ldots \in \mathcal{B}$. This stochastic function can be written as

$$Z(x) = \mu(x) + \varepsilon(x)$$

with the fitted value at an unobserved point X written as

$$\hat{Z}(x) = \sum_{i=1}^{n_{ir}} \lambda_i Z(x_i)$$

taken from a linear combination of $Z(x_i)$ observed values.

To develop the details of the Kriging mathematical model a quick review of the fundamental properties of linear combinations of random variables is provided. The derivations and proofs can be found in [9].

Let $Z(x)$ be a random variable of a continuous random function, where the FLC $x = [x_1\ x_2\ \dots x_d]^T$ then for any constant coefficient λ the following holds true

$$E[\lambda Z(x)] = \lambda E[Z(x)]$$

This is derived from the distributive property of the linear mathematical first moment operator $E[\cdot]$, $E[\lambda Z(x)] = \int_{-\infty}^{\infty} \lambda\, Z(x) f(z) dz$, where $f(z)$ is the probability density function, λ is a constant and can therefore be pulled out of the integral resulting in

$$\lambda E[Z(x)] = \lambda \int_{-\infty}^{\infty} Z(x) f(z) dz = \lambda \mu(x)$$

where the integral is the expected value of $Z(x)$.

Then for any coefficient λ_i the following summation holds true

$$E(\hat{Z}) = E\left[\sum_{i=1}^{n_{ir}} \lambda_i Z(x_i)\right] = \sum_{i=1}^{n_{ir}} \lambda_i E[Z(x_i)] = \sum_{i=1}^{n_{ir}} \lambda_i \mu(x_i)$$

Next the second moment needs to be determined. Let $Z(x)$ be a random function of location. Then for any coefficient λ_i the following holds true

$$V(\hat{Z}) = E\left[\left\{\sum_{i=1}^{n_{ir}} \lambda_i Z(x_i)\right\}^2\right] - \left\{\sum_{i=1}^{n_{ir}} \lambda_i E[Z(x_i)]\right\}^2 = \sum_{i=1}^{n_{ir}} \sum_{j=1}^{n_{ir}} \lambda_i \lambda_j E\left[Z(x_i) Z(x_j)\right],$$

which reduces to

$$V(\hat{Z}) = \sum_{i=1}^{n_{ir}} \sum_{j=1}^{n_{ir}} \lambda_i \lambda_j \sigma_{ij}$$

According to [9], the simple Kriging mathematical model is based on the following three assumptions followed by two definitions:

1. The sampling is a partial realization of a random function $Z(x)$ where x denotes spatial location.
2. The random function is second order stationary. This implies that moments involving up to two variates are insensitive to any joint spatial translation, depending on the Euclidean distance.
3. The assumption that the mean is known. This assumption is unique to simple Kriging.

There are definitions that must be stated in order to accurately develop the Kriging mathematical model. Let Z be a second order stationary random function with mean μ. The estimator $\hat{Z}(x_0)$ at input "location" x_0 is given by the following linear combination of random variables at sites x_i, where x_i respresents a FLC, considered in the sampling

$$\hat{Z}(x_0) = \mu + \sum_{i=1}^{n_{ir}} \lambda_i (Z(x_i) - \mu) = \sum_{i=1}^{n_{ir}} \lambda_i Z(x_i)$$

Now let $Cov(x_i, x_j)$ be the covariance of a second order stationary random function $Z(x)$; then the general expression for the variance σ^2 at unsampled site x_o written as $\sigma^2(x_o)$ for convenience is equal to

$$\sigma^2(x_0) = V\big[\hat{Z}(x_0)\big] = V\sum_{i=1}^{n_{ir}} \lambda_i Z(x_i) = V\sum_{i=1}^{n_{ir}} \lambda_i[\mu(x_i) + \varepsilon_i]$$

$$\sigma^2(x_0) = V[\sum_{i=1}^{n_{ir}} \lambda_i \varepsilon(x_i)]$$

$$\sigma^2(x_0) = V\left[\sum_{i=0}^{n_{ir}} \lambda_i \varepsilon(x_i)\right] when\ \lambda_0 = -1$$

$$\sigma^2(x_0) = \sum_{i=0}^{n_{ir}}\sum_{j=0}^{n_{ir}} \lambda_i\lambda_j Cov(\varepsilon(x_i), \varepsilon(x_j))$$

Separating the $i = 1$ term and using $Cov((x_i), (x_j)) = Cov((\varepsilon(x_i), \varepsilon(x_j))$, then

$$\sigma^2(x_0) = \sum_{i=0}^{n_{ir}}\sum_{j=1}^{n_{ir}} \lambda_i\lambda_j Cov(x_i, x_j)$$

According to [9], the purpose of simple Kriging is to find a set of weights for the estimator that yields the minimum mean square error. The solution to this system of equations is known as a set of normal equations. Now, let m be a positive integer and $\{\alpha_1, \alpha_2, \ldots, \alpha_m\}$ be a set of real numbers, and let $\{x_1, x_2, \ldots, x_m\}$ be a set of points in an n-dimensional Euclidean space. Then the continuous function $\phi(x_i, x_j)$ is said to be positive definite if

$$\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i\alpha_j\phi(x_i, x_j) > 0$$

Let λ_i be the weights, written in matrix notation as $\mathbf{\Lambda}$, for the simple Kriging estimator and let $Cov(\cdot)$ be the covariance for the random function. If the covariance is positive definite then two conclusions can be made according to [9]. The first conclusion is the weights produce the minimum estimation variance and are the solution to the covariance matrix Σ below:

$$\sum_{i=1}^{n_{ir}} \lambda_i Cov(x_i, x_1) = Cov(x_0, x_1)$$

$$\sum_{i=1}^{n_{ir}} \lambda_i Cov(x_i, x_2) = Cov(x_0, x_2)$$

$$\dots\dots\dots\dots\dots\dots\dots\dots\dots$$

$$\sum_{i=1}^{n_{ir}} \lambda_i Cov(x_i, x_n) = Cov(x_0, x_n)$$

and the second conclusion is that the estimation variance is positive definite.

To minimize the above system of equations, the partial derivatives of for $i = 1, 2, \dots, n_{ir}$ must be taken and set to zero. This will minimize the mean square error given by the weights.

This can be shown through the following

$$Minimize\ \Sigma \rightarrow \ \frac{\partial \sigma^2(x_0)}{\partial \lambda_i} = 0\ for\ i = 1, 2, \dots, n_{ir}$$

Recall the following to demonstrate the nonnegative variance

$$\sigma^2(x_0) = \sum_{i=0}^{n_{ir}} \sum_{j=1}^{n_{ir}} \lambda_i \lambda_j Cov(x_i, x_j)$$

With the unbiased estimator established, the second or stationary assumption proven to hold, and the variance minimized in a mean square error sense through differentiation, it's important to note the simple estimation variance, $\sigma_{SK}^2(x_0)$, which is given by

$$\sigma_{SK}^2(x_0) = Cov(x_0, x_0) + \sum_{i=1}^{n_{ir}} \lambda_i Cov(x_0, x_i)$$

The $Cov(x_0, x_0) = 0$ when the nugget = 0 which is the assumption within this research. Thus, the resulting estimation variance is

$$\hat{\sigma}_{SK}^2(x_0) = \sum_{i=1}^{n_{ir}} \lambda_i Cov(x_0, x_i)$$

The simple Kriging estimation variance is turned into a constrained optimization problem by introducing a LaGrange multiplier for ordinary Kriging. This addition will be seen in the ordinary Kriging section to follow. This variance derivation allows for the developments in step 1 of the advanced DFK process to aide in the initial sample size selection.

For simplicity and completeness the simple Kriging system of equations will be discussed using matrix notation. First, let x_i's be the sampling sites of a discreet sample subset of size $n, i = 1,2, \dots, n$ and let $Cov(x_i, x_j)$'s be covariances. The covariance matrix, $\boldsymbol{\Sigma}$ is defined as

$$\boldsymbol{\Sigma} = \begin{bmatrix} Cov(x_1, x_1) & Cov(x_2, x_1) & \dots & Cov(x_{n_{ir}}, x_1) \\ Cov(x_1, x_2) & Cov(x_2, x_2) & \dots & Cov(x_{n_{ir}}, x_2) \\ \dots & \dots & & \dots \\ Cov(x_1, x_{n_{ir}}) & Cov(x_2, x_{n_{ir}}) & \dots & Cov(x_{n_{ir}}, x_{n_{ir}}) \end{bmatrix}$$

The above equation can also be written with the following notation where $\boldsymbol{C}$ represents the matrix of covariances, shown in the next section, is derived from the variogram function γ.

$$\boldsymbol{C} = \begin{bmatrix} Cov(0) & Cov(x_2, x_1) & \dots & Cov(x_{n_{ir}}, x_1) \\ Cov(x_1, x_2) & Cov(0) & \dots & Cov(x_{n_{ir}}, x_2) \\ \dots & \dots & & \dots \\ Cov(x_1, x_{n_{ir}}) & Cov(x_2, x_n) & \dots & Cov(0) \end{bmatrix}$$

Let λ_i be the optimal weights for the estimator and let $\sup T$ stand for the transpose of the matrix. Then **Λ** is the matrix

$$\boldsymbol{\Lambda} = [\lambda_1 \ \lambda_2 \ \ldots \ \lambda_{n_{ir}}]^T$$

For the third matrix development, let $Cov(\cdot)$ be the covariance of the random function, x_0 be the estimation location, and the x_i's be sampling sites of a discreet sample subset of size $n, i = 1, 2, \ldots, n$. Then $\boldsymbol{c}$ is the vector

$$\boldsymbol{c} = [Cov(x_0, x_1) \ \ Cov(x_0, x_2) \ \ \ldots \ \ Cov(x_0, x_{n_{ir}})]^T$$

The final matrix definition as explained in [9], states that $Z(x_i)$ be random variables of a random function with mean μ and let x_i be sampling sites, $i = 1, 2, \ldots, n$. Then the matrix $\boldsymbol{Z}$ is

$$\boldsymbol{Z} = \begin{bmatrix} Z(x_1) - \mu \\ Z(x_2) - \mu \\ \ldots \\ Z(x_{n_{ir}}) - \mu \end{bmatrix}$$

The developments in this section now provide the reader with an algorithm to perform simple Kriging. In summary, the algorithm consists of five steps. The steps are as follows:

1. Calculate each term in matrix **C**
2. Calculate each term in vector **c**
3. Solve the system of equations

   CΛ = **c** where $\boldsymbol{\Lambda} = [\lambda_1 \ \lambda_2 \ \ldots \ \lambda_{n_{ir}}]^T$

4. Compute the estimate(s).

   $\widehat{\boldsymbol{Z}} = \mu + \boldsymbol{Z}^{\boldsymbol{T}}\boldsymbol{\Lambda}$ where $\boldsymbol{\Lambda} = \boldsymbol{C}^{-1}\boldsymbol{c}$

5. Calculate the estimation variance

$$\hat{\sigma}_{SK}^2(x_0) = \sum_{i=1}^{n_{ir}} \lambda_i Cov(x_0, x_i)$$

Under the consideration that the formulation is independent from the physical nature of the spatial attribute, the algorithm is completely general and applies to the characterization of any spatial attribute satisfying the assumptions in [9]. The reader should note that $\boldsymbol{C}$ cannot be a singular matrix. If $\boldsymbol{C}$ is singular, then there will be no unique solution that will exist for the problem.

3.2 Ordinary Kriging

Ordinary Kriging is the most widely used form of Kriging and is the method of Kriging used in this dissertation. Simple Kriging requires information about the mean in order to solve the system of equations while minimizing the variance of the estimation error. Ordinary Kriging does not have the requirement of knowing information about the mean. This changes the problem from an unconstrained optimization problem into a constrained optimization problem. In order to solve the constrained optimization problem, a Lagrange method of multipliers is introduced [9].

In order to predict a response at x_0 the data values from n neighboring samples points x_k are combined linearly with weights λ_k resulting in the following

$$\hat{Z}(x_0) = \sum_{k=1}^{n_{ir}} \lambda_k Z(x_k)$$

As described in the literature, the sum of the weights must sum to one and the assumption is that the data are part of a realization of an intrinsic random function with a variogram $\gamma(h)$. Collecting the variances with the variogram is warranted for ordinary Kriging but not simple Kriging. This is due to the fact that simple Kriging does not include a constraint on the weights. Further proof that the weights sum to one and the variogram is authorized and shown in [9]. The estimation variance is

$$\sigma_{OK}^2 = E[\left(\hat{Z}(x_0) - Z(x_0)\right)^2]$$

and through linear combination can be calculated by

$$\sigma_{OK}{}^2(x_0) = \sum_{i=0}^{n_{ir}} \sum_{j=1}^{n_{ir}} \lambda_i \lambda_j Cov(x_i, x_j) + \lambda_{n_{ir}+1} \omega_{OK}$$

Ordinary Kriging is an exact interpolator in the sense that if x_0 is identical with a data location then the estimated value is identical with the data value at that point $\hat{Z}(x_0) = Z(x_k) \; if \; x_0 = x_k$ [9]. Minimizing the estimation variance with the constraint on the weights the ordinary Kriging system is obtained. The system is as described below

$$\begin{pmatrix} \gamma(x_1 - x_1) & \cdots & \gamma(x_1 - x_{n_{ir}}) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \gamma(x_{n_{ir}} - x_1) & \cdots & \gamma(x_{n_{ir}} - x_{n_{ir}}) & 1 \\ 1 & \cdots & 1 & 0 \end{pmatrix} \begin{pmatrix} \lambda_1^{OK} \\ \vdots \\ \lambda_{n_{ir}}^{OK} \\ \omega_{OK} \end{pmatrix} = \begin{pmatrix} \gamma(x_1 - x_0) \\ \vdots \\ \gamma(x_{n_{ir}} - x_0) \\ 1 \end{pmatrix}$$

where λ_k^{OK} are weights to be assigned to the data values and ω_{OK} is the Lagrange multiplier. The above equation can be rewritten as

$$\boldsymbol{C\Lambda} = \boldsymbol{c}$$

Although the data contained in these sets of matrices is different than the data contained in the simple Kriging formulation, we maintain the same notation for simplicity.

The purpose of Lagrange multiplier is to covert an unconstrained minimization problem into a constrained one [8]. This is accomplished by setting the partial first derivative of σ_{OK}^2 to zero. This produces n equations and n unknowns without adding any more unknowns. The following steps are taken

1. Calculate each term in matrix **C** through the fitted variogram function $\gamma(h)$
2. Calculate each term in vector **c** through the fitted variogram function $\gamma(h)$
3. Solve the system of equations

   $\boldsymbol{C\Lambda} = \boldsymbol{c}$ where $\boldsymbol{\Lambda} = [\lambda_1 \ \ \lambda_2 \ \ \ldots \ \lambda_{n_{ir}} \ \omega]^T$

4. Compute the estimate(s)

   $\boldsymbol{\widehat{Z}} = \boldsymbol{Z^T\Lambda}$ where $\boldsymbol{\Lambda} = \boldsymbol{C^{-1}c}$

5. Calculate the Ordinary Kriging estimation variance

$$\hat{\sigma}_{OK}{}^2(x_0) = \sum_{i=0}^{n_{ir}} \sum_{j=1}^{n_{ir}} \lambda_i \lambda_j Cov(x_i, x_j) + \lambda_{n_{ir}+1}\omega_{OK}$$

In summary, the ordinary Kriging methodology is similar to that of simple Kriging. Without knowledge of the population mean, the problem is turned into a constrained optimization problem which varies from simple Kriging by the addition of a Lagrange multiplier.

3.3 Variogram and Covariance in Kriging

The variogram provides the underlying calculations that allow the Kriging system of equations to be generated and solved. The variance increases as the distance increases until at a certain distance away from a point the variance will equal the variance around the average value, and will therefore no longer increase, causing a flat region to occur on the variogram called a sill. From the point of interest to the distance where the flat region begins is termed the range or span of the regionalized variable. Within this range, locations are related to each other, and all known samples contained in this region, also referred to as the neighborhood, and must be considered when estimating the unknown point of interest. This can be visualized on a two dimensional graph where the x-axis represents the lag (or bins) with a maximum value located at the range and the y-axis represents the variance with a maximum value located at the sill (the nugget + the partial sill).

The variogram describes the variance of the difference between two observations and is the backbone in ordinary Kriging [28]. The variogram is obtained through fitted functions that most closely resemble the experimental variogram. The experimental variogram is the array of calculations based on sampled response data. After the fitted variogram has been plotted, then the fitted functions are compared through minimum mean square error to determine the closest fit. The function with the closest fit is used to calculate the matrix $\boldsymbol{C}$ as previously defined. The traditional fitted functions ensure that variogram calculations result in a non-decreasing function. This holds the Kriging assumption that the covariance is non-decreasing as a function of spatial separation. Without using the fitted function for variograms, the Kriging estimates will be unreliable. The empirical variogram is calculated by

$$\gamma(h) = \frac{1}{2N(h)} \sum_{k=1}^{N(h)} [z(u_k + h) - z(u_k)]^2 \;\; h \in \mathbb{R}^d$$

where $N(h)$ is the number of distinct lag pairs $N(h) = \{u_k + h, u_k: u_k + h - u_k = h, k = 1, \ldots, N(h)$ and $[z(u_k + h) - z(u_k)]^2$ represents the difference square of sampled data at FLCs $u_k + h$ and u_k. As stated, for Kriging, we need to replace the empirical variogram with an acceptable variogram model. One reason is that the Kriging algorithm needs variogram values for lag distances other than the ones used in the empirical variogram. Another reason is the variogram models used in the Kriging algorithm need to be positive definite, in order the system of Kriging equations to be non-singular. Therefore, it is generally accepted that the application of Kriging must choose from a list of acceptable variogram models. A list of four frequently used models and the ones that are used in the test planning portion of this research are shown in Table 2.

Table 2: Variogram Models

	$\gamma(h) = c_0 +$
Linear	$c_1\left(\frac{h}{a}\right) if\ h \le a;\ c_1\ otherwise$
Spherical	$c_1\left(1.5\left(\frac{h}{a}\right) - 0.5\left(\frac{h}{a}\right)^3\right)\ if\ h \le a;\ c_1\ otherwise$
Exponential	$c_1\left(1 - exp\left(\frac{-3h}{a}\right)\right)$
Gaussian	$c_1\left(1 - exp\left(\frac{-3h^2}{a^2}\right)\right)$

Other variogram models, although not as common, exist and can be explored as future research as required.

To illustrate the variogram methodology completely, it is conducted through an example. Here we are drilling and taking a hypothetical soil sample value at distances of one foot apart from 1 foot to 10 feet. The values at each lag (increment) are calculated by the difference of the sampled values squared. The table below shows a summary of I/O in the first two columns, the lag h in the top row, and $[z(u_k + h) - z(u_k)]^2$ in the columns associated with each lag $\{h: h = 1, 2, \dots, 9\}$.

Table 3: Experimental Variogram Squared Calculations

FLC_i	Observation (Z)	Lag=1	2	3	4	5	6	7	8	9
1	5									
2	6	1								
3	4	4	1							
4	7	9	1	4						
5	7	0	9	1	4					
6	4	9	9	0	4	1				
7	2	4	25	25	4	16	9			
8	1	1	9	36	36	9	25	16		
9	3	4	1	1	16	16	1	9	4	
10	1	4	0	1	9	36	36	9	25	16

For proof of correlation, all pairs are calculated as described above and plotted below in the variogram cloud.

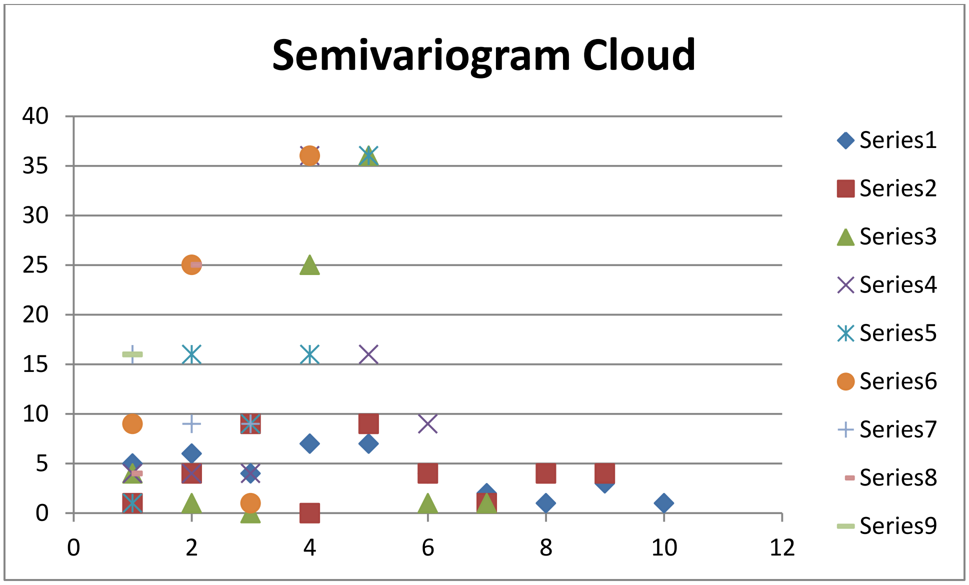


Figure 6: Variogram Cloud

The next calculation is to generate the experimental variogram values. This is done for each lag increment and is calculated by

$$\gamma(h) = \frac{1}{2N(h)} \sum_{\alpha=1}^{N(h)} [z(u_\alpha + h) - z(u_\alpha)]^2$$

where $[z(u_\alpha + h) - z(u_\alpha)]^2$ has previously been calculated. The calculations are shown in the table below.

Table 4: Experimental Variogram

	Lag (h)=1	2	3	4	5	6	7	8	9
Experimental Var	2	3.4375	4.8571	6.0833	7.8	8.875	5.6666	7.25	8

This allows for the experimental variogram to be plotted with the lag on the abscissa and the calculated γ values on the ordinate. The graph is shown below.

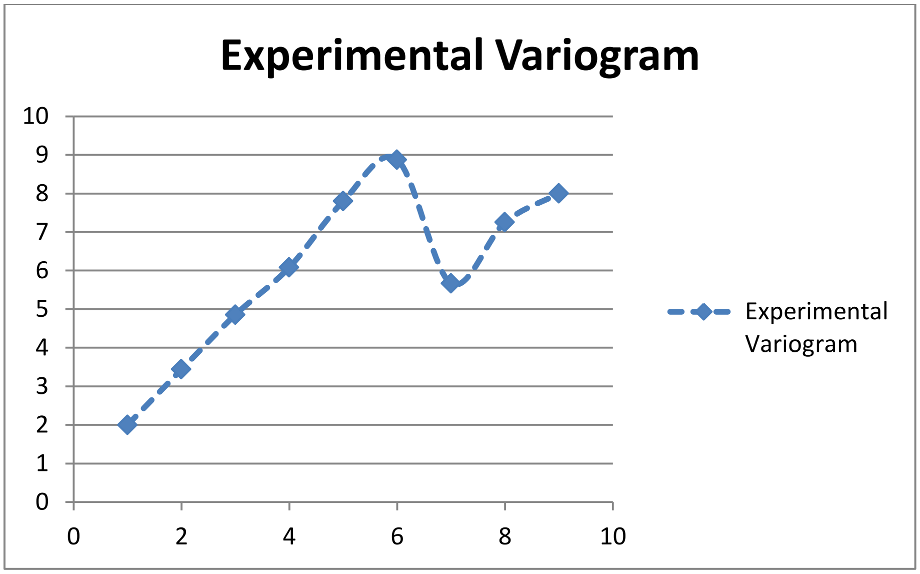


Figure 7: Plotted Experimental Variogram

The next objective is to fit a model which is then used to calculate the final γ values. This dissertation provides software that models variogram in three ways (1) to aid in the process of user selection of the appropriate model and user defined nugget, sill, and range values, (2) automatic selection of a model through sill and range parameter tweaking, and (3) through a standardized variogram model that is explained later in the research.

The nugget value is associated with measurement error or variation at distances smaller than the smallest lag. For the purpose of this research the nugget value is assumed zero by default but can be altered by the user after specific variogram selection. The range, a, is generally selected near the abscissa of the variogram where data no longer exhibit spatial autocorrelation, or at the maximum lag-value. The sill is the corresponding ordinate at "a". Autocorrelation is similarity of observations based on a separation unit. The software code base is in Appendix A and can be referenced for precise calculations. The resulting sill and range are

$c_1 = 7.79$, and $a = 7.69$ respectively. Using the parameters a, c_0, c_1, the following results are obtained for $\gamma(h)$:

Table 5: Fitted Variogram Models

	h=1	2	3	4	5	6	7	8	9
Linear	1.013	2.026	3.039	4.052	5.065	6.078	7.091	8.104	9.117
Sphere	1.511	2.970	4.327	5.530	6.527	7.267	7.699	7.771	7.432
EXPO	2.516	4.220	5.373	6.154	6.682	7.040	7.282	7.446	7.557
Gaus	0.385	1.431	2.855	4.330	5.599	6.536	7.141	7.487	7.662

Plotting these values and comparing them to the experimental variogram shows that the spherical model is a good fit for the sample data.

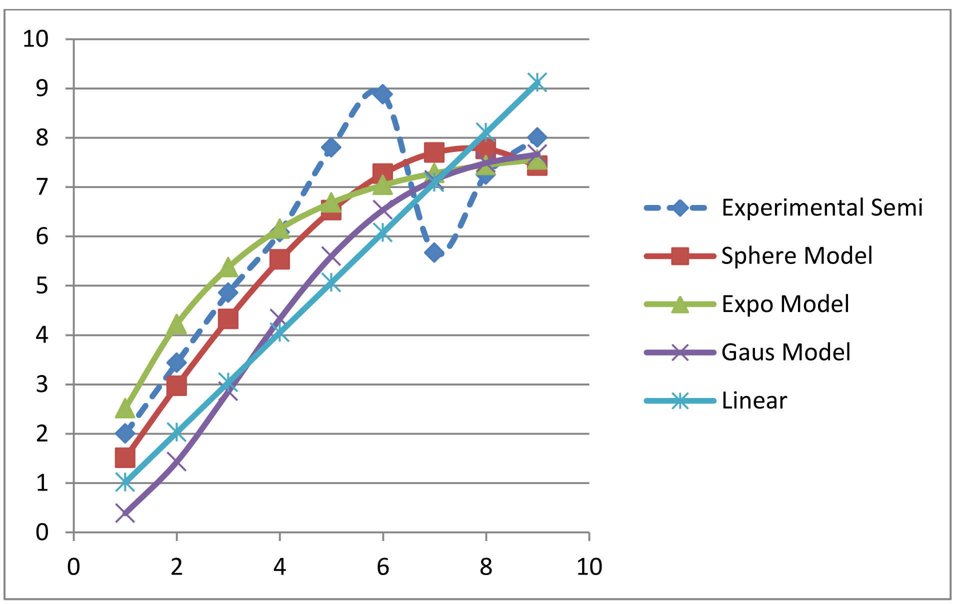


Figure 8: Fitted Variogram vs. Experimental Variogram

To prove the spherical model is the closet fit to the experimental variogram, the squared distances at each lag point are summed for each fitted model and shown in the table below. The spherical model has the lowest squared difference summation out of the four models.

Table 6: Summation of Squared Fitted Model Differences

	Squared Differences per lag h_i									Summation
Linear	0.974	1.992	3.306	4.126	7.480	7.823	2.029	0.729	1.248	29.707
SPH	0.239	0.218	0.281	0.306	1.621	2.586	4.129	0.271	0.323	9.974
EXPO	0.267	0.612	0.266	0.005	1.249	3.367	2.610	0.039	0.196	8.611
Gaus	2.607	4.027	4.007	3.073	4.847	5.472	2.175	0.056	0.114	26.378

Now that the $\gamma(h)$ values have been determined, this information is used to generate the covariance matrix

$$\begin{pmatrix} \gamma(x_1 - x_1) & \cdots & \gamma(x_1 - x_{n_{ir}}) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \gamma(x_{n_{ir}} - x_1) & \cdots & \gamma(x_{n_{ir}} - x_{n_{ir}}) & 1 \\ 1 & \cdots & 1 & 0 \end{pmatrix}$$

also referred to as simply $\boldsymbol{C}$ with the included Lagrangian multiplier. In addition, $\gamma(h)$ is also used to generate $\boldsymbol{c}$.

3.3.1 Lag

The lag or bin selection h is a critical component in the formation of the variogram. The determination of the lag size can help smooth out outlier data and reduce the computational intensity of the variogram calculations. The lag is generally and initially set to the smallest spatial difference or increment. Depending on the amount of FLCs, this can lead to a large set of data points that need to be calculated in order to generate the variogram model. Certain approaches and strategies are discussed throughout the literature, but with current computational capabilities and the scope of this research, the lag is set to the smallest increment of all the FLCs and remains unchanged throughout the DFK process.

3.3.2 Neighborhood

Another important decision has to be made about neighborhood selection. The neighborhood is the group(s) of data that exhibit similar characteristics. Distributing data into neighborhoods allows for multiple variogram models to be developed and for the use of multiple "instances" of Kriging to be performed and analyzed independently gaining a more accurate global interpolation. Although, numerous strategies have been discussed in the literature, this research places the entire factor space into one "global" neighborhood. This approach does not change the scope of the research and allows for future research as described in the final chapter.

3.4 The Bootstrap

Bootstrap methods allow for generation of data using resampling techniques [71]. The bootstrap is a data resampling simulation technique that is used to make estimations. It originates from the phrase "to pull oneself up by one's bootstrap." The bootstrap characteristics lend itself to I/O modeling in simulation experiments. The parametric bootstrap assuming normality is generally used in deterministic simulations, while distribution free bootstrapping is used in discrete-event random simulations and is used in Chapter 4.

In the literature presented by [71], a parametric bootstrap allows a method to apply a goodness-of-fit test to be performed on real data to determine the fit of different statistical distributions based upon calculating probabilities resulting from the bootstrap approximations. Applying a goodness-of-fit test in this manner should occur only when a wide array of known data has already been gathered from the system. Bootstrapping has the potential to aide in determining unknown statistical distributions in conjunction with goodness-of-fit tests against real system data. Other literature describes that the bootstrap method, a resampling technique, is also used in validating models with one data point when another data point is not feasible. This technique is described in [72] as the validity of the Bay of Biscay model and was tested against the historic documented encounter of the Allies against the German U-Boat threat just off the Bay of Biscay. The validation used two six month intervals of the battle and assumed a $(1 - \alpha) = 0.8$ confidence bound. The parameters of interest were U-Boat sightings and U-Boat kills. Questions about the model validity arose after 20 replications were executed and resultant data were analyzed using two tailed t-tests. These questions led [72] to consideration of non-parametric bootstrapping for model validation since this technique results in an estimated statistical distribution against which the model can be compared. The resampling was accomplished using Monte Carlo runs and the two scenario results were statistically more significant using bootstrapping.

Recently nonparametric bootstrapping has been combined with Kriging to generate more accurate and effective metamodels. Bootstrapping allows estimation of prediction variances for inputs that are yet to be simulated; therefore, it resamples and replaces outputs for each previous scenario already simulated under the assumption of independent and identically distributed data (IID). Promising results using bootstrapping have also been shown in these trace-driven simulations or simulations that have at least one of the same inputs as the real system [73]. This application is typically seen in traditional simulation, but has been shown to provide better estimates of variance when using Kriging as defined in [58].

Bootstrapped observations are generally denoted by a superscript *. For notation, bootstrap observations are formally written as

$$\{(y_{i,1}^*), \dots, (y_{i,m}^*)\}$$

where $i = 1, 2, \dots, m$ is the number of bootstrapped observations selected randomly from the replicated sample observations. Next, the bootstrapped averages are computed from the bootstrapped samples. This is formally written as

$$\bar{y}_i^*(m_i) = \frac{1}{m}\sum_{j=1}^{m} y_{i,j}^*$$

After the bootstrapped averages have been calculated, then the process is repeated for B number of replications where B is the bootstrap run size.

3.5 The Estimation Variance through Bootstrapping

It was proposed in [58] that the "correct" Kriging variance is estimated through the use of bootstrapping as the traditional Kriging variance was proven to underestimate the true variance. We return to the bootstrap sampled data

$$\bar{y}_i^*(m_i) = \frac{1}{m}\sum_{j=1}^{m} y_{i,j}^*$$

The Kriging predictors are then calculated by

$$\hat{y}^*\left(x_{n_{K_U}}\right) = \sum_{i=1}^{m} \lambda_i \bar{y}^*(x_i)$$

This is repeated for the bootstrap sample size B. Now the average of the Kriging predictors can be calculated. This is written formally as

$$\bar{\bar{y}}^*_{n_{K_U}} = \frac{1}{B}\sum_{b=1}^{B} \hat{y}^*_{n_{K_U},b}$$

With the bootstrapped Kriging interpolations calculated along with the averaged bootstrapped Kriging predictions, we now can proceed to calculate the "correct" Kriging variance. This is formally written as

$$V\left(\hat{y}^*_{n_{K_U}}\right) = \frac{1}{B-1}\sum_{b=1}^{B}(\hat{y}^*_{n_{K_U},b} - \bar{\bar{y}}^*_{n_{K_U}})^2$$

The purpose for bootstrapping section is only to give the reader familiarity with the technique as an example of the current DFK process is given in the next chapter. The scope of this research is to augment limited empirical data with an interpolation response set. Due to limited empirical data, the bootstrapping technique is not an appropriate method to be used in this research. Thus we use the traditional Kriging variance as stated in the literature and are able to derive useful and acceptable information with this technique.

3.6 Mean Square Error

Mean squared error (MSE) is a way to quantify the difference between an estimator and the true value of the quantity being estimated. Mean squared deviation or error is generally presented in the form of $MSE(\hat{\theta}) = E[(\hat{\theta} - \theta)^2]$ where θ represents a parameter of interest.

MSE is a commonly measure in statistics and is mentioned here since it plays an important part in the validation analyses.

CHAPTER 4

The Original Design for Kriging Process

This chapter describes the current DFK process step by step and is illustrated by an example to solidify the work. The experiment must first be generally defined as a set of n_{DOX} combinations with k FLCs. In defining an experiment, it must be noted that the experimental region is bounded by $lb_j \leq x_j \leq ub_j$ where $lb_j, ub_j \in R^d$ and $j = 1, \ldots, k$ [74]. R^d can take the form of a unit cube of positive dimension d. The goal of this original DFK process is to find the design where the Kriging interpolation accuracy is maximized while minimizing the amount of sampled input FLCs.

The following example provides a detailed procedure of the original DFK process. It is meant only to illustrate the combination of mathematical methods involved in DFK steps that generates satisfactory results. The original DFK process as described by [4], structures the problem statement to address interpolations of expansive simulation models where empirical data are realizable. The following example structures the problem such that the empirical dataset requires interpolation and simulation data are readily obtainable.

Soil samples are to be taken of a predefined area around the Toomer's oaks to determine the % concentration of Spike 80DF, z, contained in the soil. A minimal amount of samples must be taken because the testing cost for the Spike 80DF concentration is prohibitive. Based on the initial sample set, estimation through Kriging interpolation of the % concentration throughout the remaining unsampled locations of the factor space will be conducted. The variance must be less than 0.065 before continued sampling is no longer required. The observations at each location are to be replicated $n_{REP} = 5$ times. In practice, a common method of determining replications is to use confidence intervals with the sample interval defined by [75] as:

$$Pr(\bar{z}_1 - t_{\left(\frac{\alpha}{2}, n_1 - 1\right)} S_p \sqrt{\frac{1}{n_1}} \leq \mu_1 \leq \bar{z}_1 + t_{\left(\frac{\alpha}{2}, n_1 - 1\right)} S_p \sqrt{\frac{1}{n_1}}) = 1 - \alpha$$

where $\bar{z}$ is the response mean, t is the t-distribution with ν degrees of freedom and confidence coefficient $(1 - \alpha)$, and S_p is the pooled sample standard deviation.

A graphical representation of the factor space is shown below:

Figure 9: Factor space - Original DFK Process Example

The boundaries of the problem are $lb = (1, 1)$ and $ub = (3,3)$. The factor space needs to be formally defined for consistency and accuracy. This is a two dimensional grid $(\mathrm{x}_1, \mathrm{x}_2)$ with the following factor space

$$\{x: x \in D\} where\ D\ is\ a\ fixed\ subset\ of\ \mathbb{R}^d\ in\ the\ positive\ dimension\ d$$

$$\mathrm{x}_1 = [1, 2, 3]\ meters$$

$$\mathrm{x}_2 = [1, 2, 3]\ meters$$

$$U = \{s{:}\, s \in \mathrm{x}_1\ and\ s \in \mathrm{x}_2\ \}$$

$$Factor\ Space{:}\ U = \{(1,1), (1,2), (1,3), (2,1), (2,2), (2,3), (3,1), (3,2), (3,3)\}$$

$$where\ \mathcal{B}\ is\ a\ collection\ of\ all\ subsets\ S\ including\ \emptyset\ and\ A_1, A_2, \dots\ \in\ \mathcal{B}$$

$$Boundaries{:}\ (\mathrm{x}_1, \mathrm{x}_2) = [1,1]\ and\ (\mathrm{x}_1, \mathrm{x}_2) = [3,3]$$

$$Dimensions{:}\ d = 2$$

$$Inputs{:}\ \mathrm{x}_1, \mathrm{x}_2$$

$$Observed\ Response{:}\ Z - \%Spike\ 80DF$$

$$Prediction\ Response{:}\ \hat{Z} - \%Spike\ 80DF$$

The lag (or bin) h of the factor space is to be $h = \min\{(\mathrm{x}_{\mathrm{i}+1}, \mathrm{x}_{\mathrm{j}+1})\}$ which is the just minimum unit spacing of the Euclidean differences of U therefore maximizing the number of points to be used in the variogram model. This is shown in step four of the current DFK process. The Euclidean distances for each lag h must be determined. These distances are shown below in the table below.

Table 7: The Complete Lag Matrix - Original DFK Process

		1	1	1	2	2	2	3	3	3	x_1
x_1	x_2	1	2	3	1	2	3	1	2	3	x_2
1	1	0.000	1.000	2.000	1.000	1.414	2.236	2.000	2.236	2.828	
1	2	1.000	0.000	1.000	1.414	1.000	1.414	2.236	2.000	2.236	
1	3	2.000	1.000	0.000	2.236	1.414	1.000	2.828	2.236	2.000	
2	1	1.000	1.414	2.236	0.000	1.000	2.000	1.000	1.414	2.236	
2	2	1.414	1.000	1.414	1.000	0.000	1.000	1.414	1.000	1.414	
2	3	2.236	1.414	1.000	2.000	1.000	0.000	2.236	1.414	1.000	
3	1	2.000	2.236	2.828	1.000	1.414	2.236	0.000	1.000	2.000	
3	2	2.236	2.000	2.236	1.414	1.000	1.414	1.000	0.000	1.000	
3	3	2.828	2.236	2.000	2.236	1.414	1.000	2.000	1.000	0.000	

With the problem defined and the necessary preliminary calculations complete, the process as defined in the literature proceeds to the first step.

4.1 Step 1: The pilot design $\boldsymbol{n_{ir}}$ is selected

For simplicity and as used in the literature, a design is selected which will maximize the minimum distance between two points of the design [53]. Utilizing this type of design leads to an initial pilot design of $n_{ir} = 5$ with $x_i \in \{(1,1),(1,3),(2,2),(3,1),(3,3)\}$ $for\ i = 1\ to\ 5$. A graphical representation of the pilot design is shown below.

Table 8: Pilot Design - Original DFK Process

x_2	**Pilot Design**		
3	x		x
2		x	
1	x		x
$x_1 \rightarrow$	1	2	3

Table 9: Pilot Design ***X*** Matrix of Lags

		1	1	2	3	3	x_1
x_1	x_2	1	3	2	1	3	x_2
1	1	0.000	2.000	1.414	2.000	2.828	
1	3	2.000	0.000	1.414	2.828	2.000	
2	2	1.414	1.414	0.000	1.414	1.414	
3	1	2.000	2.828	1.414	0.000	2.000	
3	3	2.828	2.000	1.414	2.000	0.000	

Now, that the experimental pilot design been determined, the process continues to the second step.

4.2 Step 2: For $(\mathbf{x_i}, \mathbf{x_2})$ gather pilot design response data

Observe $n_{REP} = 6$ independent and identically distributed (IID) replicates for $z_{i,j}$. The observation matrix for this example is generated and is shown below.

$$\begin{bmatrix} z_{11} & z_{12} & z_{13} & z_{14} & z_{15} & z_{16} \\ z_{21} & z_{22} & z_{23} & z_{24} & z_{25} & z_{26} \\ z_{31} & z_{32} & z_{33} & z_{34} & z_{35} & z_{36} \\ z_{41} & z_{42} & z_{43} & z_{44} & z_{45} & z_{46} \\ z_{51} & z_{52} & z_{53} & z_{54} & z_{55} & z_{56} \end{bmatrix}$$

Table 10 shows the response data values for each factor level combination and for each replicate.

Table 10: Pilot Design Initial Response Data - Original DFK Process

		Observed Responses $z(x_1, x_2)$ per replicate					
x_1	x_2	1	2	3	4	5	6
1	1	0.09119	0.16054	0.09678	0.06663	0.1251	0.09119
1	3	0.25698	0.29018	0.29232	0.65918	0.28086	0.20269
2	2	0.26352	0.322	0.35138	0.8048	0.13238	0.19329
3	1	0.39189	0.4025	0.26776	0.55254	0.41169	0.21622
3	3	0.36294	0.42085	0.97557	1.10043	0.21919	0.18774

The sampled observation of this data completes the second step of the DFK process.

4.3 Step 3: Estimate the mean of pilot data observations

The purpose for step three in the original DFK process is to estimate the sample mean of the responses at each factor level combination. The estimate of the mean is shown through

$$\bar{z}(\mathrm{x}_1, \mathrm{x}_2) = \frac{\sum_{j=1}^{n_{REP}} z_{i,j}}{n_{REP}} \; for\; i = (1, 2, \ldots, n_{ir}) at\; FLC_i \in \{(1,1), (1,3), (2,2), (3,1), (3,3)\}\,.$$

By following the above formulation the estimated means were calculated and are shown in the table below.

Table 11: Estimated Average Response Data - Original DFK Process

x_1	x_2	$\bar{z}(\mathrm{x}_1, \mathrm{x}_2)$
1	1	0.106801343
1	3	0.330369242
2	2	0.344561425
3	1	0.373768358
3	3	0.544453992

The process proceeds to step four to determine the Kriging predictions.

4.4 Step 4: Compute Kriging predictions

Based on $\bar{z}(\mathrm{x}_1, \mathrm{x}_2)$ for n_{ir} inputs $(\mathrm{x}_1, \mathrm{x}_2)$ the Kriging predictors are computed for the expected output set n_K^U using inputs x^c for the set $\{x_i^c, x_{i+1}^c, \ldots, x_n^c\}$. The candidate input set is selected using a space-filling method. As described, the candidate inputs are selected halfway between the neighboring FLCs. For this example, $n_K^U = 2$ and $x_1^c = (2,1)$; $x_2^c = (3,2)$. Kriging, which uses a weighted linear combination of all observed outputs, is used to observe the $n_K^U = 2$

unobserved responses. First the experimental variogram must be calculated. This is performed through the following formula.

$$\gamma(h) = \frac{1}{2n_h}\sum_{i=1}^{n_h}(z_n - z_{i+h})^2$$

Performing the $(z_n - z_{i+h})^2$ calculation of the above formula leads to the following Table:

Table 12: Partial Experimental Variogram Calculations - Original DFK Process

Lag (h)	$(z_n - z_{i+h})^2$				$\sum_{i=1}^{n_h}(z_n - z_{i+h})^2$
h_1 =1.41421	0.05653	0.000201	0.000853	0.039957	0.097541 $i = 1\ to$ 4
h_2 =2	0.049983	0.071271	0.045832	0.029134	0.19622 $i = 1\ to$ 4
h_3 =2.82843	0.001883	0.19154	-	-	0.193423 $i = 1\ to$ 2

Finishing the experimental variogram calculation result in the following Table:

Table 13: Experimental Variogram Calculations - Original DFK Process

Gamma	$\frac{1}{2n_h}\sum_{i=1}^{n_h}(z_n - z_{i+h})^2$
γ(1.41421)	$\frac{0.097541}{2(4)} = 0.01219267$
γ(2)	0.024527482
γ(2.82843)	0.048355831

Now that the experimental variogram is estimated, we must now determine which of the four standard functions represents the experimental variogram the best. This is done by using the standard functions described in the literature such as the spherical, exponential, Gaussian, and linear functions. A nugget of $c_0 = 0$ was used. A sill of 0.047 and a range of 2.828 were selected by analyzing the graph of the experimental variogram and determining that autocorrelation was still present at the maximum lag $h = 2.828$. The following Table outlines these calculations along with the sill and range identification.

Table 14: Fitted Variogram Calculations - Original DFK Process

	Sill = 0.047		Range = 2.828	
Gamma	Linear	Sphere	EXPO	Gauss
γ(1.414)	0.023503549	0.032316493	0.036515258	0.024803802
γ(2)	0.033239038	0.041546287	0.041367762	0.036517633
γ(2.828)	0.047007099	0.046999998	0.044661068	0.044662127

A plot of the fitted variogram functions can be seen in the figure below.

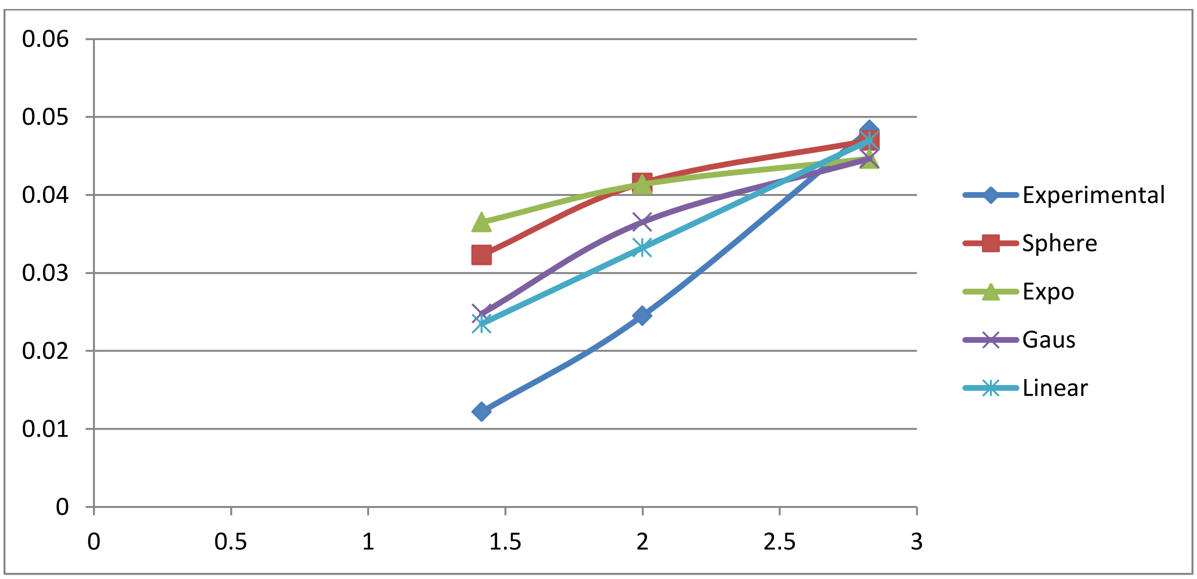


Figure 10: Fitted Variogram - Original DFK Process

To determine the best fit the following least squares selection criteria was used.

$$\min\{\sum_{i=1}^{n_h} d_j{}^2 = (\gamma(h)_i - \gamma(h)_i^f)^2\}$$

where $\gamma(h)_i^f$ is the fitted variogram γ value from Table 13 and ${d_j}^2$ is the square distance between the experimental and fitted variogram γ values. These calculations can be seen in the following Table.

Table 15: Minimum Squared Distance Variogram Calculations - Original DFK Process

Squared Differences					
$\gamma(h)$		Sphere	Expo	Gaussian	Linear
γ(1.414)	${d_j}^2$	0.000404968	0.000591588	0.000159041	0.000127936
γ(2)		0.00028964	0.000283595	0.000143764	7.58912E-05
γ(2.828)		1.83828E-06	1.36513E-05	1.36434E-05	1.81908E-06
	$\sum_{i=1}^{3} {d_j}^2$	0.000696446	0.000888835	0.000316448	0.000205646
	$\min\{\sum_{i=1}^{3} {d_j}^2\}$	0.000205646	=>		Linear

Now that it has been determined that the linear variogram should be used, the Kriging calculations can be conducted. This begins by constructing the fitted variogram matrix $\boldsymbol{C}$ from $\boldsymbol{X}$ using the linear function. This results in the following Table.

Table 16: $\boldsymbol{C}$ Matrix - Original DFK Process

$\boldsymbol{C}$ Matrix with LaGrangian Multiplier					
0.0000	0.0332	0.0235	0.0332	0.0470	1
0.0332	0.0000	0.0235	0.0470	0.0332	1
0.0235	0.0235	0.0000	0.0235	0.0235	1
0.0332	0.0470	0.0235	0.0000	0.0332	1
0.0470	0.0332	0.0235	0.0332	0.0000	1
0.0000	0.0332	0.0235	0.0332	0.0470	1
1	1	1	1	1	0

In addition $\boldsymbol{c}$ must be found for $x_1^c = (2,1)$ and $x_1^c = (3,2)$. These calculations are completed by using the same approach as to calculate $\boldsymbol{C}$.

Table 17: $\boldsymbol{c}$ Matrix - Original DFK Process

x_1	2	3	$\boldsymbol{c}$ for $x_1^c = (2,1)$ and $x_2^c = (3,2)$	
x_2	1	2		
$h =$	1	2.236068	0.0166	0.0372
	2.236068	2.236068	0.0372	0.0372
	1	1	0.0166	0.0166
	1	1	0.0166	0.0166
	2.236068	1	0.0372	0.0166
	-	-	1	1

Solving $\boldsymbol{C\Lambda} = \boldsymbol{c}$ for $\boldsymbol{\Lambda}$, $x_1^c = (2,1)$ and $x_2^c = (3,2)$ results in the following Table.

Table 18: $\boldsymbol{\Lambda}$ Vector - Original DFK Process

$\boldsymbol{\lambda}$ for $x_1^c = (2,1)$	$\boldsymbol{\lambda}$ for $x_2^c = (3,2)$
0.396017	-0.041
-0.041	-0.041
0.289962	0.289962
0.396017	0.396017
-0.041	0.396017
-8.3E-05	-8.3E-05

Finally the Kriging estimators for $x_1^c = (2,1)$ and $x_2^c = (3,2)$ are calculated as $\hat{Z}\left(x_{0_i{}^c}\right) = \boldsymbol{Z^T\lambda}$.

The Kriging estimates are shown in the table below.

Table 19: Kriging Predictors - Original DFK Process

$\hat{Z}$ for $x_1^c = (2, 1)$	$\hat{Z}$ for $x_2^c = (3, 2)$
0.254357	0.445619

This concludes step four of the original DFK process. Now the process continues to step five out of the eight total steps.

4.5 Step 5: Perform non-parametric bootstrapping per input based on $\boldsymbol{z_i}(\mathbf{x_1}, \mathbf{x_2})$ for $\boldsymbol{n_{ir}}$

The responses from the first three steps in the process will be resampled with replacement to estimate the Kriging variance when response data are known with all bootstrapped observations denoted with a superscript * [76]. The reader should note that this example does not contain enough data to perform bootstrapping in the correct statistical sense. The use of bootstrapping here is completed for illustration of the current DFK process only. Performing bootstrapping generates a response dataset of

$$\{(x_{i:1}, z_{m:1}^*), \ldots, (x_{i:n_{ir}}, z_{m:n_{ir}}^*)\} \text{ for } (b = 1, 2, 3, \ldots, B)$$

where n_{ir} is the initial sample size and m is the bootstrap sample size. The averages are computed again for m number of resamples at each B replicate. The original DFK literature uses $m = 10$, which is used in this example. The literature suggests that B range from 25 to 200, thus this example uses 25 for illustration purposes [76], [3]. The partial bootstrapped replications are shown in the table below. Please refer to Appendix B: Remaining Bootstrapped Data for Original DFK Process for the omitted data.

Table 20: Partial Bootstrap Data - Original DFK Process

Bootstrap Replicate 1										$\bar{z}^*_{i:1}$
0.067	0.067	0.125	0.067	0.067	0.091	0.091	0.067	0.091	0.091	0.082
0.659	0.257	0.281	0.290	0.281	0.257	0.659	0.659	0.257	0.281	0.388
0.322	0.322	0.132	0.193	0.805	0.805	0.193	0.322	0.322	0.805	0.422
0.268	0.216	0.268	0.268	0.553	0.392	0.268	0.403	0.553	0.412	0.360
0.421	1.100	0.363	0.976	1.100	0.421	0.219	0.188	0.219	0.363	0.537

.

.

.

Bootstrap Replicate 25										$\bar{z}^*_{i:25}$
0.091	0.091	0.091	0.067	0.091	0.097	0.125	0.161	0.161	0.091	0.107
0.257	0.290	0.203	0.290	0.203	0.203	0.290	0.290	0.257	0.290	0.257
0.132	0.322	0.322	0.805	0.132	0.322	0.132	0.132	0.805	0.351	0.346
0.392	0.412	0.268	0.392	0.216	0.392	0.392	0.392	0.553	0.216	0.362
0.188	1.100	0.363	1.100	0.976	0.976	0.421	1.100	0.976	0.421	0.762

For $\bar{z}^*_{b:B}$ with $(b = 1, 2, 3, \dots, B)$ the bootstrapped Kriging predictor is calculated for each n_{ir} of $(x_1,\ x_2)$. The table below shows the list of Kriging predictors.

$$\hat{Z}(x_i^c)^* = \sum_{i=1}^{m} \lambda^* \bar{z}^*(x_i)\ \ for\ (b = 1, 2, 3, \dots, B)$$

Table 21: Bootstrapped Kriging Predictors - Original DFK Process

B	$\hat{Z}(2,1)^*$	$\hat{Z}(3,2)^*$
1	0.25957	0.45829
2	0.26962	0.43914
3	0.23993	0.39593
4	0.27430	0.46518
5	0.27681	0.51430
6	0.25528	0.45402
7	0.24318	0.45435
8	0.21866	0.46675
9	0.25237	0.43650
10	0.28397	0.45508
11	0.22468	0.42190
12	0.26024	0.45855
13	0.21054	0.47351
14	0.23881	0.37866
15	0.23796	0.51686
16	0.27062	0.47716
17	0.23588	0.38166
18	0.27018	0.44092
19	0.27309	0.44625
20	0.24346	0.50728
21	0.25143	0.44396
22	0.24521	0.40514
23	0.23509	0.36838
24	0.23464	0.45664
25	0.24414	0.53060

This concludes step five of the process. The process proceeds to step six for variance calculations.

4.6 Step 6: Calculate the Kriging variance for candidate inputs

For each x_i^c with sample size n_K^U the Kriging variance is to be calculated by applying the following formula.

$$\hat{V}(\hat{Z}(x_b^c)^* = \frac{1}{B-1}\sum_{b=1}^{B}(\hat{Z}(x_b^c)^* - \sum_{b=1}^{B}\frac{\hat{Z}(x_b^c)^*}{B})^2$$

The calculations which are shown in the table below conclude step six of the process.

Table 22: Bootstrapped Kriging Variance Estimates - Original DFK Process

$\hat{V}(\hat{Z}(2,1)^*)$	$\hat{V}(\hat{Z}(3,2)^*)$
0.060332175	0.195948707

It is determined from the table above that all of the variances do not meet the stopping criterion of less than 0.065. Therefore we proceed to step seven of the process.

4.7 Step 7: Determine the input with the largest variance

The FLC with the largest estimated variance as a result of bootstrapping is added to the set $\{x_i^c, x_{i+1}^c, \ldots, x_n^c\}$. To formally choose the maximum input, it is defined with the following argument.

$$\max\{\hat{V}(x_i^c)\}$$

The result from this step of the process is to add $x_2^c = (3,2)$ to original set of selected 5 FLCs, and observe the responses per replicates as before to obtain the mean response $\bar{z}(x_{(3,2)}^c)$.

4.8 Step 8: Repeat steps 4-7 until the stopping criteria has been reached

The problem statement requires the variance be less than 0.065 so that steps four through seven are repeated. The updated design is shown in the table below.

Table 23: Experimental Design Iteration 1 - Original DFK Process

x_2	Experimental Design Iteration 1		
3	x		x
2		x	**x**
1	x		x
$x_1 \rightarrow$	1	2	3

The sampled response data now contain the $z_{3,2}$ replication data along with $\bar{z}(3,2)$. The set $\{x_i^c, x_{i+1}^c, \dots, x_n^c\}$ for n_K^U is determined as before.

Table 24: Pilot Design Initial Response Data - Original DFK Process

		Observed Responses $z(x_1, x_2)$ per replicate					
x_1	x_2	1	2	3	4	5	6
1	1	0.09119	0.16054	0.09678	0.06663	0.1251	0.09119
1	3	0.25698	0.29018	0.29232	0.65918	0.28086	0.20269
2	2	0.26352	0.322	0.35138	0.8048	0.13238	0.19329
3	1	0.39189	0.4025	0.26776	0.55254	0.41169	0.21622
3	**2**	**0.56767**	**0.40032**	**0.68785**	**0.24624**	**0.40058**	**0.21369**
3	3	0.36294	0.42085	0.97557	1.10043	0.21919	0.18774

Table 25: Estimated Average Response Data Iteration 1 - Original DFK Process

(x_1, x_2)		$\bar{z}(x_1, x_2)$
1	1	0.106801343
1	3	0.330369242
2	2	0.344561425
3	1	0.373768358
3	**2**	**0.419392835**
3	3	0.544453992

Table 26: Fitted Variogram Calculations Iteration 1 - Original DFK Process

Squared Differences					
$\gamma(h)$		Sphere	Expo	Gaussian	Linear
γ(1)	$d_j^{\ 2}$	0.000400132	0.000720546	0.000116945	0.000162119
γ(1.414)		0.000404968	0.000591588	0.000159041	0.000127936
γ(2)		0.00028964	0.000283595	0.000143764	7.58912E-05
γ(2.236)		0.000313897	0.000262628	0.000179205	0.000115621
γ(2.828)		1.83828E-06	1.36513E-05	1.36434E-05	1.81908E-06
	$\sum_{i=1}^{3} d_j^{\ 2}$	0.00109474	0.00159573	0.00041975	0.000365946
	$\min\{\sum_{i=1}^{3} d_j^{\ 2}\}$	0.000365946	=>		Linear

Table 27: Kriging Predictors Iteration 1 - Original DFK Process

$\hat{Z}$ for $x_1^c = (2,1)$	$\hat{Z}$ for $x_2^c = (1,2)$
0.253757162	0.236128133

Table 28: Updated Variance Estimates Iteration 1 - Original DFK Process

$\hat{V}(\hat{Z}(2,1)^*)$	$\hat{V}(\hat{Z}(1,2)^*)$
0.06427691	0.071514567

Our process has reached the precision as stated in the problem statement of a Kriging variance less than 0.065. The process is now concluded.

4.9 Summary of the Current DFK Process

The current DFK process consists of eight steps all with the intent to perform incremental interpolation based on the identification of the factor space and stopping criteria. As outlined very methodically, one can see that the process although novel, has many potential improvements and applications. This research proposes improvements in the current DFK process and extends this powerful nonlinear interpolation method into areas of applications such as experiment planning and simulation validation. This is accomplished by incorporating cost considerations along with an augmented space filling design to provide unique interpolation solutions based on dimensionality d through a standardized covariance algorithm. The process as a whole is optimized by utilizing budget constraints to reduce the number of iterations as this becomes critical as the number of FLCs increase. Unless it is cost prohibitive, a one-by-one sequential sampling becomes unrealistic in high dimensional environments as the number of factor level combinations increase dramatically.

CHAPTER 5

Advancements in the Design for Kriging Process

This chapter focuses on specific improvements in the DFK process that were outlined in Chapter 4. It discusses detailed mathematical developments, assumptions, and examples that we present in order to advance the DFK process and to utilize the process in test planning and validation type environments.

5.1 Estimation Variance with No Empirical Data

One goal in this research is to derive a method to estimate interpolation variance prior to obtaining response data. The purpose is to utilize this information in order to determine through some general heuristics an initial pilot design including sample size and FLC selection. The interpolation variance allows for an optimized determination of initial sample sizes for use in the experimentation process. Literature describes taking some number of pilot samples, computing a variogram and then estimating a fitted function. The sample size selection described in the literature varies greatly. The methods described in the literature are not presented in an effective manner to reduce total iterations in a DOX environment.

To calculate estimation variance with no empirical data, three assumptions are made. The first assumption is that inputs or dimensions are equally spaced integer values or can be represented as such through a data transformation. This allows for automatic and computationally efficient generation of factor spaces without a user requirement to input all

FLCs individually. The second assumption is that the factor space for each input or dimension is bounded by the same values including minimum, maximum, and incremental or separation distance, i.e., a block design. These designs are common in DOX literature. Further research should be conducted to relax each of these assumptions. The third assumption is that data are normally distributed. If required a test for normality can be conducted or the central limit theorem may be assumed if the sample size is sufficiently large.

We must first develop a mathematical relationship between the commonly used variogram fitted functions such that the estimation variance calculations are completely independent of the sample data. This is reasonable since that the Kriging weights are based on spatial separation of the input data and these Kriging weights determine the Kriging model. Our methodology utilizes the spherical, exponential, Gaussian, and linear variogram functions as these are the most commonly used functions as described in the literature to represent experimental variograms. To develop an algorithm to estimate initial variances without response data, the fitted functions must be related. We reiterate the functions of interest in the table below.

Table 29: Variogram Fitted Functions of Interest

	$\gamma(h) = c_0 +$
Linear	$c_1\left(\frac{h}{a}\right) if\ h \leq a;\ c_1\ otherwise$
Spherical	$c_1\left(1.5\left(\frac{h}{a}\right) - 0.5\left(\frac{h}{a}\right)^3\right)\ if\ h \leq a;\ c_1\ otherwise$
Exponential	$c_1\left(1 - exp\left(\frac{-3h}{a}\right)\right)$
Gaussian	$c_1\left(1 - exp\left(\frac{-3h^2}{a^2}\right)\right)$

One can see that c_0 and c_1 are simply linear and scalar factors of the above four equations assuming that $h \leq a$. The $h \leq a$ assumption holds with a good neighborhood selection as autocorrelation will be present at the maximum lag h. Replacing $\frac{h}{a}$ with φ and rewriting each equation leads to the table below.

Table 30: Reduced Variogram Fitted Functions of Interest

	$\gamma(h) =$
Linear	$c_1\varphi$
Spherical	$c_1(1.5\varphi - 0.5(\varphi)^3)$
Exponential	$c_1\big(1 - exp(-3\varphi)\big)$
Gaussian	$c_1\big(1 - exp(-3\varphi^2)\big)$

Since the linear fitted variogram model is equal to φ, it is clear to see now that each function is directly related to the linear variogram. This simplification carries over into the Kriging weight calculation of $\mathbf{\Lambda}$. The result is four systems of equations that are directly related. The value of n used below varies depending on the iteration of DFK process. For example during the pilot design phase $n = n_{ir}$. $n = n_{ir} = 5$ during the pilot design phase of the example presented in Chapter 4. After selecting $n_K^U = 2$, we added the candidate input with the highest estimation variance to the design, thus $n = 6$. The notation n is generically presented here as the number of FLCs in the design and varies throughout the DFK iteration process.

$$Linear\text{:} \begin{pmatrix} c_1\varphi(x_1 - x_1) & \cdots & c_1\varphi(x_1 - x_n) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ c_1\varphi(x_n - x_1) & \cdots & c_1\varphi(x_n - x_n) & 1 \\ 1 & \cdots & 1 & 0 \end{pmatrix} \begin{pmatrix} \lambda_1^{OK} \\ \vdots \\ \lambda_n^{OK} \\ \omega_{OK} \end{pmatrix} = \begin{pmatrix} c_1\varphi(x_1 - x_0) \\ \vdots \\ c_1\varphi(x_n - x_0) \\ 1 \end{pmatrix}$$

$$Sphereical: \begin{pmatrix} c_1(1.5\varphi - 0.5(\varphi)^3)(x_1 - x_1) & \cdots & c_1(1.5\varphi - 0.5(\varphi)^3)(x_1 - x_n) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ c_1(1.5\varphi - 0.5(\varphi)^3)(x_n - x_1) & \cdots & c_1(1.5\varphi - 0.5(\varphi)^3)(x_n - x_n) & 1 \\ 1 & \cdots & 1 & 0 \end{pmatrix} \begin{pmatrix} \lambda_1^{OK} \\ \vdots \\ \lambda_n^{OK} \\ \omega_{OK} \end{pmatrix}$$

$$= \begin{pmatrix} c_1(1.5\varphi - 0.5(\varphi)^3)(x_1 - x_0) \\ \vdots \\ c_1(1.5\varphi - 0.5(\varphi)^3)(x_n - x_0) \\ 1 \end{pmatrix}$$

$$Exponential: \begin{pmatrix} c_1\big(1 - exp(-3\varphi)\big)(x_1 - x_1) & \cdots & c_1\big(1 - exp(-3\varphi)\big)(x_1 - x_n) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ c_1\big(1 - exp(-3\varphi)\big)(x_n - x_1) & \cdots & c_1\big(1 - exp(-3\varphi)\big)(x_n - x_n) & 1 \\ 1 & \cdots & 1 & 0 \end{pmatrix} \begin{pmatrix} \lambda_1^{OK} \\ \vdots \\ \lambda_n^{OK} \\ \omega_{OK} \end{pmatrix}$$

$$= \begin{pmatrix} c_1\big(1 - exp(-3\varphi)\big)(x_1 - x_0) \\ \vdots \\ c_1\big(1 - exp(-3\varphi)\big)(x_n - x_0) \\ 1 \end{pmatrix}$$

$$Gaussian: \begin{pmatrix} c_1\big(1 - exp(-3\varphi^2)\big)(x_1 - x_1) & \cdots & c_1\big(1 - exp(-3\varphi^2)\big)(x_1 - x_n) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ c_1\big(1 - exp(-3\varphi^2)\big)(x_n - x_1) & \cdots & c_1\big(1 - exp(-3\varphi^2)\big)(x_n - x_n) & 1 \\ 1 & \cdots & 1 & 0 \end{pmatrix} \begin{pmatrix} \lambda_1^{OK} \\ \vdots \\ \lambda_n^{OK} \\ \omega_{OK} \end{pmatrix}$$

$$= \begin{pmatrix} c_1\big(1 - exp(-3\varphi^2)\big)(x_1 - x_0) \\ \vdots \\ c_1\big(1 - exp(-3\varphi^2)\big)(x_n - x_0) \\ 1 \end{pmatrix}$$

From $\boldsymbol{\Lambda}$, the relationship carries furthermore into variance calculations. Each variance carries a direct relationship that depends on the ratio of lag to range $\frac{h}{a}$.

$$\hat{\sigma}_L^2 = \prod (\boldsymbol{\lambda_L})(\boldsymbol{c_L})$$

$$\hat{\sigma}_S^2 = \prod (\boldsymbol{\lambda_S})(\boldsymbol{c_S})$$

$$\hat{\sigma}_E^2 = \prod(\boldsymbol{\lambda_E})(\boldsymbol{c_E})$$

$$\hat{\sigma}_G^2 = \prod(\boldsymbol{\lambda_G})(\boldsymbol{c_G})$$

Demonstrating the relationship alone, doesn't allow for selection of a fitted variogram function without knowledge of the empirical variogram. To address this issue, we look at descriptive statistical information in order to present the user of the advanced DFK software with worst case initial assessments of the estimation variance in order for determining initial sample size and FLC selection. To demonstrate the method above, we look at the following datasets, graphs, and associated descriptive statistical information. The following sets of lags are given for a one dimensional or single input sample space.

Table 31: Initial $\mathbf{x_1}$ Set

x_1 values									
1	2	3	4	5	6	7	8	9	10

Four tests were carried out to demonstrate the relationship between the variogram models:

1. Alter the range (a), while holding c_0, and c_1 constant
2. Alter the sill (c_1), while holding c_0, and a constant
3. Alter the nugget (c_0), while holding c_1, and a constant
4. Alter φ and analyze absolute and percentage difference between each variogram

The formulations of each variogram using Table 30 were computed for given lags and variograms are plotted for tests one through three based on $\{h: h = 1, 2, \dots, 9\}$. The results are shown in the variogram tables and plots that follow below for test one. Test two and three data can be found in the appendix.

Table 32: Linear Variogram $\boldsymbol{c_0 = 0}$

c_1	a	$h = 1$	2	3	4	5	6	7	8	9
1	1	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	2	0.5000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	3	0.3333	0.6667	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	4	0.2500	0.5000	0.7500	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	5	0.2000	0.4000	0.6000	0.8000	1.0000	1.0000	1.0000	1.0000	1.0000
1	6	0.1667	0.3333	0.5000	0.6667	0.8333	1.0000	1.0000	1.0000	1.0000
1	7	0.1429	0.2857	0.4286	0.5714	0.7143	0.8571	1.0000	1.0000	1.0000
1	8	0.1250	0.2500	0.3750	0.5000	0.6250	0.7500	0.8750	1.0000	1.0000
1	9	0.1111	0.2222	0.3333	0.4444	0.5556	0.6667	0.7778	0.8889	1.0000

Table 33: Spherical Variogram $\boldsymbol{c_0 = 0}$

c_1	a	$h = 1$	2	3	4	5	6	7	8	9
1	1	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	2	0.6875	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	3	0.4815	0.8519	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	4	0.3672	0.6875	0.9141	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	5	0.2960	0.5680	0.7920	0.9440	1.0000	1.0000	1.0000	1.0000	1.0000
1	6	0.2477	0.4815	0.6875	0.8519	0.9606	1.0000	1.0000	1.0000	1.0000
1	7	0.2128	0.4169	0.6035	0.7638	0.8892	0.9708	1.0000	1.0000	1.0000
1	8	0.1865	0.3672	0.5361	0.6875	0.8154	0.9141	0.9775	1.0000	1.0000
1	9	0.1660	0.3278	0.4815	0.6228	0.7476	0.8519	0.9314	0.9822	1.0000

Table 34: Exponential Variogram $\boldsymbol{c_0 = 0}$

c_1	a	$h = 1$	2	3	4	5	6	7	8	9
1	1	0.9502	0.9975	0.9999	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	2	0.7769	0.9502	0.9889	0.9975	0.9994	0.9999	1.0000	1.0000	1.0000
1	3	0.6321	0.8647	0.9502	0.9817	0.9933	0.9975	0.9991	0.9997	0.9999
1	4	0.5276	0.7769	0.8946	0.9502	0.9765	0.9889	0.9948	0.9975	0.9988
1	5	0.4512	0.6988	0.8347	0.9093	0.9502	0.9727	0.9850	0.9918	0.9955
1	6	0.3935	0.6321	0.7769	0.8647	0.9179	0.9502	0.9698	0.9817	0.9889
1	7	0.3486	0.5756	0.7235	0.8199	0.8827	0.9236	0.9502	0.9676	0.9789
1	8	0.3127	0.5276	0.6753	0.7769	0.8466	0.8946	0.9276	0.9502	0.9658
1	9	0.2835	0.4866	0.6321	0.7364	0.8111	0.8647	0.9030	0.9305	0.9502

Table 35: Gaussian Variogram $\boldsymbol{c_0 = 0}$

c_1	a	$h = 1$	2	3	4	5	6	7	8	9
1	1	0.9502	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	2	0.5276	0.9502	0.9988	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1	3	0.2835	0.7364	0.9502	0.9952	0.9998	1.0000	1.0000	1.0000	1.0000
1	4	0.1710	0.5276	0.8150	0.9502	0.9908	0.9988	0.9999	1.0000	1.0000
1	5	0.1131	0.3812	0.6604	0.8534	0.9502	0.9867	0.9972	0.9995	0.9999
1	6	0.0800	0.2835	0.5276	0.7364	0.8755	0.9502	0.9831	0.9952	0.9988
1	7	0.0594	0.2172	0.4236	0.6245	0.7836	0.8896	0.9502	0.9801	0.9930
1	8	0.0458	0.1710	0.3442	0.5276	0.6902	0.8150	0.8994	0.9502	0.9776
1	9	0.0364	0.1377	0.2835	0.4471	0.6038	0.7364	0.8371	0.9066	0.9502

The relationship from the four tables above becomes clear in the four graphs below.

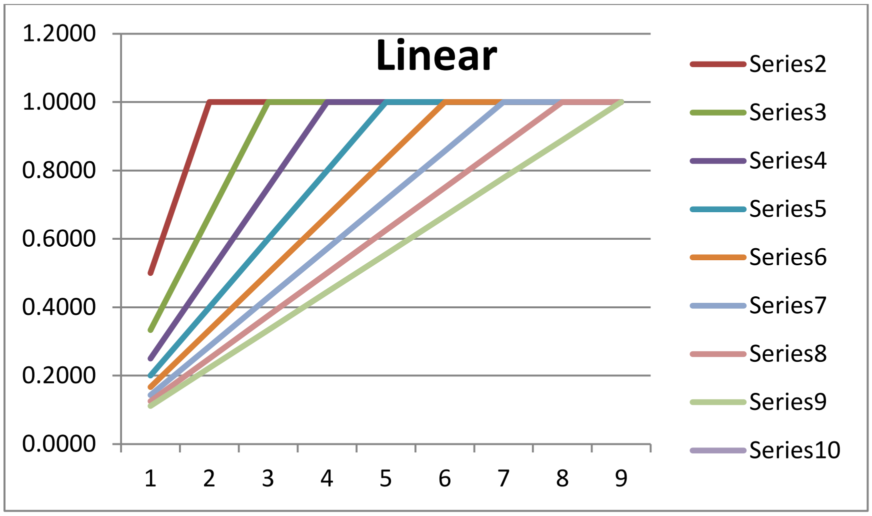


Figure 11: Linear Variogram with Varying Range

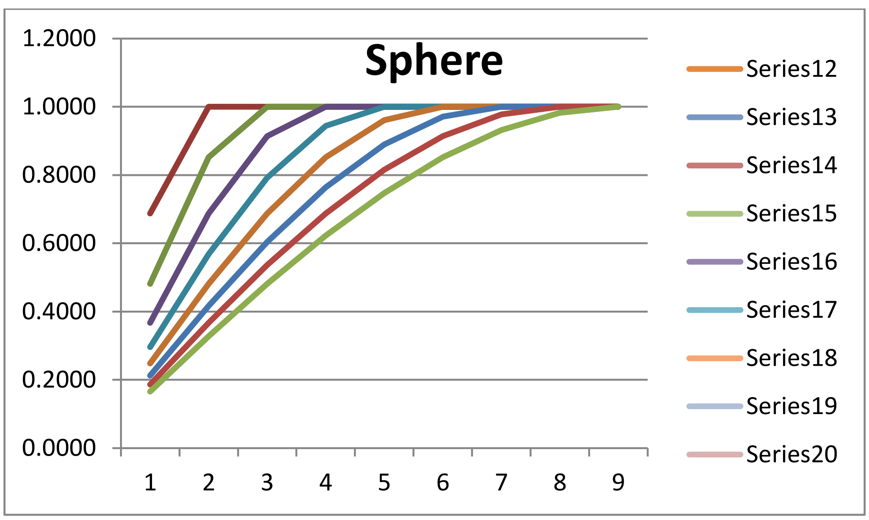


Figure 12: Spherical Variogram with Varying Range

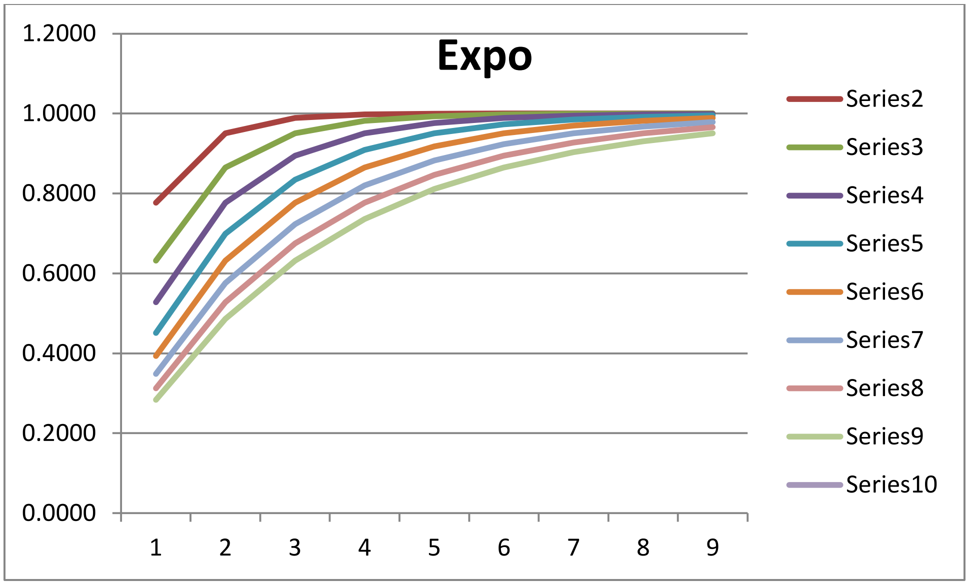


Figure 13: Exponential Variogram with Varying Range

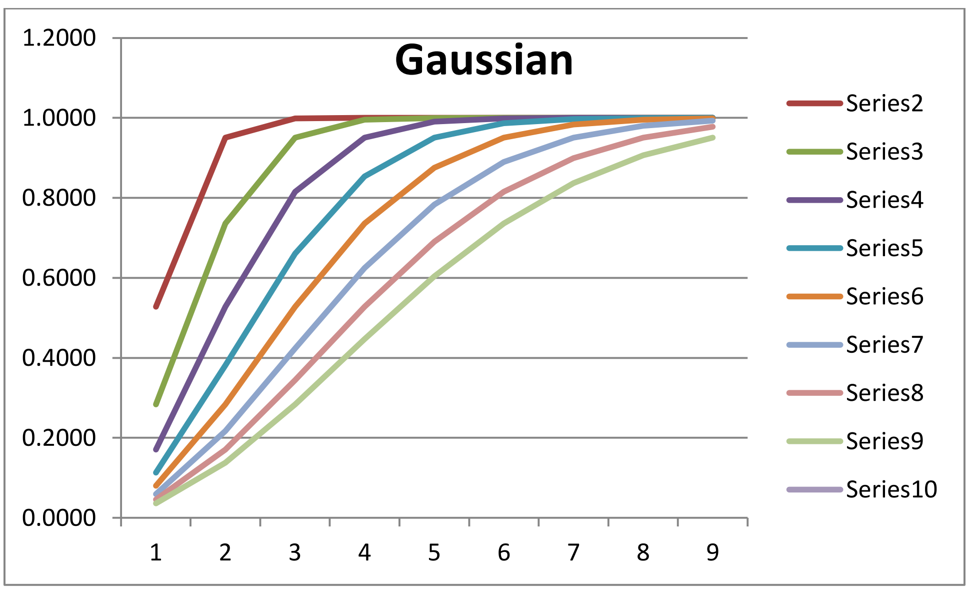


Figure 14: Gaussian Variogram with Varying Range

Each of the above four fitted variogram functions simply display a shift in the curves as the range increases up to the maximum lag value. The appendix shows information in the same format for varying the sill and nugget as described in test two and three. More interestingly, is how the estimation variance is related between the four fitted variogram functions.

The fourth test evaluated how the absolute difference and percent difference changed between each variogram model as the nugget c_0 and the sill c_1 vary. It was found that the absolute difference in the variance estimates did not change as the nugget value varied as expected. In addition, it was found that the percentage difference in the variance estimates did not change as the sill varied. This reiterates the relationship formulations presented earlier. The pertinent information is shown in the tables below. The tables were below were constructed by the following steps.

Step 1) Create a pilot design of FLCs in the pertinent factor space

Step 2) Calculate the lag matrix $\boldsymbol{X}$ and the RHS lag vector $\mathbf{x}$ for the pilot design of each candidate FLC

Step 3) Use the results from step 2 to obtain $\boldsymbol{C}$ and $\boldsymbol{c}$

Step 4) Obtain $\boldsymbol{\lambda} = \boldsymbol{C}^{-1}\boldsymbol{c}$ to calculate the prediction variance at each candidate FLC

Step 5) Repeat steps 1-4 adding the FLC with the highest prediction variance until only one unsampled FLC remains

Step 6) While conducting steps 1-5, alter c_0 from $\{c_0 : c_0 = 0,1,\ldots,9\}$ during each iteration

The columns in Table 36 represent the differences in variances for the fitted variogram functions listed in the header-row, while the rows represent iterations of the steps discussed above. The columns in Table 37 represent the percent differences in variances for the fitted variogram functions listed in the header, while the rows represent iterations of the steps discussed above.

Table 36: Varying Nugget – Absolute Variance Estimation Difference

Spherical vs. Linear Absolute Difference $\{c_0 : c_0 = 0,1,\ldots,9\}$							
0.306429	0.639306	0.935541	1.106539	1.106539	0.935541	0.639306	0.306429
0.451834	0.910264	1.258831	1.425296	1.385754	1.165628	0.837552	
0.430943	0.830027	1.092439	1.165212	1.049539	0.799686		
0.40978	0.750073	0.930732	0.922292	0.756282			
0.388195	0.670094	0.774729	0.703147				
0.366372	0.591838	0.632645					
0.346939	0.52865						
0.350336							

Spherical vs. Exponential Absolute Difference $\{c_0 : c_0 = 0,1,\ldots,9\}$							
2.313099	3.039183	3.124317	3.060789	3.060789	3.124317	3.039183	2.313099
2.307736	3.056868	3.204878	3.209759	3.206988	3.059568	2.309114	
2.305049	3.07687	3.26705	3.268351	3.079581	2.306676		
2.300171	3.078186	3.249915	3.07988	2.301494			
2.283753	3.016416	3.017045	2.284591				
2.234084	2.784468	2.234477					
2.093243	2.093349						
1.672668							

Spherical vs. Gamma Absolute Difference $\{c_0 : c_0 = 0,1,\ldots,9\}$							
1.533124	1.595178	0.916754	0.31965	0.31965	0.916754	1.595178	1.533124
2.064983	3.025081	2.768716	2.044696	1.698081	1.790431	1.543276	
2.110336	3.399144	3.624909	3.0996	2.435846	1.658772		
2.056569	3.289461	3.550938	2.975659	1.844279			
1.965832	2.966895	2.920537	1.902855				
1.832771	2.454796	1.82443					
1.619611	1.61906						
1.207473							

Table 37: Varying Sill - Percent Variance Estimation Difference

Spherical vs. Linear % Difference $\{c_1: c_1 = 1,2,\ldots,9\}$						
13.6559	15.6815	17.4118	18.3079	18.3079	17.4118	15.6815
20.5962	23.4725	25.5018	26.9113	28.0850	30.0784	38.2027
20.1880	22.7727	24.6836	26.3356	28.8166	37.4908	
19.8734	22.3318	24.4910	27.4731	36.7015		
19.7340	22.4421	25.9518	35.7598			
19.9871	24.0731	34.5209				
21.4204						

Spherical vs. Exponential % Difference $\{c_1: c_1 = 1,2,\ldots,9\}$						
103.082558	74.54815	58.14802	50.64145	50.64145	58.14802	74.54815
105.195019	78.82576	64.92535	60.60421	64.99587	78.95038	105.324
107.98273	84.41712	73.81874	73.86986	84.55437	108.1413	
111.552974	91.64626	85.51739	91.74298	111.6889		
116.094957	101.0226	101.0649	116.187			
121.878713	113.2586	121.9263				
129.23916	129.2542					
138.525575						

Spherical vs. Gamma % Difference $\{c_1: c_1 = 1,2,\ldots,9\}$						
68.3232161	39.12814	17.06211	5.288677	5.288677	17.06211	39.12814
94.1294465	78.0061	56.08945	38.60638	34.41493	46.20104	70.3924
98.8611404	93.25903	81.90454	70.05582	66.8797	77.76635	
99.7388627	97.93651	93.43844	88.63846	89.50075		
99.9333554	99.36407	97.83207	96.77315			
99.9853926	99.84915	99.55171				
99.9966018	99.96913					
99.9995129						

As a result of these tests, the differences in the relationship of the variogram functions are reduced to the ratio $\varphi = \frac{h}{a}$. As stated earlier, our defined factor spaces are limited to a single neighborhood. Since a single neighborhood is used, we are under the initial assumption that the lag distances still have some influence on the point being estimated. Under this assumption, we set the initial range, a, to be the maximum lag distance. This allows for the ratio φ to range from $(0,1]$. An iterative computational model was developed for the four fitted variogram functions to determine which model produces the highest estimation variance, therefore which model should be used for sample size selection. The estimation points were taken at all locations that were not part of the evolving input matrix. The maximum or worst variance was taken following the iteration of the model. The results are shown in the table below.

Table 38: Variogram Testing for Initial Selection Summary

	$a = 10$								
	$c_1 = 1$	$c_1 = 2$	$c_1 = 3$	$c_1 = 4$	$c_1 = 5$	$c_1 = 6$	$c_1 = 7$	$c_1 = 8$	$c_1 = 9$
L	0.444	0.889	1.333	1.778	2.222	2.667	3.111	3.556	4.000
S	0.756	1.511	2.267	3.022	3.778	4.533	5.289	6.044	6.800
E	1.006	2.012	3.018	4.024	5.030	6.036	7.042	8.048	9.054
G	0.441	0.882	1.323	1.764	2.206	2.647	3.088	3.529	3.970

As shown from Table 38, we select the exponential as the initial fitted variogram as it provides the highest estimation variances out of the four common fitted variogram functions that were chosen. In order to further validate the exponential selection, a simple 3 FLC MC model was developed in Microsoft Excel®. The nugget was set to zero in the MC model since we have no direct evidence of micro-variability or measurement based error prior to data sampling. The MC model was executed with 100 replications by varying $h, c_1, and\ a$ from

$RAND\{Min\colon 1\,; Max\colon 100\}$ where $a \leq h$ and $FLC_1 < FLC_2 < FLC_3$. The exponential model resulted in the highest estimation variance in all replications.

This allows us to choose the exponential fitted variogram model for test planning purposes. It is important to discuss the parameter values for $c_0, c_1, and\ a$. Recall the assumption that the data are normal. Under this assumption, the response data, although unknown but normal, can be actually standardized according to

$$\mathrm{Z} = \frac{X - \mu}{\sigma}$$

However, in OK μ and σ have to be estimated as their values are unknown. It can be shown that the random variable $\frac{X-\mu}{S}$ has a Student-t distribution with $(n–1)$ degrees of freedom, and hence we make the approximation

$$t_\upsilon \cong \frac{X - \bar{x}}{S}$$

Since the standard normal variance is one, we therefore set the sill, which represents data variability, also equal to one. Recall that the nugget (c_0) is assumed to be zero and that the range (a) equal to $\max\{h_i\}$ due to the single neighborhood selection. Utilizing these parameter settings allows for generic block pilot designs of n dimensions to be produced with no knowledge of the response data as described in Section 5.3. For reiteration, the reader should recall that the Kriging model is a linear weighting method where the weights are based solely on separation distance or lag h. This along with the single neighborhood assumption allows for these parameter setting and fitted variogram selection through

$$\hat{\sigma}_E^2 = \prod(\boldsymbol{\lambda_E})(\boldsymbol{c_E})$$

Examples of constructing these designs and their feasibility are discussed in the next chapter.

5.2 Initial $\boldsymbol{n_{ir}}$ Size Based on Estimation Variance, Expected Improvement, and Budget

Initial sample size recommendations are displayed in the list box on the test planning module of the software application. The expected improvement is considered as the percent reduction in variance as n_{ir} increases. The calculation is based on an incremental variance ratio summarized by

$$EI_i = \frac{\hat{\mathrm{V}}_i}{\sum_{i=1}^{N} \hat{\mathrm{V}}_i} * 100$$

Specifically, as the number of FLCs are incremented an improvement (estimated variance reduction) is realized and the amount of that improvement is based on the FLC that was chosen to be incorporated into the sequential design during the ith iteration. $\hat{V}_i$ is divided by total amount of variance contained within $\sum_{i=1}^{N} \hat{\mathrm{V}}_i$. The amount given in the list box is the percentage of total variance that was reduced by sampling at FLC_i of the current $\{\mathrm{x_i}, \mathrm{x_{i+1}}, \ldots, \mathrm{x_d}\}$. As one would expect, the total amount reduced by the addition of the all FLCs of the factor space is 100%. The incremental cost is calculated in one of two methods. The first method is when no linear scale is incorporated into the pilot design. This method states that cost per sample remains the same. Under this method the budget remaining column in the left list box of the test planning portion of the software and is calculated by:

$$Budget\ Remaning = Total\ Budget - \sum_{i=1}^{N} cost\ per\ test$$

If a linear cost scale is added then the equation becomes:

$$Budget\ Remaning = Total\ Budget - cost\ per\ test_1 - \Psi \sum_{i=2}^{N} cost\ per\ test_{i-1}$$

The addition of the linear scale can be positive or negative. This allows for users of DFK software to adjust the cost per test as the cost to obtain each sample increases or decreases. Unless specified, the software continues to calculate the pilot design after the budget is exhausted in order to give what-if analysis capability. This can be changed by users of DFK software by selecting to stop the program when the budget reaches zero.

For further clarification on how these calculations are applied, the reader should refer to Section 7.3. The software application continues to calculate estimation variances as n_{ir} is incremented until the upper bound of the factor space N is reached. The information gathered from $\hat{V}_i$ and the test budget are analyzed to produce a set of heuristics used in aiding the user in initial sample size selection. The results are discussed in Chapter 6 and a table is presented to summarize recommendations on sample size selection. The results from the block designs as discussed in this chapter and analyzed in Chapter 6 are located in the appendix. Further analyses can be conducted in future studies to redefine the heuristics developed herein. The reader should note that a single n_{ir} value that can be applied in all situations and environments does not exist. We present a generalized approach to determine n_{ir}.

5.3 DFKS - Selection of the input FLCs through augmented spatial sampling

In this section, we describe an augmented spatial sampling scheme that lends itself to Kriging methodology. Terminology that we use to describe this technique is Design for Kriging Sampling (DFKS). Based on $\mathcal{N}(n \times n)$, $\{x_i, x_{i+1}, \dots, x_n\}$ is selected randomly using a spatial sampling scheme based on the set of lags as defined in the factor space. The idea behind our algorithm is taken from the well-known simulating annealing algorithm. The simulated annealing algorithm pseudo code as taken directly from [77] is found below.

{

Step 1: Parameter Initialization;

1.a) Set the annealing parameters;

T_{in};

el_{max};

α;

1.b) Initialize the iteration counter;

$el = 0$;/* el: outer loop counter */

1.c) GENERATE the initial solution (generate the solution randomly or start with a known solution). Calculate the objective function, Get; $solution_o,\ objective_o$;

Step 2: Annealing schedule;

'Execute steps 2.a-2.g until conditions in 2.g are met'

2.a) Inner loop initialization;

$il = 0$; /* il; inner loop counter */

2.b) Initialize solution for the inner loop;

$solution_o = solution_{el}$;

$objective_o = objective_{el}$;

2.c) Achiving equilibrium at every temperature. Execute inner loop stes 2.c.1-2.c.5 until conditions in 2.c.5 are met;

2.c.1) $il = il + 1$;

2.c.2) Generate a neighboring solution and calculate the new objective function

2.c.3) $\varepsilon = objective_{il} - objective_{il-1}$;

2.c.4) IF ($\varepsilon \leq 0$) OR Random(0,1) $\leq e^{(-\frac{\varepsilon}{T_{el}})}$;

THEN accept $solution_{il}, objective_{il}$;

ELSE reject $solution_{il}$;

$solution_{il} = solution_{il-1}$;

$objective_{il} = objective_{il-1}$;

2.c.5) IF ($il \geq LMC$)

THEN terminate the inner loop GOTO step 2.d

ELSE continue the inner loop GOTO step 2.c.1

2.d) $el = el + 1$;

2.e) $solution_{el} = solution_{il}$;

$objective_{el} = objective_{il}$;

2.f) $T_{el+1} = \alpha^* T_{el}$;

2.g) IF $(el \geq el_{max})$

THEN terminate the outer loop GOTO Step 3

ELSE continue outer loop GOTO Step 2.a

Step 3: Terminate the best solution obtained and stop

}

Prior to presenting the pseudo code for our selection process, it is important to show the developmental steps of the sampling process and to formally define the problem. The mathematical model is defined by:

Objective:

$$\text{maximize } \sigma_E^2 = \prod (\boldsymbol{\Lambda}_E)(\boldsymbol{c}_E) \; where\ sub\ E\ is\ the\ exponential\ fitted\ variogram$$

Subject to:

$$c_0 = 0,$$

$$c_1 = 1,$$

$$a = \max\{h_i\},$$

$$\sum_{i=1}^{N} \boldsymbol{\lambda}_E = 1$$

Where:

$$\boldsymbol{\Lambda}_E = \boldsymbol{C}_E{}^{-1}\boldsymbol{c}_E$$

$$\mathcal{N}(n \times n)$$

$$\{x_i, x_{i+1}, \dots, x_d\}$$

To devise a solution to the problem stated, we begin by creating a two dimensional factor space array. The first dimension uniquely identifies the element from x_{min} to N is the total FLCs contained in the factor space. The second dimension contains the FLC elements themselves. A pictorial of this array is shown below.

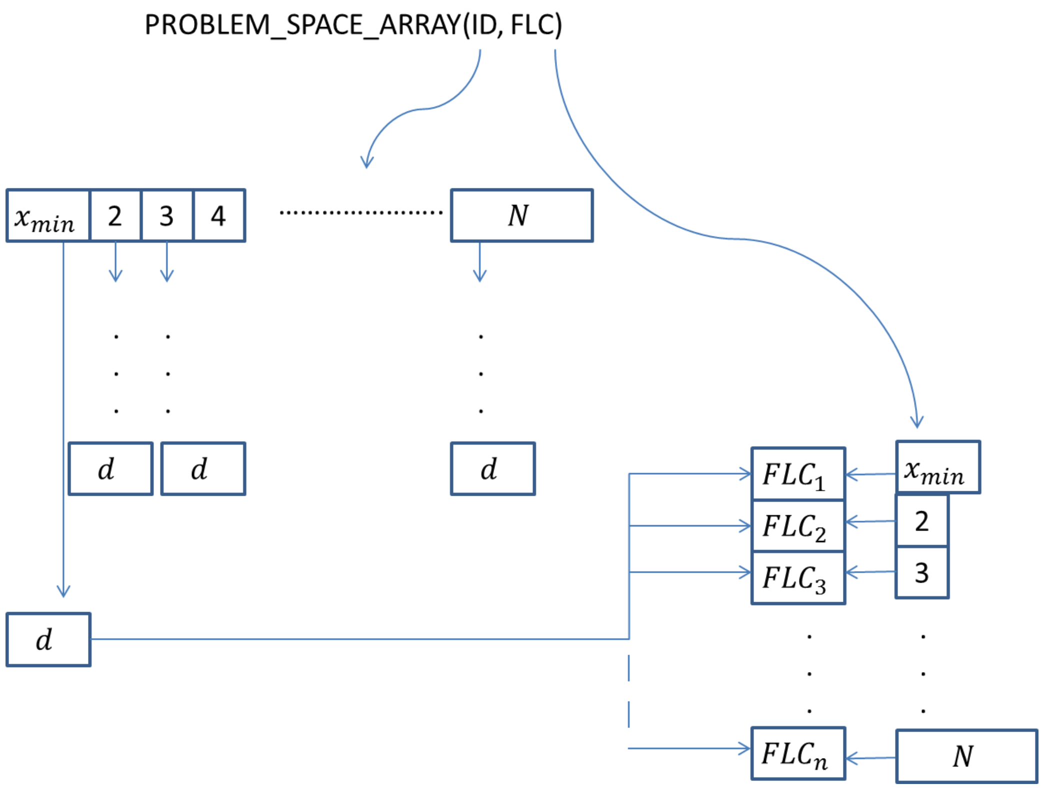


Figure 15: Factor space Array

Next, a lag array is assigned two unique again as a two dimensional array. The first dimension is the unique identifier and the second dimension is the ith and jth lag elements of the Euclidean distance matrix $\boldsymbol{X}$. Similarly, the covariance matrix $\boldsymbol{C}$ is defined in the same manner. This is made clear in the diagram below. The reader should note that the Distance/COV in the figure below does not mean divided by, instead it is simply showing the two arrays have the same composition in one figure.

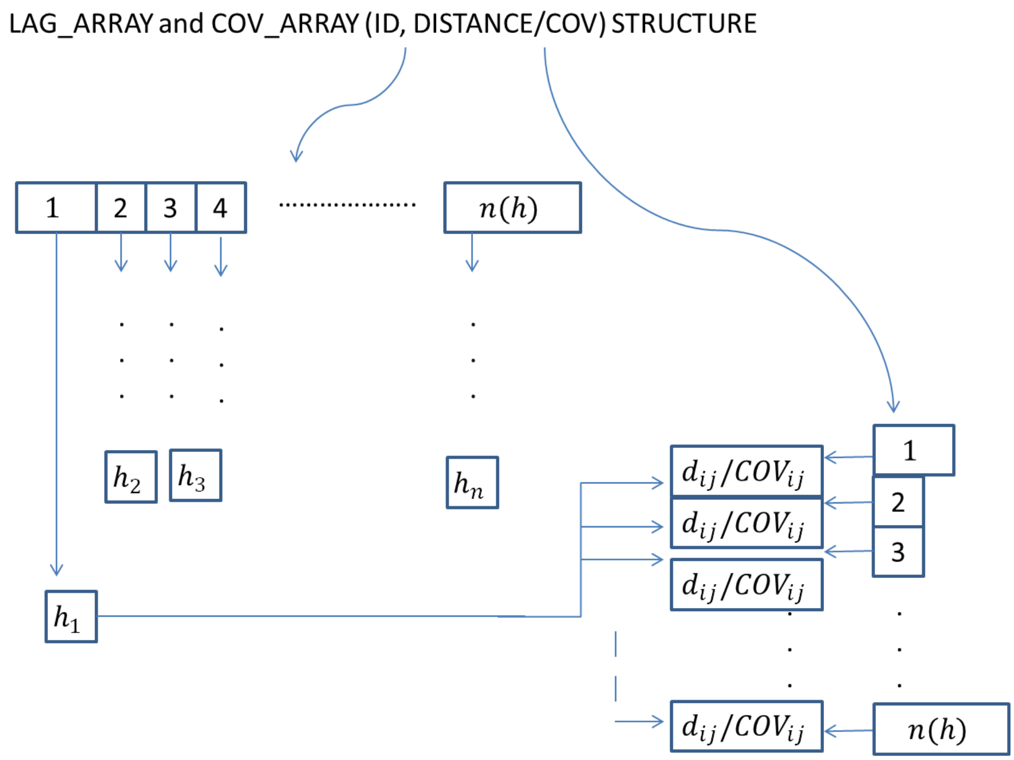


Figure 16: Spatial Sampling Design Matrix

After the array assignments have been completed and the initial lags have been calculated and populated, the DFKS process begins. Prior to beginning the augmented simulated annealing process for the estimation process we must state our software initialization requirements. The factor space must be known and is defined upfront by the user. The factor space consists of three user inputs: (1) x_{min}, (2) x_{max}, and (3) d. With the user definition of the factor space, we proceed to the initial sampling rule:

- The sample locations at $n_{ir} = 2^d$ are all permutations of $\{x_{min}, x_{max}\}$, i.e., all FLCs located at the factor space boundaries. Kriging is an interpolation process, not useful for extrapolation and therefore the boundaries need to be sampled as a minimum, therefore we set $n_{ir} = 2^d$ as the minimum FLC selection for pilot design
  - After FLC selection, the center point, $\frac{FLC_{min}+FLC_{max}}{2}$, rounded up of the factor space is selected. FLC_{min} and FLC_{max} are the first and last index values for the factor space. This allows for the highest possible estimation variance as the initial starting point of the search algorithm

Since the initial FLCs and the initial estimated variance have been defined as in traditional DOX literature, the DFKS process begins. This search process is described through the use of pseudo code. Recall our array definitions as they will be referenced throughout the pseudo code explanation.

{

Step 1: Parameter Initialization;

1.a) Set variogram parameters;

$c_0 = 0;$

$c_1 = 1;$

$a = \max\{h_i\};$

1.b) Gather user defined data;

Minimum lag increment = 1;

$x_{min} = user\ defined;$

$x_{max} = user\ defined$

1.c) Initial Dimensioning of Arrays;

Problem_Space_Array;

Distance_Array;

Covariance_Array

1.d) GENERATE initial FLC selection;

Select $n_{ir} = 2^d$ through all permutations of x_{min}, x_{max};

1.e) GENERATE initial candidate input;

Select center point of factor space;

1.f) Initial Dimensioning of Arrays;

Candidate_Distance_Array;

Candidate_COV_Array;

1.g) GENERATE initial solution. Calculate initial estimation variance at the design midpoint, Get; $krig_{var}$;

1.h) SET direction_flag = 1 (next search will be an increasing FLC search);

Step 2: Search Schedule;

2.a) Outer Loop Initialization ;

'Execute until FLC LISTCOUNT = N or Budget ≤ 0 if selected'

2.b) Perform Array Slicing (see next section);

2.c) Inner Loop Initialization;

‘Execute WHILE FLC LISTCOUNT <> N’

Step 3: Solution Search;

3.a) Select next input candidate based on direction_flag;

3.a.1) IF direction_flag = 1 THEN search increasing FLCs

3.a.2) IF direction_flag = 0 THEN search decreasing FLCs

3.a.3) $CAND_INPUT_{ID} = CAND_INPUT_{ID\pm1}$where $ID$is the index of the most recently added FLC and the $\pm$ is based on the direction_flag (Generate a neighboring solution);

3.b) Check $CAND\_INPUT_{ID\pm1}$ against the current (prevent duplicate FLC selection);

3.c) Calculate the new object function

3.c.1) If $objective_i > objective_{i-1}$ THEN accept tenative $objective_i$ as $solution_i$;

3.c.2) ELSEIF $objective_i \leq objective_{i-1}$ and inner loop <> $LBOUND(problem\_space\_array)$ OR $UBOUND(problem\_space\_array)$ THEN CONTINUE;

3.c.3) ELSEIF inner loop = $LBOUND(problem\_space\_array)$ OR $UBOUND(problem\_space\_array)$ THEN EXIT inner loop;

Step 4: Continue inner loop

4.a) FOR EACH $objective_i > objective_{i-1}$ THEN accept tenative $objective_i$ as $solution_i$;

4.b) IF inner loop = $UBOUND(problem\_space\_array)$ and factor space search exhausted THEN EXIT inner loop;

Step 5: Accept $solution_i$;

5.a) Add candidate as the next FLC;

5.b) Add $estimation\ variance, sample\ size, budget\ remaing$ and FLC to the GUI;

5.c) IF outer loop = $UBOUND(problem\_space\_array)$ THEN EXIT outer loop;

Step 6: Terminate the best solution obtained;

6.a) Add EI to the GUI;

6.b) and STOP.

}

5.4 Dynamic Array Slicing of the Covariance Matrix

With sequential DFK, the fitted variogram functions must be completely recalculated during each iteration to incorporate the introduction of new FLCs into $\boldsymbol{X}$ and $\boldsymbol{C}$. This can occur with random sampling, minimax sampling, Latin Hypercube Sampling (LHS) sampling, and DFKS process proposed here. To avoid recalculating the distance matrix and the covariance matrix altogether during each iteration we introduce array slicing into the software package. Visual Basic for Applications (VBA) does not include multidimensional array slicing commands. We therefore created our own method of performing this operation. The logic behind the array slicing technique is based solely on sequential sampling. For clarity, we again present the logic in form of pseudo code.

{

Step 1: Gather Initial State;

Distance_Array;

Covariance_Array;

Step 2: Initialize Outer Loop (m);

FOR NEXT through array rows;

Step 3: Initialize Inner Loop (n);

FOR NEXT through array columns up to m; (the reason for looping up to m is to gain computational efficiency by only looking at half of the matrix due to its symmetrical nature)

Step 4: Determine if slice is required;

IF $m = n$ then 0 since diagonals are 0 with $c_0 = 0$;

ELSEIF $INDEX_{CD} < INDEX_{CI}$ THEN data remains in $element_{m,n}$; (where CD is the current design and CI is the candidate input that was chosen to be added through DFKS)

ELSEIF $INDEX_{CI} < INDEX_{CI+1}$ THEN perform slice;

Step 5: Determine Slice Method;

IF $move_{counter} > n$ THEN move all subsequent data down a row $(m + 1)$;

ELSEIF $move_{counter} \leq n$ THEN move all subsequent data down a row $(m + 1)$ and across a column $(n + 1)$.

}

After the slicing operation has occurred, the only blank elements left in the arrays are elements that have yet been calculated. A subsequent dual row and column loop in the software calculates the new Euclidean distances and γ values based on the FLC chosen from the DFKS algorithm. The result of this process is that only the new elements of the array are calculated through the use of the slicing logic that is inherently incorporated into the software. This technique allows for CPU time optimization.

5.5 The Covariance Function by Use of a Standard Method

Fitted variogram functions in the literature range from user interpretation of graphs under subjective judgment to MSE calculations, through parameter tweaking, of the empirical variogram versus the fitted variogram. We aim to develop a standard model for determining variograms. This standard model aims to optimize the variogram function, not necessarily the parameters c_0, c_1, a. In fact, the method we present is based on the assumption that $c_0 = 0$. We aim to present a standardized model for generating the fitted variogram based on the empirical semivariogram values during sequential Kriging, not the test planning portion of the software as that part of the software is relevant prior to data sampling. For clarity, the test planning portion of the software uses the exponential fitted function due to lack of empirical data as previously described. Our standardized model is the default choice for variogram modeling in DFK software.

In order to develop a standardized model some number $N(h)$ of empirical variogram points and the $N(h)$ lags associated with generating the variogram are determined. Prior to performing our iterative regression approach, we first determine the shape of the empirical variogram curve. If the software determines the empirical variogram curve does not exhibit spatial correlation, then the iterative regression model is limited to simple linear regression which is the optimal solution for performing Kriging under these assumption violations. After, determination of the empirical variogram curve a simple linear regression model is initialized:

$$y = \beta_0 + \beta_1 \mathrm{x} + \varepsilon$$

where y represents the empirical variogram $\gamma(h)$ calculated by

$$\gamma(h) = \frac{1}{2N(h)} \sum_{k=1}^{N(h)} [z(u_k + h) - z(u_k)]^2 \;\; h \in \mathbb{R}^d$$

and x represents the individual lags. The regression model that results may not meet the non-negative requirement for the variogram. In order to eliminate any decreasing $\hat{y}$ values, we smooth the regression model. This is accomplished by defining and solving the following problem.

Objective:

$$\min \sum_{i=1}^{N(h)} (\gamma(h) - \hat{y})^2$$

Subject to:

$$\beta_0 = 0$$

FOR $add\_constraint$ = 1 to $N(h)$ STEP 1

$$\hat{y}_{add\_constraint} \geq \hat{y}_{add\_constraint-1}$$

NEXT $add\_constraint$

By changing:

$$\{\beta_0 \ldots \beta_n\}\ where\ n\ is\ the\ current\ iteration$$

This problem is solved through the use of DFK software by first calling the regression analysis pack in Excel® and secondly by calling the solver add-in in Excel®. After the software sets the parameters for the solver, the software initially sets the solver algorithm to GRG which solves non-linear but smooth problems. This method generates solutions with very high computational efficiency. If the empirical variogram points are non-linear non-smooth, then software switches over to EA to provide a solution to the model. In this event, the computational effort may be high and generating an optimized solution possibly requires CPU times on the order of minutes. After the solver operations are finished, an iteration is considered complete. Iterations continue until either one of the two requirements are met: (1) the reduction in the sum of squares of the residuals of the current iteration is the same or greater than the previous iteration or (2) a $R^2 \geq 80\%$ and at least a 3rd order model has been obtained. The stopping criterion of the 2nd requirement was observed through multiple testing. The program terminates without finding a solution utilizing the standard model if the software has completed an 8th order model without satisfactory results or $R^2 = 100\%$ as results become unreliable.

Based on this approach, the sequential Kriging software generates a standard and optimized fitted variogram solution. These calculations are all performed in the background of the software. There is one limitation to this approach. GRG is a local optimizer thus may not always produce acceptable results. The software recognizes when this event happens and automatically uses the EA algorithm. Results from this process are shown in the following chapter.

5.6 Reduction in Sampling Iterations and Stopping Criterion Based on Budget

Sequential sampling in the sense of one sample at a time may not be a realistic achievement in the physical testing environment. Physical test environments may experience many limitations such as personnel availability, range availability, schedule, repeatability, and environmental change. The iterations in this research are based on budget availability. Users can use the test planning portion of the software to get initial estimates of sample size based on user inputs of budget constraints. This method of planning gives users advance knowledge prior to gathering sample responses. After sample responses are gathered, Kriging is performed with knowledge of the sample response data. After analyzing the estimation variance, additional sample data is gathered strictly based on remaining/additional budget. In the event that additional budget is required, utilization of the test planning to determine an updated n_{ir} and sequential Kriging for interpolation can give the user good indications additional budget requirements in order to perform tests necessary to gather the adequate sample data.

The idea behind this approach aims to reduce the number of sampling iterations and to allow for careful consideration when selecting a sample size during these individual iterations if required. Utilizing test budgets in this process allows for a basic, yet realistic approach in iteration reduction. The following chapter provides a demonstration of utilizing budgets for iteration reduction and how the combined use of the test planning and sequential Kriging software can be used as a tool for determining additional budgets when required.

5.7 Summary of Advancements

This chapter developed mathematical methodologies and algorithms that are required in order to develop advanced DFK software and to test the software in the test planning and validation environments that are presented in the next chapter. In this chapter we introduced a novel approach at selecting the fitted variogram function without the presence of empirical data. We also introduced the concept of cost constraints into the sequential sampling process. Since our focus is on the interpolation of physical systems, it is imperative that cost constraints be taken into consideration as it affects the overall amount of samples that are collected. This presents users with a trade-off decision of interpolation accuracy versus additional cost. We also introduced a unique random spatial sampling scheme that focuses on sampling directly from lag information instead of FLC information after the problem boundaries have been established. This technique is combined with the sample size selection process to yield information to users of the advanced DFK software about initial sample size and associated FLCs. After the initial samples are taken, users can begin the sequential Kriging process. In this process, a standard model for fitting the variogram through the use of iterative regression and GRG or EA is developed. This method holds to variogram assumptions and allows for optimization in this area along with accurate Kriging predictions. Finally, we introduced a methodology to decrease the number of sequential samples by first, optimizing the sample size and sample locations selections during the sequential sampling process, and second, basing the iterative samples on an overall test budget. The next chapter discusses applications of the research through utilization of the software.

CHAPTER 6

Complete Methodology for the Advanced DFK Process

This chapter focuses on test planning and validation applications that are possible through the use of DFK software. It presents results of the software in four distinct areas or demonstrations. The first is the difference in MSE using standard fitted variogram models versus the standard model as defined previously. The second demonstration examines developing general heuristics for initial sample size selection based on expected improvement (EI) from the test planning portion of the software. The third discusses the combined use of the test planning and sequential Kriging aspects of the software in determining additional budget required. The final demonstration is a study on validation of black box simulations against verified MC models using all aspects of the DFK software. The chapter concludes with a brief summary.

6.1 Application Area 1: Standard Variogram versus Traditional Models

In the first applications area we compare, through the use of MSE, our standard variogram model approach against the linear, spherical, exponential, and Gaussian literature models. It is of note to remind the reader that a single optimal neighborhood selection was used in the standard model development. To demonstrate results, we used a noisy empirical variogram with 20 unique lag distances. The empirical variogram is shown below.

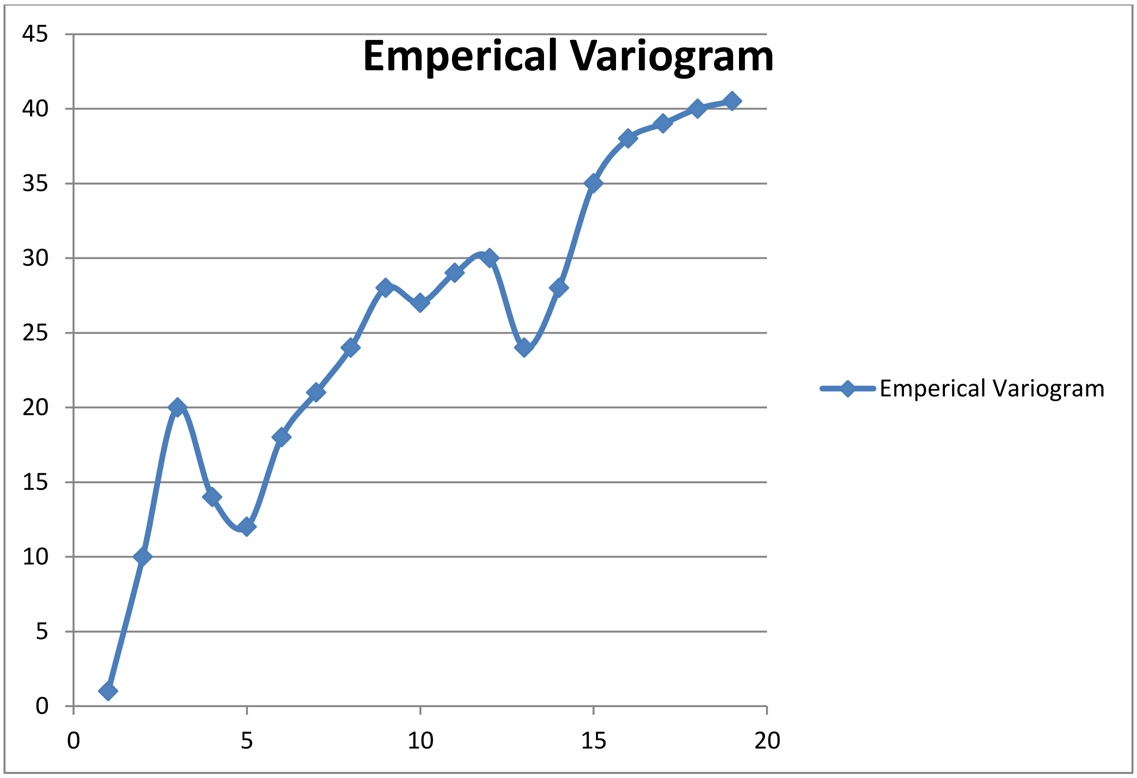


Figure 17: Noisy Empirical Variogram

The first objective is to fit the above empirical variograms with linear, spherical, exponential, and Gaussian models while running GRG or EA on $\{c_o, c_1\}$ and setting $a = \max\{h_{N(h)}\}$. This is defined through the following model:

Objective:

$$\min \sum_{i=1}^{N(h)} (\gamma(h) - \gamma(h)_i^f)^2$$

Subject to:

$$Fitted\ Model\ Function$$

$$c_0 = 0$$

By changing:

$$c_1$$

$$a$$

The optimized models were found to be:

Table 39: Parameter Values for Fitted Variogram

	c_0	a	c_1
Linear	0	20	41.5
Spherical	0	20	39.131
Exponential	0	20	37.432
Gaussian	0	20	41.5

Table 40: Optimized Variogram Values for Traditional Functions

Linear	Residuals	Spherical	Residuals	Exponen	Residuals	Gaussian	Residuals
2.0750	1.16	2.9324	3.73	5.2139	17.76	0.3101	0.48
4.1500	34.22	5.8501	17.22	9.7016	0.09	1.2265	76.97
6.2250	189.75	8.7385	126.82	13.5642	41.42	2.7088	298.99
8.3000	32.49	11.5828	5.84	16.8887	8.34	4.6928	86.62
10.3750	2.64	14.3685	5.61	19.7502	60.07	7.0953	24.06
12.4500	30.80	17.0807	0.85	22.2131	17.75	9.8198	66.92
14.5250	41.93	19.7050	1.68	24.3329	11.11	12.7628	67.85
16.6000	54.76	22.2265	3.15	26.1575	4.65	15.8205	66.90
18.6750	86.96	24.6306	11.35	27.7279	0.07	18.8945	82.91
20.7500	39.06	26.9026	0.01	29.0796	4.32	21.8968	26.04
22.8250	38.13	29.0280	0.00	30.2429	1.54	24.7534	18.03
24.9000	26.01	30.9918	0.98	31.2443	1.55	27.4068	6.72
26.9750	8.85	32.7796	77.08	32.1061	65.71	29.8163	33.83
29.0500	1.10	34.3767	40.66	32.8479	23.50	31.9581	15.67
31.1250	15.02	35.7683	0.59	33.4864	2.29	33.8233	1.38

33.2000	23.04	36.9398	1.12	34.0360	15.71	35.4158	6.68
35.2750	13.88	37.8765	1.26	34.5090	20.17	36.7498	5.06
37.3500	7.02	38.5637	2.06	34.9161	25.85	37.8465	4.64
39.4250	1.16	38.9868	2.29	35.2665	27.39	38.7318	3.13
41.5000	0.00	39.1311	5.61	35.5681	35.19	39.4338	4.27
SS(RES)	647.97	**SS(RES)**	**307.93**	SS(RES)	384.49	SS(RES)	897.16

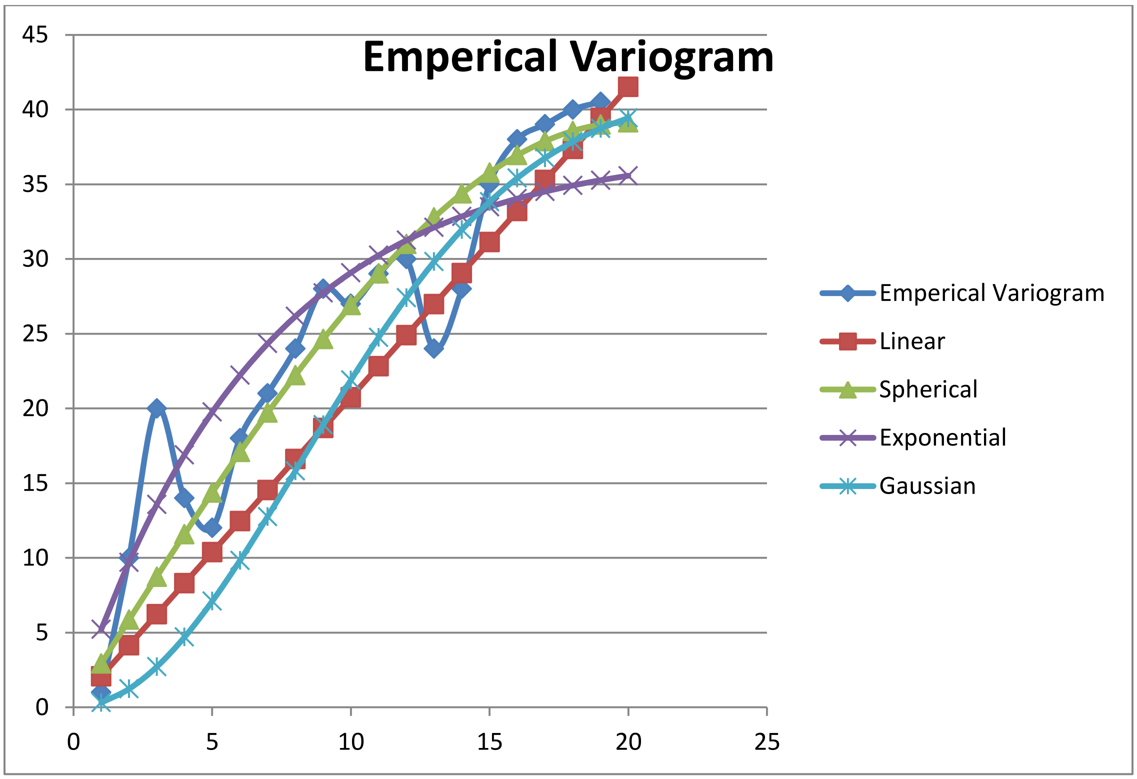


Figure 18: Graphical Variogram Summary

From the experimental trials above, the spherical variogram had the lowest $SS(RES) = 307.93$ of all the models. We will now compare the $SS(RES)_S$ with $SS(RES)_{SM}$, where SM is the standard model proposed in this dissertation. We start by defining the new model:

Objective:

$$\min \sum_{i=1}^{N(h)} (\gamma(h) - \hat{y}_i)^2$$

Subject to:

$$\beta_0 = 0$$

FOR $add\_constraint$ = 1 to 20 STEP 1

$$\hat{y}_{add\_constraint} \geq \hat{y}_{add\_constraint-1}$$

NEXT $add\_constraint$

by changing:

$$\{\beta_0 \ldots \beta_n\} where\ n\ is\ the\ current\ iteration$$

Solving the above problem, allows us to smooth the regression model to obtain a non-decreasing function. We begin by performing iterative regression analysis. The first order simple linear regression resulted in an $R^2 = 0.89837$. After smoothing the 1st iteration, the $SS(RES)_{SM} = 647.96875$ where $SS(RES)_{SM} > SS(RES)_S = 307.93$. The reader should be cognizant of the fact that $SS(RES)_{SM}$ is not to be confused with $SS(RES)$ in a general regression analysis because $SS(RES)_{SM}$ is the sum of the square of the residuals after the GRG algorithm has been applied to smooth the function. Next, the stopping criterion were analyzed and since $R^2 > 80\%$ but the iterations were ≤ 3 the iterations continued. Adding the second regressor, we obtain $R^2 = 0.9057$ and $SS(RES)_{SM} = 216.0297318$, where $SS(RES)_{SM} < SS(RES)_S = 307.93$. Since our aim was to show $SS(RES)_{SM} < SS(RES)_S$ we manually stop the regression iterations. A diagram of the standard model titled "y-hat" included into the previous figure shows a visual comparison and contrast.

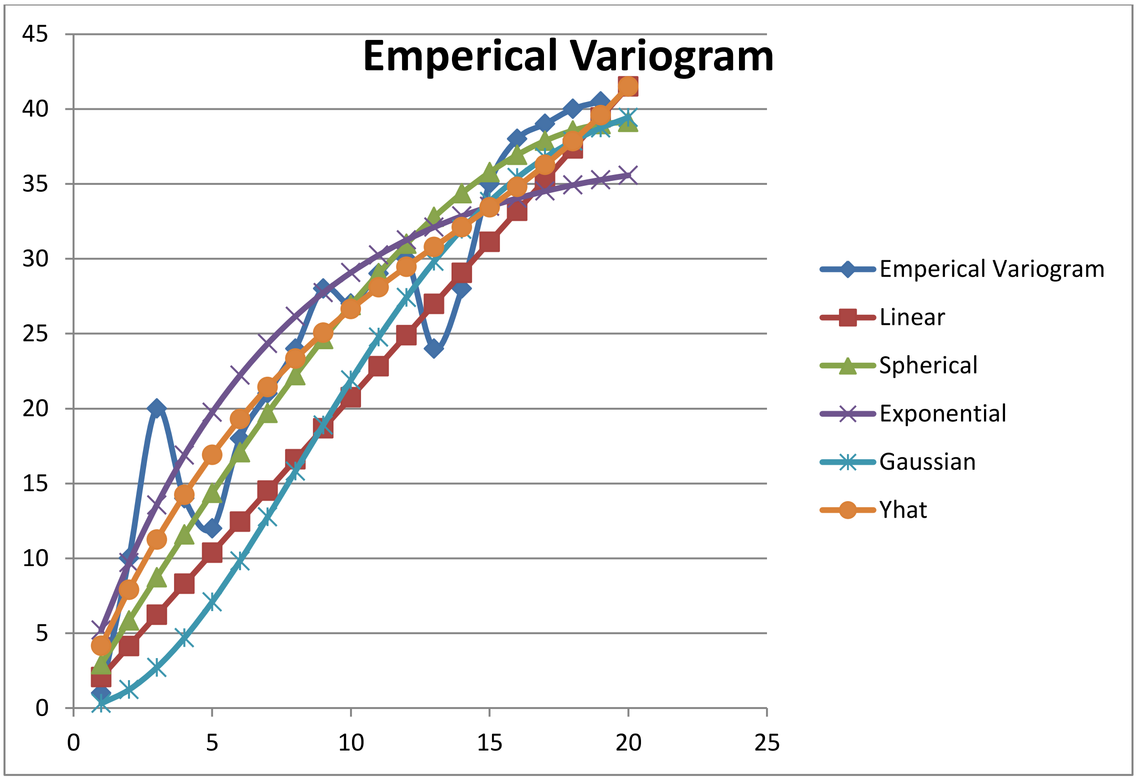


Figure 19: Standard Model Variogram

A more accurate variogram model, leads to better Kriging predictions. We propose that using this standardized approach, taking advantage of modern computer systems, is a viable alternative to the traditional methods of attempting to fit multiple variogram functions under the current assumptions. Additional variogram models along with various model testing confirm our approach. If $R^2 = 100\%$, GRG may yield results that are not feasible. Note that an $R^2 = 100\%$ is statistically attainable but we had to constraint $R^2 < 100\%$ because GRG wil not yield a usable solution. The software is designed such that it recognizes this requirement and prevents the next iteration of the regression model in the event that $R^2 => 100\%$. Future research may be conducted to ensure the lowest MSE is presented using the standard model while simplifying the complexity of the regression model.

6.2 Application Area 2: Sample Size Selection during Test Planning

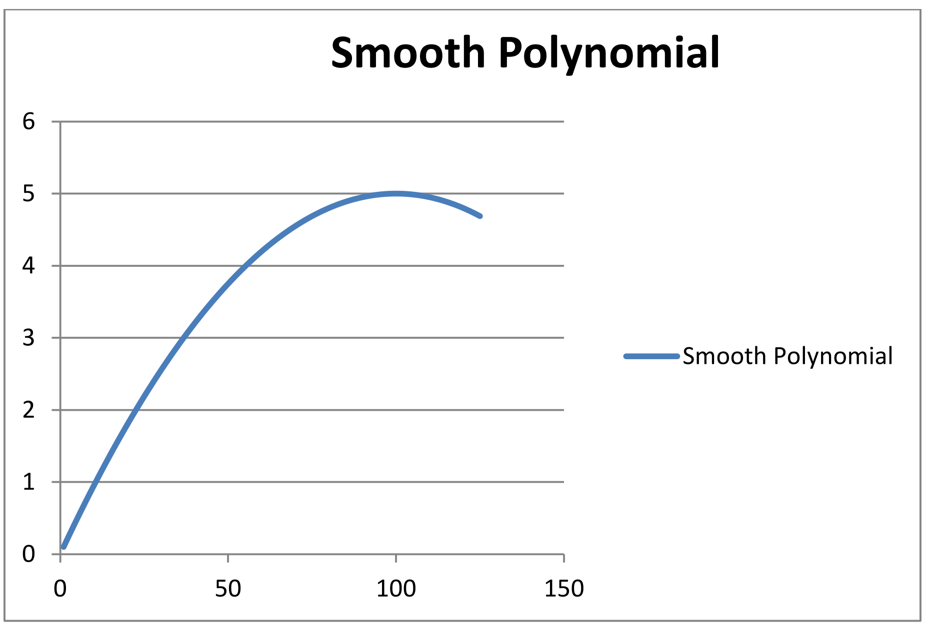
The test planning portion of the software is designed to aid in the determination of initial sample size and FLC selection that is adequate based on the designed experiment. The approach we use is to examine the total amount of variance reduction by increasing sample size. Based on various percentages of reduction, we compare our results to actual response data to determine heuristics for initial sample size n_{ir} and initial FLC selection. We generated 28 pilot designs with user inputs in the DFK software as shown in the table below.

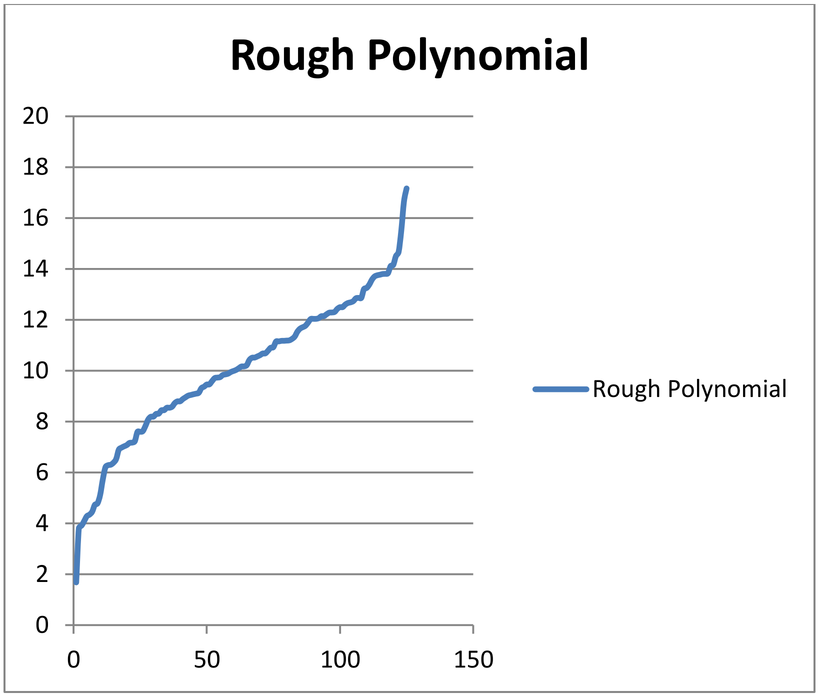
Table 41: Application Area 2 – 28 Pilot Designs

# Inputs/Dimensions	# of FLCs	% VAR Reduction
1	25	10
1	25	25
1	25	33
1	25	50
1	25	66
1	25	75
1	25	90
2	25	10
2	25	25

2	25	33
2	25	50
2	25	66
2	25	75
2	25	90
1	125	10
1	125	25
1	125	33
1	125	50
1	125	66
1	125	75
1	125	90
3	125	10
3	125	25
3	125	33
3	125	50
3	125	66
3	125	75
3	125	90

These inputs were selected to represent a varying quantity of FLCs, to cover dimensionality impacts and common percentages that users would generally specify. Each of the pilot designs were tested against three response models. The first response model portrays a smooth curve. The second model portrays an increasing response but “rough” curve, while the third model depicts a noisy response curve. The user input settings and model designs span similar situations as would be encountered in field studies or industry. Further testing should be conducted for models that display large noise in certain areas such as the tails, models with large variability given a small variability in FLCs, or other various models that are commonly encountered in practice. The three response curves that were selected are presented in the figures below.





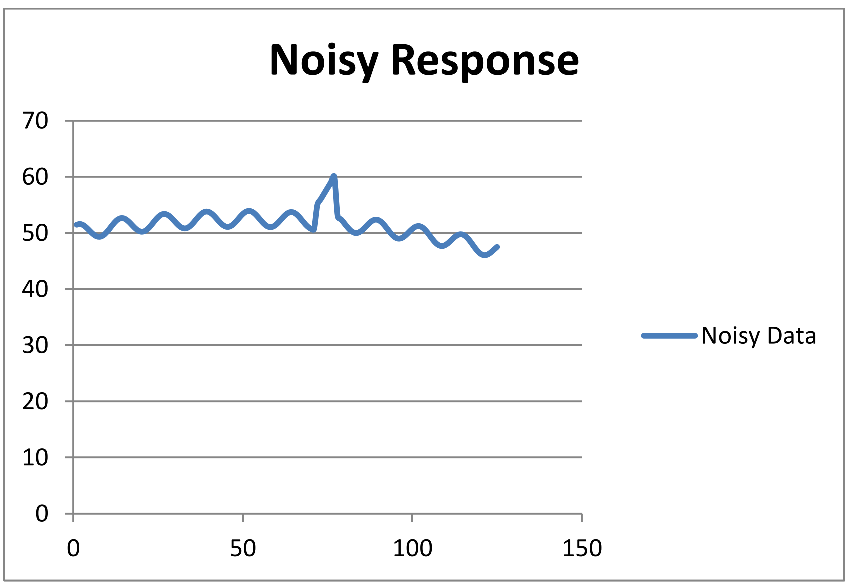


Figure 20: Response Data for Sample Size Selection

For each of the 28 pilot designs, we selected two FLCs at random and performed Kriging based on each of the three response data models. After collecting the Kriged data along with the prediction variances, we determined the actual observed response. This allowed for the calculation of the squared error between the response data and the Kriging model, e_i^2 $(i\colon i = 1, 2, \ldots n)$, where n is the intial sample size based on the selected % variance reduction. The following tables summarize the findings of the prediction variance and e_i^2. A brief discussion of the analysis of the results follows these tables. The complete table of data used to generate the summary tables below is found in Appendix D.

Table 42: Summary Data for Pilot Design Studies

		Summary of Random FLC Selection for n_{ir} Selection Smooth Response		
$N = 25$	$d = 1$	Candidate FLC	Variance	e_i^2
Variance Reduction (Pilot Design)	25%	5	0.4362048	0.000256
	33%	4	0.3785	0.00018225
	50%	8	0.120273272	7.92176E-06
	66%	3	0.071747605	2.5E-07
	75%	6	0.070012506	3.1783E-06
	90%	6	0.069653811	2.5E-07

		Summary of Random FLC Selection for n_{ir} Selection Smooth Response		
$N = 25$	$d = 2$	Candidate FLC	Variance	e_i^2
Variance Reduction (Pilot Design)	25%	(1, 2)	0.258012564	0.000981459
	33%	(2, 2)	0.30082494	0.002663559
	50%	(1, 4)	0.010271132	2.37865E-05
	66%	(2, 1)	0.05349981	0.000500417
	75%	(2, 3)	0.005026733	2.62449E-05
	90%	(3, 2)	0.162164651	2.43318E-05

		Summary of Random FLC Selection for n_{ir} Selection Rough Response		
$N = 125$	$d = 1$	Candidate FLC	Variance	e_i^2
nce Redu ction (Pilot Desig	25%	93	1.404718222	0.004019103
	33%	120	3.571362043	2.851391695

	50%	120	3.571362043	0.3842741
	66%	90	1.33942208	0.012527153
	75%	80	0.609687765	0.001007153
	90%	70	0.386930139	8.99955E-05

		Summary of Random FLC Selection for n_{ir} Selection Rough Response		
$N = 125$	$d = 3$	Candidate FLC	Variance	e_i^2
Variance Reduction (Pilot Design)	25%	(4, 4, 1)	~0	0.080620736
	33%	(5, 1, 2)	0.689085195	0.350832228
	50%	(4, 5, 5)	0.763073107	0.050146604
	66%	(3, 5, 4)	0.422687635	0.061600135
	75%	(3, 4, 5)	0.374146817	0.30880224
	90%	(4, 4, 2)	0.373281973	0.059318233

Analysis of these data did not show any clear methods for n_{ir} selection due to the randomness of the FLC selection. The random FLCs have a wide range of estimated variability based on the FLC selected and N. Candidate FLC variance would be much smaller when taken directly next to observed FLC responses and would increase as the candidate FLC would be further away (spatially) from observed responses due to the linear weight assignment method in Kriging.

A more definitive technique had to be established. Again, we conducted all the $1d$ tests as previously described except that we studied all unobserved FLCs in the factor space instead of random FLCs throughout the factor space. The same data of interest, e_i^2 and $V(\hat{Z})$ were collected except for each unsampled FLC in the factor space. The tables below summarize the finding from the secondary study. The first and third table show e_i^2 and $V(\hat{Z})$ values while the second and fourth table show the percent reduction in e_i^2 and $V(\hat{Z})$ as the percent of variability was reduced in the pilot design as generated from the test planning portion of the software.

Table 43: Summary of $\boldsymbol{e_i^2}$ and $\boldsymbol{V(\widehat{Z})}$ for all Unobserved FLCs

		Sum of Variability for all Unobserved FLCs in Factor Space		
$N = 25$	$d = 1$	Smooth Response	Rough Response	Noisy Response
Variance Reduction (Pilot Design)	25%	7.7971608	64.7506721	3.289102713
	33%	5.470734589	49.16320671	3.091723569
	50%	2.453574742	3.737812542	2.304255731
	66%	1.411036229	2.924561928	1.089949344
	75%	0.910162579	1.561493688	0.904060708
	90%	0.348269054	0.43705094	0.276952231
$N = 125$	$d = 1$	Smooth Response	Rough Response	Noisy Response
Variance Reduction (Pilot Design)	25%	99.01041773	928.9869839	81.41313689
	33%	29.0301759	512.1710515	71.06256482
	50%	24.4743451	512.1710515	30.50388668
	66%	4.178449566	76.19619544	14.20431279
	75%	8.807956097	48.77145576	22.71659314
	90%	2.943178333	15.8641357	9.839428735

		% Reduction in Variability for all Unobserved FLCs in Factor Space		
$N = 25$	$d = 1$	Smooth Response	Rough Response	Noisy Response
Variance Reduction (Pilot Design)	25%	42.39675433	52.82543656	30.02089645
	33%	29.74690356	40.10873976	28.21934163
	50%	13.34121589	3.049413587	21.03182197
	66%	7.672453844	2.385940648	9.948383875
	75%	4.948973125	1.273910881	8.251707313
	90%	1.893699248	0.356558564	2.527848768
$N = 125$	$d = 1$	Smooth Response	Rough Response	Noisy Response
Variance Reduction (Pilot Design)	25%	58.77924442	44.3608223	35.43708721
	33%	17.23426528	24.45710155	30.93174398
	50%	14.52961765	24.45710155	13.2775733
	66%	2.480608748	3.63850726	6.182779466
	75%	5.228995253	2.328925937	9.887960628
	90%	1.747268647	0.757541405	4.282855415

		e_i^2 for all Unobserved FLCs in Factor Space		
$N = 25$	$d = 1$	Smooth Response	Rough Response	Noisy Response
Variance Reduction (Pilot Design)	25%	0.004147	9.07303615	56.59759201
	33%	0.002203	9.07531689	36.29107218
	50%	0.0001375	2.988934423	3.79253055
	66%	1.425E-05	0.20451854	0.715526668
	75%	7.25E-06	0.208055907	0.320130762
	90%	1.25E-06	0.144014964	0.069381913
$N = 125$	$d = 1$	Smooth Response	Rough Response	Noisy Response
Variance Reduction (Pilot Design)	25%	0.954304	122.8576312	366.9507408
	33%	0.795937645	33.28298388	382.1229022
	50%	0.030811673	33.28298388	396.7815453
	66%	0.00102	10.27969455	79.2147675
	75%	0.001134845	2.858423796	47.26833224
	90%	0.000719842	0.072907515	13.89661911

		% Reduction in e_i^2 for all Unobserved FLCs in Factor Space		
$N = 25$	$d = 1$	Smooth Response	Rough Response	Noisy Response
Variance Reduction (Pilot Design)	25%	63.69955071	41.82302777	57.87889527
	33%	33.83894628	41.83354106	37.11265959
	50%	2.112054069	13.77777905	3.878389003
	66%	0.218885603	0.942747769	0.731725354
	75%	0.111362851	0.9590536	0.327378148
	90%	0.019200492	0.66385075	0.070952638
$N = 125$	$d = 1$	Smooth Response	Rough Response	Noisy Response
Variance Reduction (Pilot Design)	25%	53.49453551	60.6301274	28.52906096
	33%	44.61713943	16.42512177	29.70864031
	50%	1.727181417	16.42512177	30.84829553
	66%	0.057177195	5.073019755	6.158654773
	75%	0.063614948	1.410629501	3.674937757
	90%	0.040351507	0.035979791	1.080410664

Analysis of the data from the secondary study yields conclusive results. It is clear from the tables above that there are distinct breaking points in the reduction of e_i^2 and $V(\hat{Z})$ at the 50-66% pilot design variance reduction. With this knowledge we generate sample size n_{ir} guidelines in

Table 44 below. The complete table of FLC selections during the DFK software pilot design data generation process based on variance reduction is contained in Appendix E. An additional table for varying dimensions and N is contained in Appendix G. Even further studies to expand the table in Appendix G is recommended to yield a detailed list of recommended sample size and FLC locations based on N and dimension d.

Table 44: Recommended Pilot Sample Size and FLC Selection

$N = 25$ $d = 1$	Recommended n_{ir}	FLC Selection
Smooth Response	7 (Based off 50% Pilot Design Variance Reduction)	1, 4, 7, 13, 19, 22, 25
Rough Response	10 (Based off 66% Pilot Design Variance Reduction)	1, 2, 4, 7, 10, 13, 16, 19, 22, 25
Noisy Response	10 (Based off 66% Pilot Design Variance Reduction)	1, 2, 4, 7, 10, 13, 16, 19, 22, 25
$N = 125$ $d = 1$	Recommended n_{ir}	
Smooth Response	14 (Based off 50% Pilot Design Variance Reduction)	1, 16, 24, 32, 47, 55, 63, 71, 79, 94, 102, 110, 118, 125
Rough Response	28 (Based off 66% Pilot Design Variance Reduction)	1, 8, 12, 16, 20, 24, 28, 32, 39, 43, 47, 51, 55, 63, 67, 71, 75, 79, 83, 87, 94, 98, 102, 106, 110, 114, 118, 125
Noisy Response	28 (Based off 66% Pilot	1, 8, 12, 16, 20, 24, 28, 32, 39,

	Design Variance Reduction)	43, 47, 51, 55, 63, 67, 71, 75, 79, 83, 87, 94, 98, 102, 106, 110, 114, 118, 125

This section established a methodology for selecting initial sample size and FLC selection. With use of DFK software, additional designs can be replicated and studied with little effort to expand the above table.

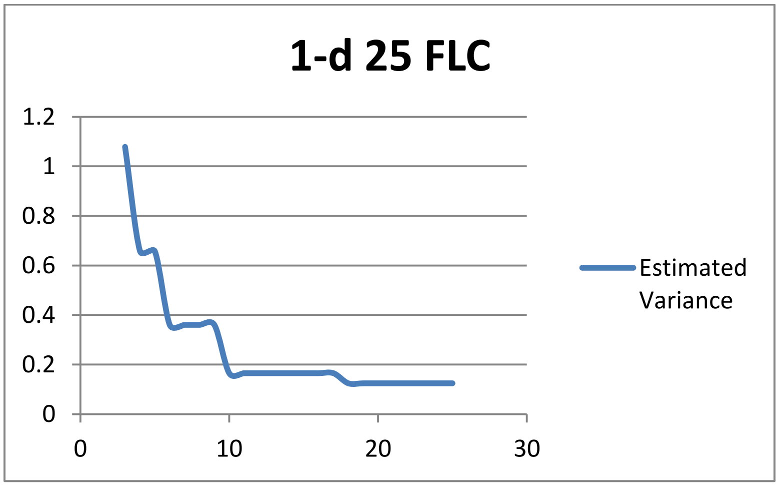
6.3 Application Area 3: Additional Budget Determination

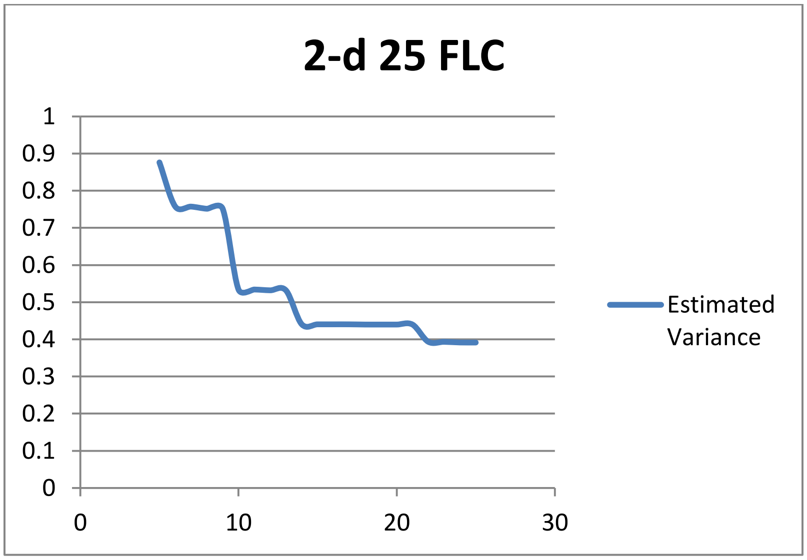
Based on the analysis of previous section, we pursue, through the use of DFK software, a method of determining additional budget in the event that a pilot design sample size did not produce adequate results. A result of this portion of research is to determine a subsequent sample size in the event that another iteration of experimental sampling is required. To address this issue, we focus on variance reduction results from the test planning portion of the software. We studied four designs for this application area. The four designs are shown below in Table 45.

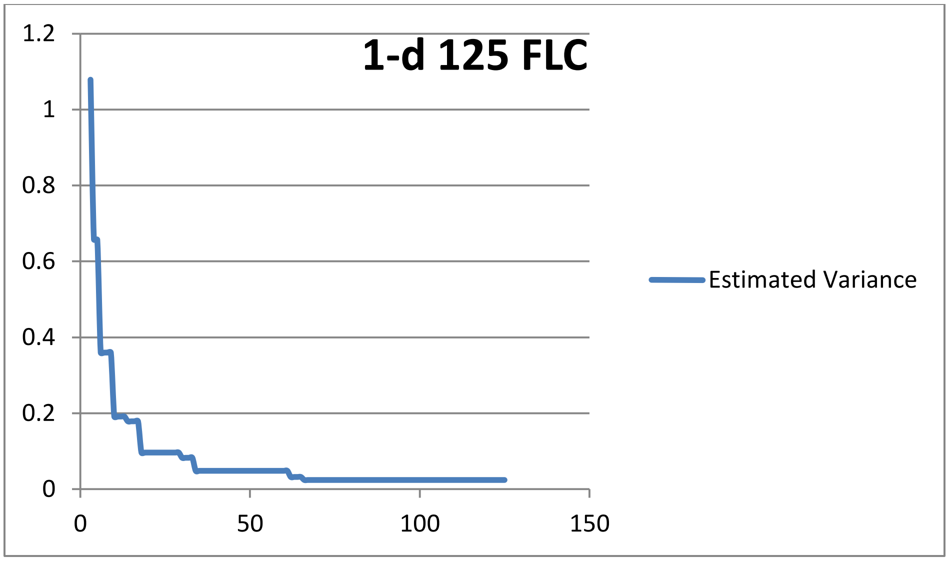
Table 45: Addition Budget Determination: Dimension and Factor Space Selection

Dimensions - d	Factor Space - N
1	25
2	25
1	125
3	125

For each of the four designs, a graph of variance reduction as N increased are plotted and shown below.







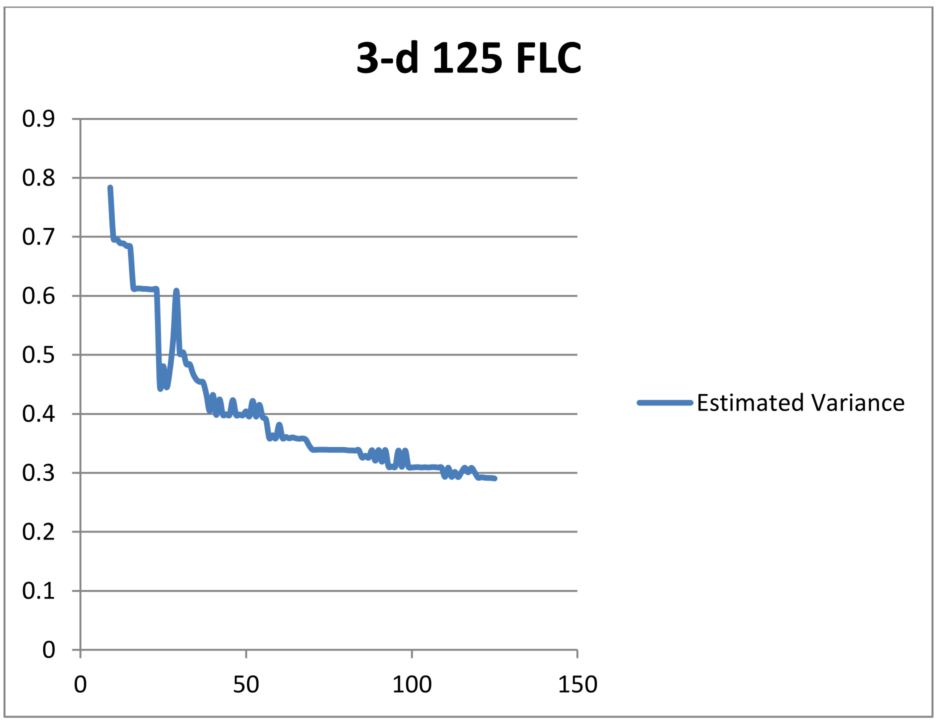


Figure 21: Variance Reduction in Pilot Designs

All four graphs display distinct breaking points in variance estimation as N increased. A comparison of the dips in the $1d$ 25 FLC and $1d$ 125 FLC graphs against the n_{ir} selection described in the previous section displays a direct correlation between n_{ir} selection and a major dip in the graph. Based on analysis of the graphs and the studies in the previous section, it is determined that in the event additional samples are required, users of DFK software should base an additional sample size selection where the next dip in the graph occurs. For example, the second large dip in the $1d$ 25 FLC design occurs as $n_{ir} = 7$ which corresponds to our initial sample size selection in Table 44. Users should examine the next dip in the graph which occurs at $n_{ir} = 10$ and thus request additional test budget for the three additional samples if required.

6.4 Application Area 4: Validation of MC Simulation against Limited Empirical Data

This section presents two distinct black box representations of physical systems that were independently generated by Dilcu Helvaci and provided for analysis. The test planning portion of the software is used to determine initial sample size and sample location positions. The sequential sampling portion of the software is used to complete analysis for sampling and to produce interpolations over the entire factor space. Monte Carlo simulations are used to generate the system responses. We analyze validity statements of the Monte Carlo model with and without the use of the advanced DFK software to demonstrate software effectiveness. The results are shown in the case below.

We begin by establishing an approach in which to compare two datasets. The chosen statistic is the Kolmogorov-Smirnov as it is a nonparametric test that determines if there exists a significant difference between two datasets. The null hypothesis for the Kolmogorov-Smirnov statistic is $H_0: cdf_{MC} = cdf_S$. The result of this test will determine whether our data sets cumulative distributions are practically the same at a 5% Level of Significance (LOS). In the event that we cannot reject $H_0: cdf_{MC} = cdf_S$ we will consider that MC model a good representation of the actual data. If H_0 is rejected, we will augment Y^S with the Kriging interpolations and perform the test again except the hypotheses will now use Y instead of Y^S. The factor space for our two black box problems are both determined to be $N = 243$ with $x_{min} = 1$, $x_{min} = 243$, and dimensionality of $d = 1$. The actual observed values are shown in the table below. The MC values are found in Appendix F.

Table 46: Observed Values from System 1 and System 2

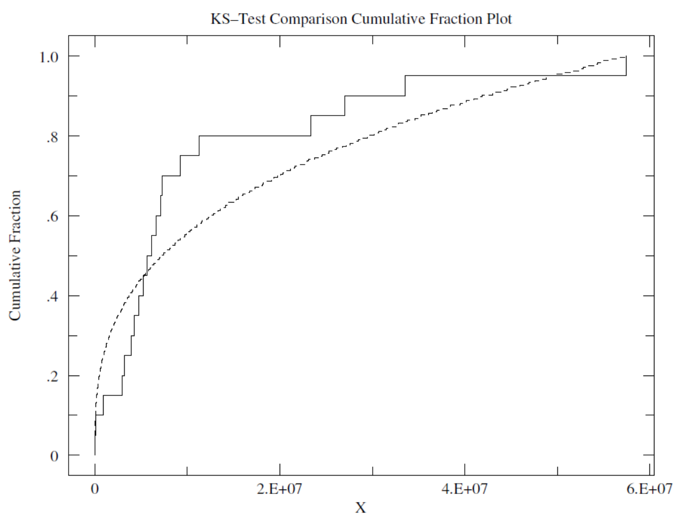
x	Y^S- System 1	Y^S- System 2
1	6.8	32.90095391
30	108722	37.0198053
60	866940	38.21338068
90	2922600	41.40984528
93	3224527.2	45.94042492
99	3889222.8	47.23850124
102	4253353.2	49.78464302
106	4773256.8	52.10657163
109	5189832.8	54.35796892
112	5629965.2	54.36469918
115	6094304	55.80191681
118	6583497.2	56.80046242
121	7098192.8	60.31219199
122	7275541.2	63.14901965
132	9214055.2	63.40949893
141	11229052.8	63.46611498
180	23354202	67.56625352
189	27034012.8	67.74433258
203	33495053.2	70.68682079
243	57443273.2	73.14682206

We proceed to perform the K-S test on MC each data set Y^{MC}. A table of the $p-value$s is provided below.

Table 47: Black Box: $\boldsymbol{p-values}$

System 1: $p-value$ ($Y^{MC} vs. Y^{S}$)	System 2: $p-value$ ($Y^{MC} vs. Y^{S}$)
0.275	0.089

Since the $p-value$s for both systems are > 0.05, we cannot reject the null hypothesis that the datasets are statistically the same. We draw evidence from the K-S test to state that the MC models for either system is an accurate representation of the sampled data sets Y^{S}. Standard K-S graphs are provided for visual clarification. The first figure represents System 1 and the second figure represents System 2.



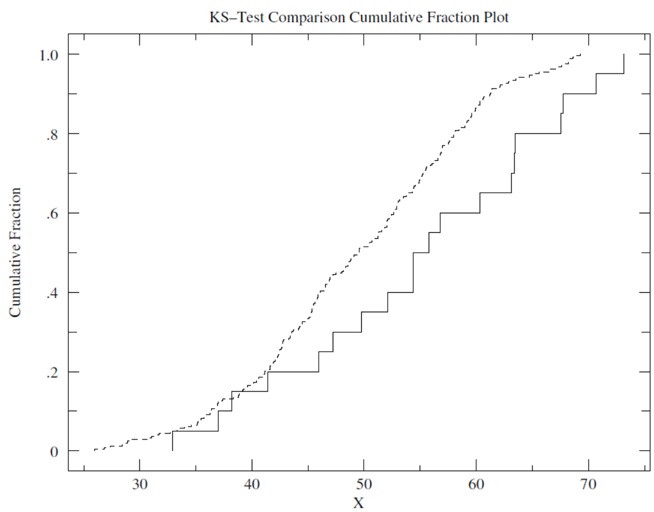


Figure 22: K-S Cumulative Probability Plots of MC vs. System Responses

Next, we repeat the process after we perform sequential Kriging on both Y^S data sets to interpolate at all unsampled FLCs in order to augment the data sets Y^S. The augmented data sets Y are found in Appendix F. After performing Kriging, the $p-values$ were again obtained and shown below.

Table 48: Black Box: $\boldsymbol{p-values}$ utilizing DFK Software

System 1: $p-value$ ($Y^{MC} vs. Y$)	System 2: $p-value$ ($Y^{MC} vs. Y$)
0.311	~ 0

The K-S plots are also shown in the following figures.

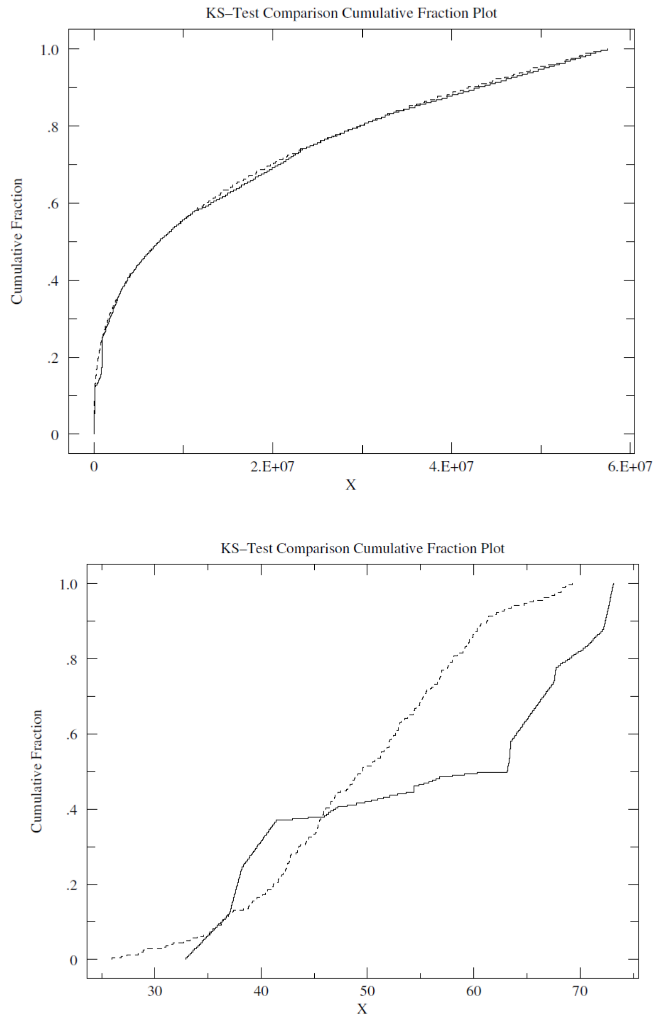


Figure 23: K-S Cumulative Probability Plots of MC vs. Augmented System Responses

The results show that for System 1, after the Kriging data was gathered, we more strongly cannot reject that the MC model is an adequate representation of the physical system. The MC model representing System 1 is considered valid. In System 2, the null hypothesis was not rejected initially. After generating the Kriging data, the null hypothesis now must be rejected, thus the MC model is not a good representation of System 2. The MC model representing System 2 is not considered valid.

6.5 Application Summary

This chapter demonstrates results of the methodologies presented in Chapter 5. We began the chapter by discussing the results of standardizing the variogram model based solely on necessary variogram requirements. No standard variogram model analyzed in this research demonstrated more accurate Kriging predictions in terms of e_i^2 versus the four common variogram functions found in Geostatistical literature even after optimizing the parameters (c_0, c_1, a) through GRG or EA as required. Our method allows for a dynamic smoothing model which minimizes the sum of squares of residuals based on an empirical variogram model. Our second objective was to demonstrate heuristics for initial sample size and sample location selection. We used an effective method based on percent of total estimated variance reduction along with percent total reduction in e_i^2. Our presented tables accurately demonstrate a distinct capability in determining sample size and FLC location selections when Kriging interpolation is performed. Our third objective was to determine the possibility of additional sample size selection in the event that the initial sample size n_{ir} did not produce adequate results. We were able to clearly demonstrate a trend through the graphical use of pilot design variance reduction data that directly corresponded to our results in Section 6.2. This allows users of DFK software a tool to determine additional cost requirements to obtain the required number of additional samples. Our final objective was to demonstrate validation capabilities while DFK software tool. We performed a K-S test on two systems in which we had no/little knowledge of the underlying response curve. The K-S tests in conjunction with DFK software, did in fact allow us to make claims about the validity of the corresponding MC models that were used.

CHAPTER 7

The Advanced DFK Application Software

This chapter introduces the DFK software. The attainment, installation, and use of the software are described in complete detail throughout the chapter. The chapter also includes many graphical aides that can be referenced during the use of the software. In addition, help features are available throughout the software GUI and can be accessed by simply clicking the appropriate help icon. The research conclusions immediately follow this chapter.

7.1 Introduction

The advanced DFK software is an add-in for Microsoft Excel® version 2007 and higher. The software implements all the advancements and mathematical derivations contained in this dissertation. It automates all calculations into two major steps allowing the process developed in this research to be used for data analysis. Without this unique software tool, this research would remain impractical for use in its intended applications and beyond.

7.2 Overview

The advanced DFK software is written in VBA. This platform is a high level language that incorporates versatility and the convenience of using Excel® while maintaining a satisfactory level of performance for the scale of problems presented herein. For major industrial purposes and analysis, the software application should be introduced into mainstream statistical software such as Minitab® or MATLAB®. To obtain a copy of the software for personal use, please request a copy from jlb0014@aubrn.edu. After receiving the file, copy it to one of two places depending on your Windows based operating system

- <x>:\Users\<user_name>\AppData\Roaming\Microsoft\AddIns (Windows 7)
- <x>:\WINDOWS\Application Data\Microsoft\AddIns\ (Windows XP).
- Open Excel®
- Go to File (or the Windows bubble in the top left corner if Office 2007 is installed)
- Options > Add-Ins > Go… > Check “DFK” and press OK
- Restart Excel®

The software has been successfully installed. To execute the software application, select the “Add-Ins” ribbon menu item, then select “Kriging”. The user may also request the automated installer to simplify the above steps. The software application was designed to be intuitive and user friendly. Upon opening the software, “Kriging Start”, the user is presented the following screen:

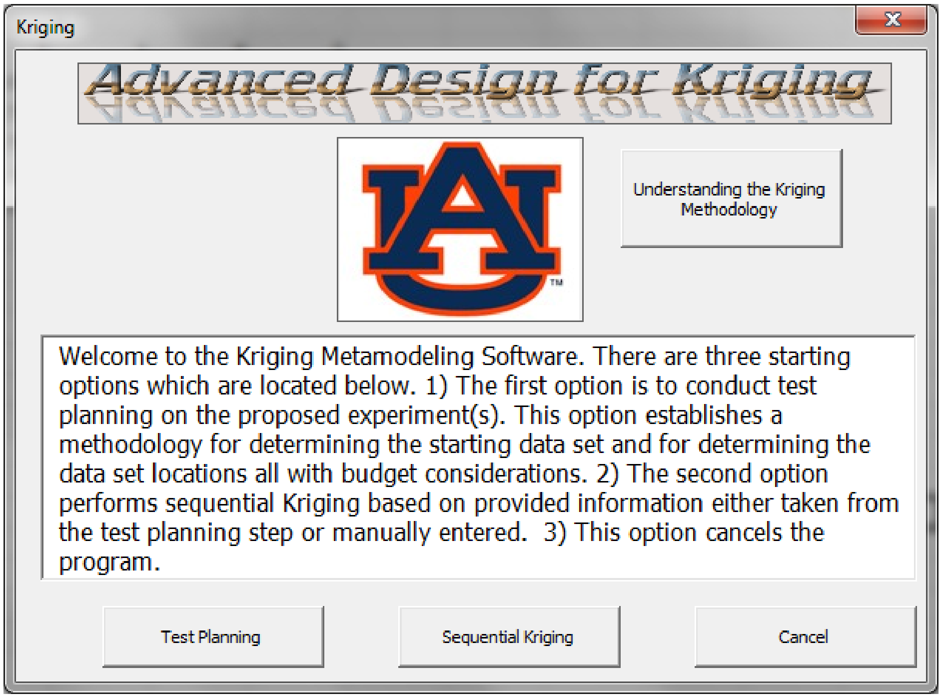


Figure 24: Advanced DFK Software Introduction Screen

From the introduction screen, users can perform the test planning tools as discussed and derived in this research or skip directly to performing Kriging depending on user needs. From this screen there is also a brief introduction to Kriging. This introduction is shown in the figure below.

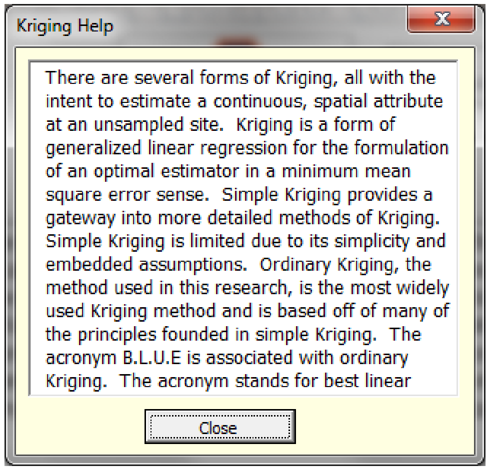


Figure 25: DFK Software - Kriging Introduction

The primary purpose of the above figure is to introduce users into the software application's common help feature. Anytime the help icon  is seen, users can click on it to open the above figure with pertinent help information. After returning back to the beginning GUI, a selection from one of the three buttons must be made, then the start-up screen will be unloaded and the program must be executed again in order to load it.

7.3 Initial Sample Size and Location Selection

To launch the test planning portion of the software application, execute the software which leads to the introduction GUI and then click on the "Test Planning" button across the bottom. This action brings users to the following screen:

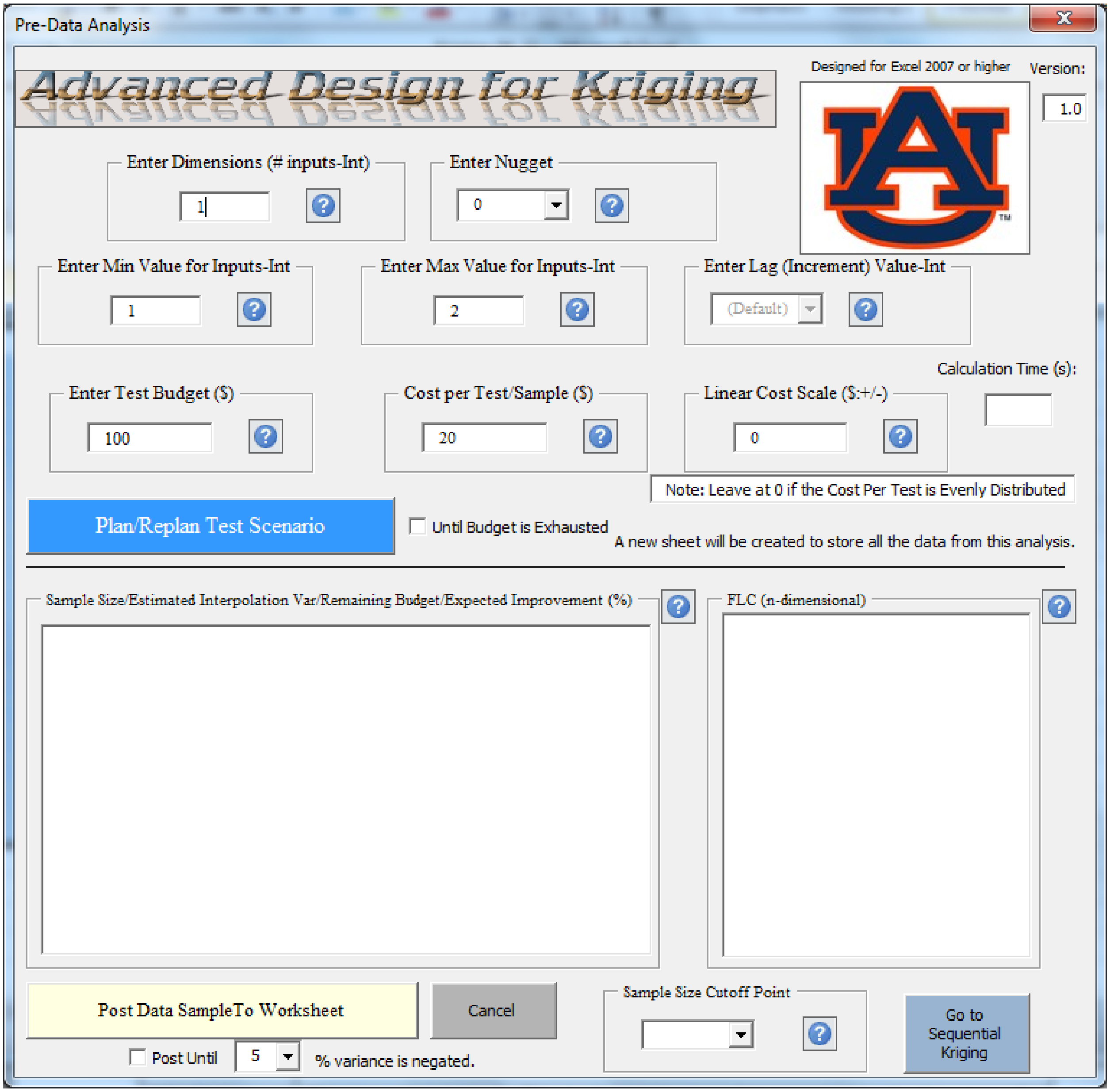


Figure 26: Advanced DFK Software - Test Planning

This screen requires user input prior to executing the "Plan/Replan Test Scenario" or the "Agree and Post Data To Worksheet for Sequential Kriging" with the latter button requiring that data in the two lists directly above it be populated by executing the code behind the "Plan/Replan Test

Scenario". There are a number of inputs that are required by the user. They are described in the table below.

Table 49: Test Planning User Input Table

Input	**Description**
Dimensions (d)	This input is required. This is the number of inputs to a system or the number of spatial dimensions.
Min Value	This input is required. This is the minimum value or starting point/location of the inputs. This must be greater than zero. For this software the minimum value for each input must be the same or balanced.
Max Value	This input is required. This is the maximum value or starting point/location of the inputs. This must be greater than zero. For this software the maximum value for each input must be the same or balanced.
Lag (h)	This input is optional and disabled.
Test Budget	This input is required. This input constrains the output from the analysis against a monetary figure (in dollars).
Cost per Test	This input is required. This input box provides the user to the ability to specific a monetary amount associated with each test/sample (in dollars).
Linear Scale (Ψ)	This input is optional. If left blank, then it is assumed that each test/sample cost an equal amount. This input is essentially a linear multiplier and can be positive or negative.

After the user has identified and entered the appropriate input information as described above, then the “Plan/Replan Test Scenario” button is pressed to execute calculations described throughout Chapters 3-6.

Upon completion of the processing, two list boxes will be populated with data. The left-most list box will contain the following data:

- sample size
- variance Estimation
- remaining budget (including over budget scenarios)
- and EI.

The right-most list box includes a list of initial FLCs, which is dimensional dependent. Users make a decision at this point as to output the data onto the spreadsheet for use in sequential Kriging or to adjust inputs and execute the planning code until satisfactory outputs are generated. If users choose to output data onto the spreadsheet, one of three outcomes is possible. The first possible outcome is the program will output all data. The second possible outcome is that the program will output only data that results in the test scenarios that do not exceed the total budget. The third option is to only output data until a certain amount of total variability has been negated.

Users should be aware that just because the budget has not been exhausted, this doesn’t imply that the correct sample size has been determined. The optimal sample size selection should be based more on the estimated variance. Users should also be aware that this initial variance estimation can underestimate the true variance of data. Therefore, users have to use judgment before proceeding to actual physical testing of a system. This type of analysis is consistent with most software applications as users should not assume results are always accurate, understand where they come from, and be able to make judgments based on data presented. After appropriate data have been obtained, users should execute the program again and proceed to the “Sequential Kriging”.

7.4 Sequential Kriging

To perform this portion of the advanced DFK software application, users should execute the software from the ribbon menu, and select the “Sequential Kriging” button. This brings users to the following GUI:

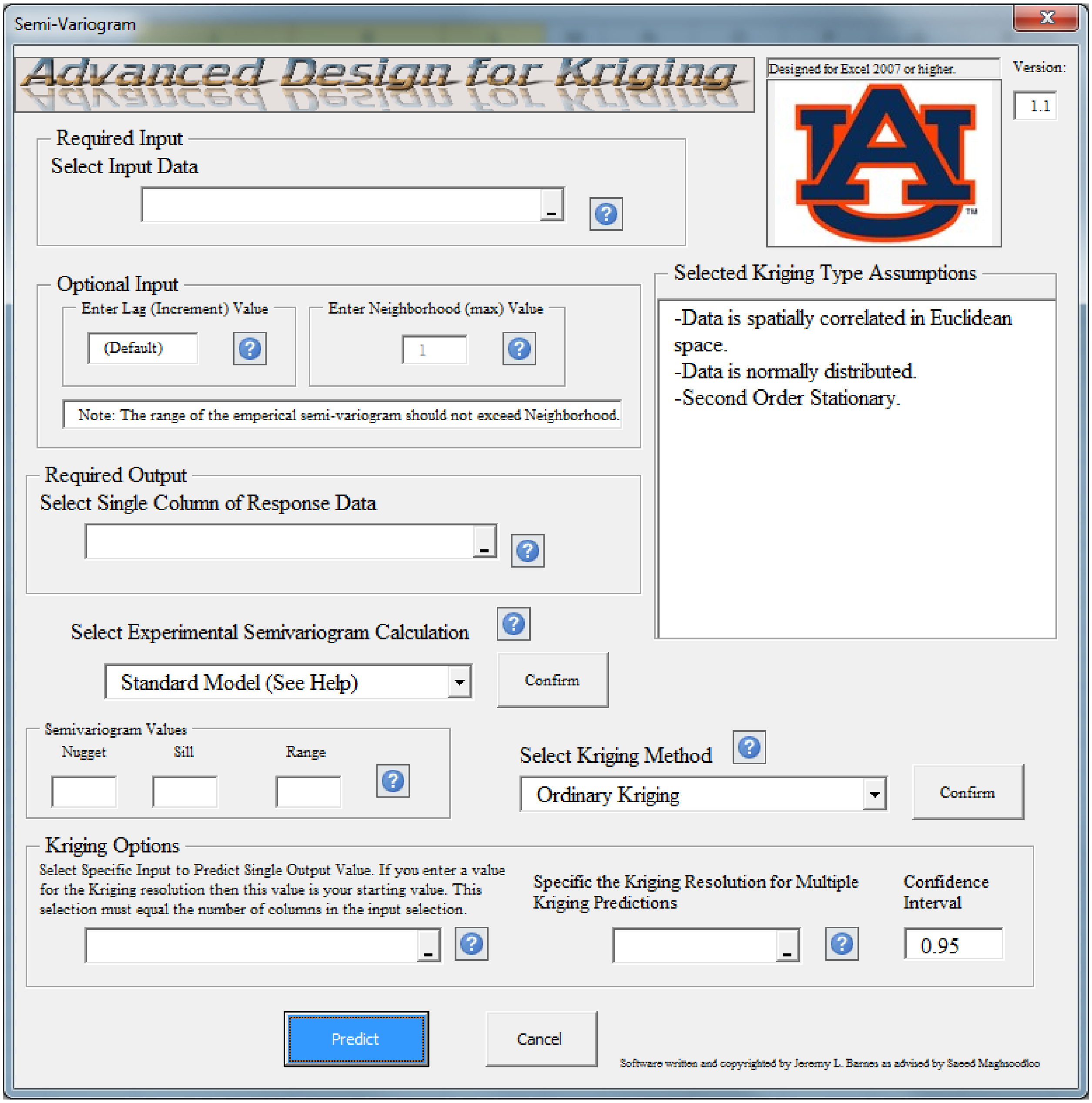


Figure 27: Advanced DFK Software - Sequential Kriging

After this portion of the program is loaded, users are required to select the inputs from a spreadsheet, either from the test planning portion of the software or from the user's own supplied data. The observed response data that accompanies the pilot sample is also required. Finally users must select the Kriging estimation point and the resolution (optional) for sequential Kriging. The software will perform an individual iteration where it estimates the next candidate input set and records them on a program generated spreadsheet. Users can also generate values to use as long as the format used in the program's self-generating sheet tool is not altered in any way. Confidence intervals are also given. The data is exported to a program generated sheet. The user may notice the following screen after the "Predict" button is pressed:

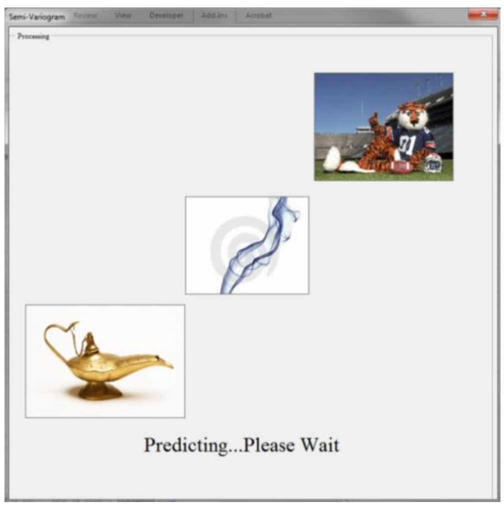


Figure 28: Advanced DFK Software – Processing

This is the nominal screen to indicate that code is being processed. After code execution has completed, users are presented with the following output screen:

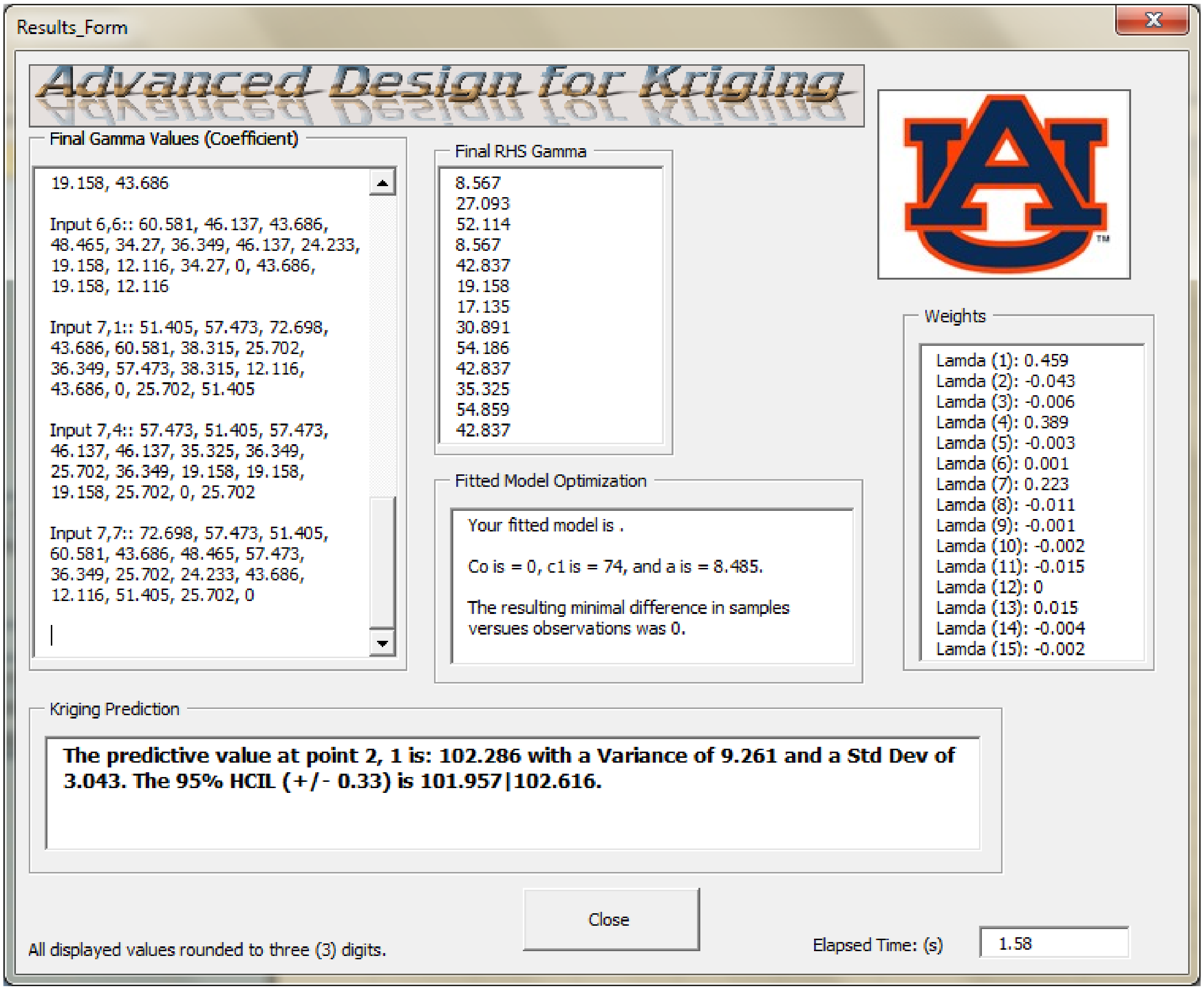


Figure 29: Advanced DFK Software – Output

The user can utilize this information as needed to perform analysis.

7.5 Data Analysis and Recording

Data calculations are recorded. This allows users to perform further analyses based on the generated datasets. In this research, these data are used to perform MC validation in such a manner as to make inferences about use of the advanced DFK process for the application of validation.

7.6 Conclusions

The software developed for this research presents users with a unique capability to perform analyses currently unavailable or overly time-consuming by hand-calculations. The software is easily understandable and user-friendly. Help is presented throughout the program to assist users in understanding all capabilities that are presented. A streamlined GUI was designed and developed to further make the software application available to a larger audience. Methodology and formulation presented in this dissertation are utilized as the underlying code. The code base is presented in Appendix A for review and understanding.

CHAPTER 8

Conclusions and Proposed Future Research

Results of this research have shown the extension, expansion, and augmentation of the "Customized Sequential Designs for Random Simulation Experiments: Kriging Metamodeling and Bootstrapping" process introduced by van Beer's. Introduction of Design of Experiments (DOX) with Kriging is an expanding field of research with many areas of optimization and application. This thesis reiterates the phrase Design for Kriging (DFK) recently coined by [52] as it accurately and precisely describes the intent of the process. It summarizes extensions of the DFK process including initial sample size determination, space filling designs for Kriging, optimization of sequential sampling methods, and inclusion of a standard variogram model. It also introduces test planning and validation applications that take advantage of DFK methodology. Summarized details of our conclusions are found in the following section.

8.1 Conclusions

Based on the data analysis and results presented in Chapter 6, a number of conclusions are drawn. Through our standardized variogram model, we are able to minimize the error between an experimental variogram model and our model thus allowing for accurate Kriging predictions. Secondly, we presented a method of utilizing the DFK software in order to determine initial sample size and sample location determination. Multiple summarized tables of data were constructed to demonstrate these results. Additionally, we demonstrated a trend based on pilot design variance reduction for additional budget considerations that directly correspond to our results in Section 6.2. As an important but closing note, we discuss validation utilizing DFK software when limited empirical data are present. It was found, through the use of the K-S test along with DFK software, that the additional data provided by the software allowed us to more closely scrutinize the differences between two datasets. This allowed us to make claims about MC models that were designed to represent such systems.

8.2 Future Work

The work herein and especially development of the advanced DFK software has opened further avenues for future research, which can determine the feasibility of a permanent computer learning model adaptation such as in neural networks or central database storage of pilot designs. This would highly benefit the overall process and software. The software could recognize exact or similar problems as has been performed previously and perform decision analysis on the factor space without having to perform all required DFK calculations. Another research topic is determining the feasibility of the process to be multi-threaded into processing individual Kriging instances for increased speed and accuracy. Porting the software into a platform that is more suitable for multi-threading is almost certainly required. Future research may include advanced selection of FLCs including non-integer values. A potential future advancement is to expand the software inputs to include distributions as inputs and incorporation of separate ranges per input thus introducing non-blocked designs. Kriging with multiple response variables can also be studied for feasibility. This exploratory study could open avenues for future Kriging research and software advancement. Additionally, the incorporation of weight standardization to correct for negative Kriging weights would allow the DFK software to be more robust. Finally, Kriging with real-time determination of neighborhoods and separation of neighborhoods (possibly into different threads) would allow the overall process to be more efficient and produce more accurate results. The study of multiple simultaneous neighborhood processing would be a large leap forward in the current software design and would require a large research effort. A mathematical study to precisely identify the next FLC during the test planning phase would eliminate the search algorithm used in the code base, thus tremendously decreasing the computational time in creating new designs. Further, a more detailed study into the standard variogram model such as nonparametric regression could provide a more robust method of arriving at accurate variograms.

References

[1] J. Kleijnen and W. van Beers, "Customized Sequential Designs for Random Simulation Experiments: Kriging Metamodeling and Bootstrapping," *CentER*, vol. 2005–55, p. 33, Mar. 2005.

[2] M. Bachmaier and M. Backes, "Variogram or semivariogram? Understanding the variances in a variogram," *Precision Agriculture*, vol. 9, no. 3, pp. 173–175, Feb. 2008.

[3] W. van Beers, "Kriging Metamodeling for Simulation," Tilburg, Netherlands, 2004.

[4] J. Kleijnen and W. van Beers, "Kriging for Interpolation in Random Simulation," *CentER*, vol. 2001-74, 2001.

[5] D. Dahlem and W. Harrison, "Globally Optimal Multi-agent Reinforcement Learning Parameters in Distributed Task Assignment," in *Web Intelligence and Intelligent Agent Technologies, 2009. WI-IAT '09. IEEE/WIC/ACM International Joint Conferences on*, 2009, vol. 2, pp. 28–35.

[6] R. B. Gramacy, H. K. H. Lee, and W. G. Macready, "Parameter space exploration with Gaussian process trees," *Proceedings of the twenty-first international conference on Machine learning*, p. 45–, 2004.

[7] R. Webster and M. Oliver, *Geostatistics for Environmental Scientists*. John Wiley & Sons, LTD., 2001.

[8] E. Isaaks and R. Srivastava, *An Introduction to Applied Geostatistics*. Oxford University Press, Inc., 1989.

[9] R. Olea, *Geostatistics for Engineers and Earth Scientists*. Kluwer Academic Publishers, 1999.

[10] N. Cressie, *Statistics for Spatial Data*. Wiley-Interscience, 1993.

[11] O. Balci, "A Methodology for Certification of Modeling and Simulation Applications," *TOMACS*, vol. 7, no. 4, pp. 352 – 377, 2001.

[12] R. Sargent, "A Tutorial on Verification and Validation of Simulation Models," in *Proceedings of the 39th conference on Winter simulation*, Washington, D.C., United States, 2007.

[13] J. Kleijnen and W. Van Groenendaal, *Simulation: A Statistical Perspective*. Baffins Lane, Chichester, West Sussex PO19 1UD, England: John Wiley & Sons, LTD., 1992.

[14] A. Forrester, "Black-Box Calibration for Complex-System Simulation," *Philosophical Transactions of the Royal Society*, vol. 368, pp. 3567–3579, Jul. 2010.

[15] B. Ankenman, B. Nelson, and J. Staum, "Stochastic Kriging for Simulation Metamodeling," in *Proceedings of the 40th Conference on Winter Simulation*, Miami, Florida, United States, 2008, pp. 362 – 370.

[16] G. Matheron, "Principles of Geostatistics," *Economic Geology*, vol. 58, no. 8, pp. 1246 – 1266, Dec. 1963.

[17] H. O. A. Wold, *A Study in the Analysis of Stationary Time Series*, 2nd ed. Almqvist & Wiksell, 1954.

[18] R. H. Bishop and I.-T. I., Systems, and Automation Society, *Sur 1'interpolation et Extrapolation des Suites Stationnaires*. CRC Press, 2002.

[19] A. N. Kolmogorov, "Dissipation of Energy in the Locally Isotropic Turbulence," *Proceedings: Mathematical and Physical Sciences*, vol. 434, no. 1890, pp. 15–17, Jul. 1991.

[20] N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series: With Engineering Applications*. The MIT Press, 1964.

[21] H. Liu, "Taylor Kriging Metamodeling for Simulation Interpolation, Sensitivity Analysis and Optimization," Dissertation, Auburn, Auburn, AL, 2009.

[22] D. G. Krige, *Lognormal-De Wijsian Geostatistics for Ore Evaluation*, 2nd Revised ed. South African Institute of Mining and Metallurgy, 1981.

[23] J. K. Ord, "Kriging, Entry," *Encyclopedia of Statistical Sciences*, vol. 4. John Wiley & Sons, Inc., New York, pp. 411–413, 1983.

[24] P. Hemyari, "Analytical Solution for Punctual Kriging in One Dimension.".

[25] N. Cressie, “The Origins of Kriging,” *Mathematical Geology*, vol. 22, no. 3, pp. 239–252, Apr. 1990.

[26] I. Clark, *Practical Geostatistics*. Appliced Science Publishers LTD, 1979.

[27] I. Clark and W. Harper, *Practical Geostatistics 2000*. Ecosse North America LLC., 2000.

[28] W. van Beers, “Kriging Metamodeling in Discrete-Event Simulation: An Overview,” in *Proceedings of the 37th conference on Winter simulation*, Orlando, Florida, United States, 2005, pp. 202 – 208.

[29] P. Goovaerts, “Factorial Kriging Analysis: A Useful Tool for Exploring the Structure of Multivariate Spatial Soil Information,” *Journal of Soil Science*, vol. 43, no. 4, pp. 597–619, Dec. 1992.

[30] A. P. Reis, A. J. Sousa, E. Ferreira da Silva, C. Patinha, and E. C. Fonseca, “Combining Multiple Correspondence Analysis with Factorial Kriging Analysis for Geochemical Mapping of the Gold–Silver Deposit at Marrancos (Portugal),” *Applied Geochemistry*, vol. 19, no. 4, pp. 623–631, Apr. 2004.

[31] M. W. Davis and P. G. Culbane, “Contouring Very Large Data Sets Using Kriging,” *Geostatistics for Natural Resources Characterization*, vol. 2, 1984.

[32] A. G. Royle, F. L. Clausen, and P. Frederiksen, “Practical Universal Kriging and Automatic Contouring,” *Elsevier scietific publishing company*, vol. 1, pp. 377–394, 1981.

[33] F. Trochu, “A Contouring Program Based on Dual Kriging Interpolation,” *Engineering with Computers*, vol. 9, no. 3, pp. 160–177, Sep. 1993.

[34] S. De Iaco, D. E. Myers, and D. Posa, “Space-time Variograms and a Functional Form for Total Air Pollution Measurements,” vol. 41, no. 2, pp. 311–328, 2002.

[35] R. L. Bras and I. Rodriguez-Iturbe, *Random Functions and Hydrology*. Dover Publications, 1994.

[36] S. Cheng and R. Wang, “An Approach for Evaluating the Hydrological Effects of Urbanization and its Application,” *Hydrological Processes*, vol. 16, no. 7, pp. 1403–1418, May 2002.

[37] B. André, F. Trochu, and J. Dansereau, “Approach for the Smoothing of Three-Dimensional Reconstructions of the Human Spine Using Dual Kriging Interpolation,” *Med Biol Eng Comput*, vol. 34, no. 3, pp. 185–191, May 1996.

[38] S. Delorme, Y. Petit, J. A. de Guise, C.-E. Aubin, H. Labelle, C. Landry, and J. Dansereau, “Three-Dimensional Modelling and Rendering of the Human Skeletal Trunk from 2D Radiographic Images,” in *Second International Conference on 3-D Digital Imaging and Modeling, 1999. Proceedings*, 1999, pp. 497–505.

[39] P. De Doncker, X. Cognet, J. Dricot, R. Meys, M. Hélier, and W. Tabbara, “Electromagnetic Wave Propagation Prediction Using Spatial Statistics: Experimental validation,” in *Proc. of the 9th Symp. on Communications and Vehicular Technology in the Benelux*, 2002.

[40] J. Dricot, R. Meys, M. Hélier, and W. Tabbara, “Kriging the Fields: A New Statistical Tool for Wave Propagation Analysis,” in *Proc. of the Int. Conf. on Electromagnetics in Advanced Applications*, 2003.

[41] J. Echaabi, F. Trochu, and R. Gauvin, “A General Strength Theory for Composite Materials Based on Dual Kriging Interpolation,” *Journal of Reinforced Plastics and Composites*, vol. 14, no. 3, pp. 211 –232, Mar. 1995.

[42] G. Kamanayo, F. Trochu, and B. Sanschagrin, “Prediction of Shrinkage by Dual Kriging for Injection Molded Polypropylene Plaques,” *Advances in Polymer Technology*, vol. 13, no. 4, pp. 305–314, Dec. 1994.

[43] A. Limaiem and H. A. ElMaraghy, “Automatic Inference of Parametric Equations in Geometric Modeling Using Dual Kriging,” in *, 1996 IEEE International Conference on Robotics and Automation, 1996. Proceedings*, 1996, vol. 2, pp. 1499–1504 vol.2.

[44] P. McLean, P. Léger, and R. Tinawi, “Post-Processing of Finite Element Stress Fields Using Dual Kriging Based Methods for Structural Analysis of Concrete Dams,” *Finite Elements in Analysis and Design*, vol. 42, no. 6, pp. 532–546, Mar. 2006.

[45] R. A. V. Rossel, P. Goovaerts, and A. B. McBratney, “Assessment of the Production and Economic Risks of Site-Specific Liming Using Geostatistical Uncertainty Modelling,” *Environmetrics*, vol. 12, no. 8, pp. 699–711, 2001.

[46] R. Chaveesuk and A. E. Smith, *Dual kriging: an Exploratory Use in Economic Metamodeling: An Article from: Engineering Economist*. Thomson Gale, 2005.

[47] R. R. Barton, “Metamodeling: A State of the Art Review,” in *Simulation Conference Proceedings, 1994. Winter*, 1994, pp. 237– 244.

[48] J. Kleijnen and W. van Beers, “Robustness of Kriging when interpolating in random simulation with heterogeneous variances: Some experiments,” *ejor*, vol. 165, no. 2005, pp. 826–834, 2005.

[49] H. Gunes, H. E. Cekli, and U. Rist, “Data Enhancement, Smoothing, Reconstruction and Optimization by Kriging Interpolation,” in *Proceedings of the 40th Conference on Winter Simulation*, Miami, Florida, United States, 2008, pp. 379 – 386.

[50] T. Simpson, T. Mauery, J. Korte, and F. Mistree, “Comparison of Response Surface and Kriging Models for Multidisciplinary Design Optimization,” Penn State.

[51] W. Biles, J. Kleijnen, W. van Beers, and I. van Nieuwenhuyse, “Kriging Metamodeling in Constrained Simulation Optimization: An Explorative Study,” in *Proceedings of the 39th conference on Winter simulation*, Washington, D.C., United States, 2007, pp. 355 – 362.

[52] J. Kleijnen, *Design and analysis of simulation experiments*. New York: Springer, 2008.

[53] J. R. Koehler and A. B. Owen, *Computer Experiments. Handbook of Statistics*, vol. 13. S. Gosh and C.R. Rao, 1996.

[54] J. Sacks, W. Welch, T. Mitchell, and H. Wynn, “Design and Analysis of Computer Experiments,” *JSTOR*, vol. 4, no. 4, pp. 409–423, 1989.

[55] T. J. Mitchell and M. D. Morris, “The Spatial Correlation Function Approach to Response Surface Estimation,” in *Proceedings of the 24th conference on Winter simulation*, New York, NY, USA, 1992, pp. 565–571.

[56] J. Régnière and A. Sharov, “Simulating Temperature-Dependent Ecological Processes at the Sub-Continental Scale: Male Gypsy Moth Flight Phenology as an Example,” *International Journal of Biometeorology*, vol. 42, no. 3, pp. 146–152, Feb. 1999.

[57] J. Kleijnen and W. van Beers, “Application-Driven Sequential Designs for Simulation Experiments: Kriging Metamodeling,” *CentER*, vol. 2003–33.

[58] D. den Hertog, J. Kleijnen, and A. Siem, “The Correct Kriging Variance Estimated by Bootstrapping,” *JORS*, vol. 57, no. 4, pp. 400–409, 2006.

[59] N. Flipo, N. Jeannée, M. Poulin, S. Even, and E. Ledoux, "Assessment of nitrate pollution in the Grand Morin aquifers (France): Combined use of geostatistics and physically based modeling," *Environmental Pollution*, vol. 146, no. 1, pp. 241–256, Mar. 2007.

[60] W. Ren, M. Steurer, and S. Woodruff, "Accuracy evaluation in power hardware-in-the-loop (PHIL) simulation center for advanced power systems," in *Proceedings of the 2007 summer computer simulation conference,* San Diego, CA, USA, 2007, pp. 489–493.

[61] A. Ratle, "Kriging as a Surrogate Fitness Landscape in Evolutionary Optimization," *AI EDAM*, vol. 15, no. 01, pp. 37–49, 2001.

[62] D. de Frutos Escrig and C. Gregorio Rodríguez, "Constrained Simulations, Nested Simulation Semantics and Counting Bisimulations," *Electronic Notes in Theoretical Computer Science*, vol. 206, pp. 41–58, Apr. 2008.

[63] G. S. Fishman, *Monte Carlo: Concepts, Algorithms, and Applications*. Springer, 1996.

[64] R. Hill, J. Miller, and G. McIntyre, "Applications of Discrete Event Simulation Modeling to Military Applications," in *Proceedings of the 33rd Winter Simulation Conference*, Arlington, Virginia, United States, 2001, pp. 780 – 788.

[65] A. Law, *Simulation Modeling and Analysis*, Third ed. United States: McGraw-Hill, 2000.

[66] O. Balci, W. Ormsby, R. Nance, and J. Arthur, "Expanding our Horizons in Verification, Validation, and Accreditation Research and Practice," in *Proceedings of the 34th Winter Simulation Conference*, San Diego, California, United States, 2002, pp. 653 – 663.

[67] J. Arthur and R. Nance, "Independent Verification and Validation: A Missing Link in Simulation Methodology?," in *Proceedings of the 28th conference on Winter simulation*, Coronado, California, United States, 1996, pp. 230 – 236.

[68] Department of Defense, "Military Standard 499 System Engineering." Department of Defense, 06-May-1994.

[69] D. Caughlin, "An Integrated Approach to Verification, Validation, and Accreditation of Models and Simulations," in *Proceedings of the 32nd conference on Winter simulation*, Orlando, Florida, United States, 2000, pp. 872 – 881.

[70] L. Chwif, P. Sérgio Muniz Silva, and L. Mitio Shimada, "A Prescriptive Technique for V&V of Simulation Models When No Real-Life Data are Available," in *Proceedings of the*

38th Winter Simulation Conference, Monterey, California, United States, 2006, pp. 911 – 918.

[71] R. Cheng, W. Holland, and N. Hughes, “Selection of Input Models Using Bootstrap Goodness-of-Fit,” in *Proceedings of the 28th conference on Winter simulation*, Coronado, California, United States, 1996, pp. 199 – 206.

[72] L. Champagne and R. Hill, “Agent-model Validation Based on Historical Data,” in *Proceedings of the 39th conference on Winter simulation*, Washington, D.C., United States, 2007, pp. 1223 – 1231.

[73] J. Kleijnen, R. Cheng, and B. Bettonvil, “Validation of Trace-Driven Simulation Models: More on Bootstrap Tests,” *Management Science - INFORMS*, vol. 47, no. 11, pp. 1533–1558, Nov. 2001.

[74] W. van Beers and J. Kleijnen, “Kriging Interpolation in Simulation: A Survey,” in *Proceedings of the 36th conference on Winter simulation*, Washington, D.C., United States, 2004, pp. 113 – 121.

[75] D. Montgomery, *Design and Analysis of Experiments*, 6th ed. Arizona State University: Wiley and Sons, Inc., 2005.

[76] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap*. United States: Chapman & Hall, 1993.

[77] A. Baykasoğlu and G. Nabil, “A simulated annealing algorithm for dynamic layout problem,” *Computers & Operations Research*, vol. 31, no. 2, pp. 313–315, Feb. 2004.

[78] P. Bratley, B. L. Fox, and L. E. Schrage, *A Guide to Simulation*, 2nd ed. Springer, 1987.

[79] J. Kleijnen, *Handbook of Simulation*. New York: Wiley and Sons, Inc., 1998.

[80] O. Balci, *Methodology and Validation*, vol. 19. Simulation Councils, Inc., 1987.

[81] W. Kelton, R. Sadowski, and D. Sturrock, *Simulation with Arena*, Third ed. McGraw-Hill, 2003.

[82] A. C. BajPai, I. M. Calus, and J. A. Fairley, *Statistical Methods for Engineers and Scientists*. Great Britain: John Wiley & Sons, Inc., 1978.

[83] J. Kleijnen, *Statistical Techniques in Simulation: Part 1*, vol. 9. United States: Marcel Dekker, Inc., 1974.

[84] J. Kleijnen, *Statistical techniques in simulation: Part II*. New York: M. Dekker, 1974.

[85] B. Nelson, *Stochastic Modeling: Analysis & Simulation*. United States: McGraw-Hill, 1995.

[86] W. Cauer, *Theorie der linearen Wechselstromschaltungen*, vol. 1. Leipzig: Akad. Verlags-Gesellschaft Becker und Erler, 1941.

[87] B. Zeigler, *Theory of modelling and simulation*. New York: Wiley, 1976.

[88] A. Bluman, *Probability demystified*. New York: McGraw-Hill, 2005.

[89] A. Bajpai, *Statistical methods for engineers and scientists : a students' course book*. Chichester ;;New York: Wiley, 1978.

[90] B. Efron, *The jackknife, the bootstrap, and other resampling plans*. Philadelphia Pa.: Society for Industrial and Applied Mathematics, 1982.

[91] M. Bartlett, *The statistical analysis of spatial pattern*. London ;New York: Chapman and Hall ;Wiley, 1975.

[92] H. Lambert, S. Vogl, A. Brewster, and K.-P. Dunn, "On Validating the Modeling and Simulation of a Generic Radar," MIT Lincoln Laboratory, Technical Report 1134, Feb. 2009.

[93] O. Balci, W. Ormsby, J. Car, and S. Saadi, "Planning for verification, validation, and accreditation of modeling and simulation applications," in *Proceedings of the 32nd conference on Winter simulation*, Orlando, Florida, United States, 2000, pp. 829 – 839.

[94] L. Champagne and R. Hill, "Simulation validation with historic outcomes," in *Proceedings of the 37th conference on Winter simulation*, Orlando, Florida, United States, 2005, pp. 1138 – 1147.

[95] J. Chew and C. Sullivan, "Verification, Validation, and Accreditation in the Life Cycle of Models and Simulations," in *Proceedings of the 32nd conference on Winter simulation*, Orlando, Florida, United States, 2000, pp. 813 – 818.

[96] D. Hartley, “Verification and Validation in Military Simulations,” in *Proceedings of the 29th conference on Winter simulation*, Atlanta, Georgia, United States, 1997, pp. 925 – 932.

[97] J. Carr, *Data Visualization in the Geosciences*. Prentice-Hall, Inc., 2002.

[98] J.-P. Chiles and P. Delfiner, *Geostatistics Modeling Spatial Uncertainty*. John Wiley & Sons, Inc., 1999.

[99] G. Box, *Empirical model-building and response surfaces*. New York: Wiley, 1987.

[100] D. Birkes, *Alternative methods of regression*. New York: Wiley, 1993.

[101] M. Schimek, *Smoothing and regression : approaches, computation, and application*. New York: Wiley, 2000.

[102] J. Fox, *Regression diagnostics*. Newbury Park Calif.: Sage Publications, 1991.

[103] S. Karlin, *Studies in spline functions and approximation theory*. New York: Academic Press, 1976.

[104] L. Sakhnovich, *Interpolation theory and its applications*. Dordrecht ;;Boston ;;London: Kluwer academic publ., 1997.

[105] S. Amari, *Methods of information geometry*. Providence RI: American Mathematical Soc., 2000.

[106] F. Chaitin-Chatelin, *Spectral approximation of linear operators*. New York: Academic Press, 1983.

[107] N. Cressie, “Fitting Variogram Models by Weighted Least Squares,” *Journal of the International Association for Mathematical Geology*, vol. 17, no. 5, pp. 563–586, Jul. 1985.

Appendix

Appendix A: Software Code Base

Appendix A contains the software application code that was used to automate the DFK process. The appendix is divided up into multiple sections. The sections include the main Kriging processing module, the test planning module, and the associated code that runs behind all the forms that is seen through the GUI while using the software. A description of how to use the software and how to install the software was previously presented in Chapter 7.

Appendix A.1: Main Kriging Module

```
'****************************************************************
'Advanced Design for Kriging
'Created by: Jeremy L. Barnes
'Contact Information: jeremylbarnes@hotmail.com
'Auburn University
'Date: 20 June 2012
'Revision 1.1
'Revision Log:
'Revision 1.1 Added the standard model selection for the variogram
'Revision 1.0 Cleaned up code to remove universal Kriging. A backup copy with
the code is maintained
'Revision 0.9 Added the entire test planning module to the software
'Rev 0.8 Fixed Multi Dimension Universal Kriging and Kriging with resolution
'Rev 0.7 Added Lag and Neighborhood
'Rev 0.6 Added Data Recording and Manual Selection of Fitted Variograms
'Rev 0.5 Added Help
'Rev 0.4 Multi Dimension Kriging with code clean up
'Rev 0.3 Multi Dimension Kriging
'Rev 0.2 Single Dimension Kriging with Stochastic bug fix
'Rev 0.1 Single Dimension Kriging
'Beta Module
'This code contains 9 steps. The steps are explained throughout the code.
'No calculations are rounded, only the displayed results.
Option Explicit
Public resolution As Double
Public test_diff_v_counter As Integer
Public cnt_reg As Integer
Public model_name As String
```

```
Public ss_res_comp As Double
Sub pd_analyze_start()
    pd_analysis.version.Value = "1.0"
    Dim fill_var_box As Integer
    For fill_var_box = 5 To 95 Step 5
        pd_analysis.ComboBox1.AddItem fill_var_box
    Next fill_var_box
    pd_analysis.ComboBox1.ListIndex = 0
    pd_analysis.Show
End Sub
Sub kriging_start()
    Dim err_msg_1
    If Application.version < "12.0" Then
        err_msg_1 = MsgBox("The minimum required version of Excel is 2007.
The program will now terminate.", vbOKOnly, "Error Handler")
        Exit Sub
     End If
    Kriging_Intro.Show
End Sub
Sub ShowSemiForm()
    'Populate the Kriging Type Box in the GUI
    SemiForm.KrigType.Clear
    SemiForm.KrigType.AddItem "Ordinary Kriging"
    SemiForm.KrigType.Text = SemiForm.KrigType.List(0)
    'Populate the Exp_Variogram Calculation Box in the GUI
    SemiForm.exp_var_select.Clear
    SemiForm.exp_var_select.AddItem "Standard Model"
    SemiForm.exp_var_select.AddItem "Traditional with Parameter Tweaking"
    SemiForm.exp_var_select.AddItem "Linear"
    SemiForm.exp_var_select.AddItem "Exponential"
    SemiForm.exp_var_select.AddItem "Gaussian"
    SemiForm.exp_var_select.AddItem "Spherical"
    SemiForm.exp_var_select.Text = SemiForm.exp_var_select.List(0)
    'Version
    Dim rev As String
    rev = "1.1"
    SemiForm.ver_txt.Text = rev
    'Populate the Kriging Assumptions in the GUI
    SemiForm.KrigAssum.Text = "-Data is spatially correlated in Euclidean
space." & Chr(13) & "-Data is normally distributed." & Chr(13) & "-Second
Order Stationary."

'***************************************************************************
    'Step 1 - Utilize interface to gather data

'***************************************************************************
    CheckSolver
    SemiForm.Show
End Sub
Sub calc_semiv_model_value(OriginalSampleRange As String)
```

```
    '********Set Variables***************
    cnt_reg = 0
    test_diff_v_counter = 0
    model_name = ""
    ss_res_comp = 0
    '************************************
    SemiForm.Frame4.Visible = False
    SemiForm.Frame5.Visible = False
    SemiForm.Frame6.Visible = False
    SemiForm.Frame7.Visible = False
    SemiForm.Frame8.Visible = False
    SemiForm.Frame9.Visible = False
    SemiForm.Frame10.ZOrder (0)
    SemiForm.Frame10.Visible = True
    SemiForm.prog_label.Caption = "Initializing"
    DoEvents
    'SemiForm.Repaint
    'Determine start time
    Dim sngStart As Single, sngEnd As Single
    Dim sngElapsed As Single
    sngStart = Timer ' Get start time.
    'Set max_inp_output array equal to the range of the input/output values
    Dim max_inp_out_array_size As Integer
    max_inp_out_array_size =
Range(SemiForm.RefEditOriginalSampleRange.Value).Rows.Count
    '*************** Err Handling****************
    Dim err_msg_1
    If Range(SemiForm.RefEditOriginalSampleRange.Value).Rows.Count <>
Range(SemiForm.Refoutdata.Value).Rows.Count Then
        err_msg_1 = MsgBox("Your input and output values should have the same
number of rows. The program will now terminate.", vbOKOnly, "Error Handler")
        Exit Sub
    End If
    If SemiForm.Predict_Input.Value = "" Then
        err_msg_1 = MsgBox("You did not select a Kriging Prediction input.
The program will now terminate.", vbOKOnly, "Error Handler")
        Exit Sub
    End If
    If Range(SemiForm.RefEditOriginalSampleRange.Value).Columns.Count <>
Range(SemiForm.Predict_Input.Value).Columns.Count Then
        err_msg_1 = MsgBox("Your input dimensions do not match the dimensions
of your Kriging estimate. The program will now terminate.", vbOKOnly, "Error
Handler")
        Exit Sub
    End If
     If SemiForm.Mult_Pred.Value <> "" Then
        If Range(SemiForm.Mult_Pred.Value).Rows.Count > 1 Or
Range(SemiForm.Mult_Pred.Value).Columns.Count > 1 Then
```

```
            err_msg_1 = MsgBox("Your resolution dimensions exceeded one or
you selected more than a single number. The program will now terminate.",
vbOKOnly, "Error Handler")
            Exit Sub
        End If
    Dim Ins As Long
        Ins = InStr(1, SemiForm.RefEditOriginalSampleRange.Value, ":")
        SemiForm.Predict_Input.Value =
Left(SemiForm.RefEditOriginalSampleRange.Value, Ins - 1)
    End If
    '***************End Err Handling****************
    'Create a new sheet for data dumps and calculation results
    Dim NewBook1 As New Worksheet
    Application.DisplayAlerts = False
    On Error Resume Next
    Worksheets("Krig_Dump").Delete
    Worksheets("Kriging_Regression").Delete
    Application.DisplayAlerts = True
    Set NewBook1 = Worksheets.Add
    NewBook1.Name = "Krig_Dump"
    '************************************************************************
    'Step 2 - Sort Data
    'Parse Input Data before using to Sort
    Dim Addx, fAddx, outAddx, lAddx, Addx_final As String
    Dim Wkb As String
    Dim Wks As String
    Ins = InStr(1, SemiForm.RefEditOriginalSampleRange.Value, "]")
    If Ins = 0 Then
       Wkb = ActiveWorkbook.Name
    Else
       Wkb = Mid(SemiForm.RefEditOriginalSampleRange.Value, 2, Ins - 1)
    End If
    Ins = InStr(1, SemiForm.RefEditOriginalSampleRange.Value, "!")
    Wks = Left(SemiForm.RefEditOriginalSampleRange.Value, Ins - 1)
    Addx = Mid(SemiForm.RefEditOriginalSampleRange.Value, Ins + 1,
Len(SemiForm.RefEditOriginalSampleRange.Value) - Ins)
    outAddx = Mid(SemiForm.Refoutdata.Value, Ins + 1,
Len(SemiForm.Refoutdata.Value) - Ins)
    Dim rng, RngO As Range
    Dim r, c, co
    Dim LastRow As String
    Set rng = Workbooks(Wkb).Worksheets(Wks).Range(Addx)
    Set RngO = Workbooks(Wkb).Worksheets(Wks).Range(outAddx)
    r = rng.Row
    c = rng.Column
    LastRow = Last(1, RngO)
    co = RngO.Column
    Dim intI As Integer
    Dim cstring, cstringO As String
```

```
    'Limitation is that the column for the input can not be greater than the
Z coulumn in excel
    For intI = 0 To 25
        If c - 1 = intI Then
            cstring = Chr$(97 + intI)
            Exit For
        End If
    Next
    'Create the partial input range for use
    fAddx = cstring & r
    For intI = 0 To 25
        If co - 1 = intI Then
            cstringO = Chr$(97 + intI)
            Exit For
        End If
    Next
    'Create the partial output range
    lAddx = cstringO & LastRow
    'Create the final range for sorting
    Addx_final = fAddx & ":" & lAddx
    'Create the entire range to use for sorting
    fAddx = fAddx & ":" & cstring & co
    'Data has been parsed. Sort Data. This is an excel function.
    Range(Addx).Select
    ActiveWorkbook.Worksheets(Wks).Sort.SortFields.Clear
    ActiveWorkbook.Worksheets(Wks).Sort.SortFields.Add Key:=Range(fAddx), _
        SortOn:=xlSortOnValues, Order:=xlAscending, DataOption:=xlSortNormal
    'Loop to sort on multiple columns
    Ins = InStr(1, Addx, ":")
    If Mid(Addx, Ins + 2) <> UCase(cstring) Then
        For intI = c To (co - 1)
            cstring = Chr$(97 + intI)
            fAddx = cstring & r & ":" & cstring & co
            ActiveWorkbook.Worksheets(Wks).Sort.SortFields.Add
Key:=Range(fAddx), _
                SortOn:=xlSortOnValues, Order:=xlAscending,
DataOption:=xlSortNormal
        Next
    End If
    'Perform the final sort after the multiple keys have been generated
    With ActiveWorkbook.Worksheets(Wks).Sort
        .SetRange Range(Addx_final)
        .Header = xlNo
        .MatchCase = False
        .Orientation = xlTopToBottom
        .SortMethod = xlPinYin
        .Apply
    End With
    'Populate array with sorted data from the selected I/O ranges
```

```
    'Example inp_out_data(1,1) is the first input and inp_out_data(1,2) is
the output
    Dim inp_out_data() As Double
    ReDim inp_out_data(1 To max_inp_out_array_size, 1 To 2)
    Dim ii As Integer
    Dim det_dim As Integer
    Dim dim_dup_inp As Integer
    Dim dup_count As Integer
    'Determine dimensions
    det_dim = Range(SemiForm.RefEditOriginalSampleRange.Value).Columns.Count
    'Input
    'Check for duplicate input
    ii = 0
    Dim refedit_result As Variant
    refedit_result = SemiForm.RefEditOriginalSampleRange.Value
    For ii = 1 To (max_inp_out_array_size * det_dim) Step det_dim
        dup_count = 0
        For dim_dup_inp = 0 To (det_dim - 1)
            If Range(refedit_result).Item(ii + dim_dup_inp) =
Range(refedit_result).Item(ii + det_dim + dim_dup_inp) Then
                dup_count = dup_count + 1
                If dup_count = det_dim Then
                    err_msg_1 = MsgBox("You have duplicate inputs. The
program will now terminate.", vbOKOnly, "Error Handler")
                    Exit Sub
                End If
            Else
                Exit For
            End If
        Next
    Next
    'Populate Output Array.
    For ii = 1 To max_inp_out_array_size
        inp_out_data(ii, 2) =
Range(SemiForm.Refoutdata.Value).Cells(ii).Value
    Next
    '*************Err handling*********
    Dim dim_err_count As Integer
    For dim_err_count = 1 To det_dim
        If Range(SemiForm.Predict_Input.Value).Cells(1, dim_err_count).Value
>
Range(SemiForm.RefEditOriginalSampleRange.Value).Cells(Range(SemiForm.RefEdit
OriginalSampleRange.Value).Rows.Count, dim_err_count).Value Or
Range(SemiForm.Predict_Input.Value).Cells(1, dim_err_count).Value <
Range(SemiForm.RefEditOriginalSampleRange.Value).Cells(1,
dim_err_count).Value Then
            err_msg_1 = MsgBox("The Kriging Prediction number can not be
higher (or lower) than the largest (smallest) input value. The program will
exit now.", vbOKOnly, "Error Handler")
            Exit Sub
```

```
        End If
    Next
    '*************End Err handling*****
    '************************************************************************
    'Step 3 - Calculate distance number of pairs, distance bewteen pairs, and
difference in measured value
    'Logic developed by Sabahattin (Gohkan) Ozden in php and translated into
vba by Jeremy Barnes
    'Calculate distances between and group equal distances together
    Dim a_count As Integer
    Dim b_count As Integer
    Dim c_count As Integer
    Dim flag As Integer
    Dim max_array_size As Integer
    'The plus 1 is added to account for the 0 difference which is always
present.
    max_array_size =
(Range(SemiForm.RefEditOriginalSampleRange.Value).Rows.Count *
(Range(SemiForm.RefEditOriginalSampleRange.Value).Rows.Count - 1) / 2) + 1
    Dim diff_array() As Double
    ReDim diff_array(1 To 1)
    'Index is unique identifier of each set
    Dim num_diff_array() As Double
    ReDim num_diff_array(1 To 1)
    Dim out_array() As Double
    'First index is unique identifier of each set. Second index is the
different output values inside the set
    ReDim out_array(1 To max_array_size, 1 To max_inp_out_array_size)
    'Three counters for use in determining the differnces in the data set
    a_count = 0
    b_count = 0
    c_count = 0
    Dim inp_out_counter
    'Inital set of diff_array will always be zero
    diff_array(1) = -1
    'For next from 1 to selectd I/O range
    '*************************************
    Dim det_dim_s As String
    'det_dim_s = Chr(65 + det_dim) & 1
    Worksheets("Krig_Dump").Range("A1").Value = "Lag Frequency"
    Worksheets("Krig_Dump").Range("B1").Value = "FLC Lags (" & det_dim & "D)"
    Worksheets("Krig_Dump").Range("C1").Value = "RHS Lags"
    '*************************************
    Dim dim_count As Integer
    Dim dim_count_1 As Integer
    Dim euc_cal As Double
    For a_count = 1 To max_inp_out_array_size
        For b_count = 1 To a_count
            'c_count has the unique identifier for the differences
            c_count = 1
```

```
            'Flag is used to determine if a unique set of difference data
already exists
            flag = 0
            Do While c_count <= test_diff_v_counter
                'If set exists add new input output variable
                'Calculate all Euclidean distances by looping through each
dimension
                'Euclidean calculation sqrt(SUM(1 to n) (psubi - qsubi)^2
                euc_cal = euc_values(a_count, b_count, det_dim)
                'Floating comparision is impossible. Must convert to string
for accurate comparision.
                If Str(diff_array(c_count)) = Str(euc_cal) Then
                    'Output difference array data to Krig Dump sheet
                    'ActiveCell.FormulaR1C1 = diff_array(c_count)
                    'Store number of pairs
                    num_diff_array(c_count) = num_diff_array(c_count) + 1
                    'Store output difference to be used in the Exp-
semiovariogram
                    '**************************************
                    'Put Input differences onto krig_dump sheet
                    Worksheets("Krig_Dump").Range("B1").Offset(c_count, 0) =
diff_array(c_count)
                    Worksheets("Krig_Dump").Range("A1").Offset(c_count, 0) =
num_diff_array(c_count)
                    '**************************************
                    'Dynamic REDIM
                    If num_diff_array(c_count) >=
WorksheetFunction.Max(num_diff_array) Then
                        ReDim Preserve out_array(1 To max_array_size, 1 To
WorksheetFunction.Max(num_diff_array))
                    End If
                    out_array(c_count, num_diff_array(c_count)) =
Abs(inp_out_data(a_count, 2) - inp_out_data(b_count, 2))
                    flag = 1
                    Exit Do
                End If
                c_count = c_count + 1
            Loop
            If flag = 0 Then
                'There is no set. Create one.
                euc_cal = euc_values(a_count, b_count, det_dim)
                'Redim for only the used array elements therefore providing
efficient code.
                ReDim Preserve diff_array(1 To c_count)
                ReDim Preserve num_diff_array(1 To c_count)
                diff_array(c_count) = euc_cal
                'Store number of pairs
                num_diff_array(c_count) = 1
                'Increment set counter
                test_diff_v_counter = test_diff_v_counter + 1
```

```
                'Store output difference to be used in the Exp-semiovariogram
                out_array(c_count, num_diff_array(c_count)) =
Abs(inp_out_data(a_count, 2) - inp_out_data(b_count, 2))
                '************************************
                'Put Input differences onto krig_dump sheet
                Worksheets("Krig_Dump").Range("B1").Offset(c_count, 0) =
diff_array(c_count)
                Worksheets("Krig_Dump").Range("A1").Offset(c_count, 0) =
num_diff_array(c_count)
                '************************************
            End If
        Next
    Next
    'Memory management. This code Redims the inp_1, inp_2, and out_array
therefore eliminating empty elements.
    'Idea taken from http://www.xtremevbtalk.com/showthread.php?t=82476
    'Counters
    Dim m As Integer
    Dim n As Integer
    Dim iTemp() As Double  'Temporary array
    ReDim iTemp(1 To test_diff_v_counter, 1 To
WorksheetFunction.Max(num_diff_array))
    'Copy original array into temp array:
    For m = 1 To test_diff_v_counter    'Loop for 1st dimension
        For n = 1 To WorksheetFunction.Max(num_diff_array)  'Loop for output
            iTemp(m, n) = out_array(m, n)
        Next n
    Next m
    'Put values back from temporary array
    ReDim out_array(1 To test_diff_v_counter, 1 To
WorksheetFunction.Max(num_diff_array))
    For m = LBound(iTemp, 1) To UBound(iTemp, 1)       'Loop for 1st dimension
        For n = LBound(iTemp, 2) To UBound(iTemp, 2)   'Loop for 2nd dimension
            out_array(m, n) = iTemp(m, n)
        Next n
        '************************************
        'Put output array differences into the new worksheet
        Worksheets("Krig_Dump").Range("C1").Offset(m, 0) = out_array(m, 1)
        '************************************
    Next m
    'This statements purges the original inp_out array since it is not used
in the code anymore therefore making the code more efficient.
    Erase iTemp

'**************************************************************************
    Dim emp_semiv() As Double
    Dim summation_var As Double
    Dim j As Integer
    SemiForm.prog_label.Caption = "Empirical Variogram"
    DoEvents
```

```
    If SemiForm.exp_var_select.Text = "Standard Model" Then
        'Error Check
        If test_diff_v_counter >= 100 Then
            err_msg_1 = MsgBox("The problem contains more than 100 unique
lags, which will max out the Excel Solver. Switching to Traditional with
Parameter Tweaking.", vbOKOnly, "Error Handler")
            SemiForm.exp_var_select.Text = "Traditional with Parameter
Tweaking"
        End If
    End If
    If SemiForm.exp_var_select.Text = "Traditional with Parameter Tweaking"
Then
        'Determine lag prior to calculating the experimental semivariogram
        Dim bin As Double
        If SemiForm.ComboBoxlag.Value <> "(Default)" Then
            bin = CDbl(SemiForm.ComboBoxlag.Value)
        End If
        'Step 4 - Calculate exp semi gamma values using 1/2n(h)*SUM(1 to
n(h))[var(difference in values)]
        'Set the emp_semiv array to the number of elements in the diff_array
because for each individual difference there will be 1 exp-semi point
        ReDim emp_semiv(1 To UBound(diff_array, 1))
        'This variable is the SUM(1 to n(h))[var(difference in values)] part
of the exp-semi equation
        'Counters
        'Start with 2 instead of 1 because 1 is simply the 0 difference array
and all semi-variogram values would be calculated as 0.
        ii = 2
        j = 0
        Worksheets("Krig_Dump").Range(Chr(68) & ii - 1).Value = "Emp-Var
Value"
        Do
            det_dim_s = Chr(68) & (ii + 1)
            'Reset the summation variable for individual difference
            summation_var = 0
            'SUM(1 to n(h))[var(difference in values)]
            For j = 1 To num_diff_array(ii)
                summation_var = summation_var + (out_array(ii, j)) ^ 2
            Next
            'This is the 1/2n(h)* summation_var part of the equation
            emp_semiv(ii) = (1 / (2 * num_diff_array(ii))) * summation_var
            If ii = 2 Then Worksheets("Krig_Dump").Range(Chr(68) & ii).Value
= 0

            Worksheets("Krig_Dump").Range(det_dim_s).Value = emp_semiv(ii)
            ii = ii + 1
        Loop While ii <= test_diff_v_counter
        '***********Add Variogram
Chart****************************************
        ActiveSheet.Range("A" & test_diff_v_counter + 3).Select
        ActiveSheet.Shapes.AddChart.Select
```

```
        ActiveChart.ChartType = xlXYScatterSmoothNoMarkers
        ActiveChart.SeriesCollection.NewSeries
        ActiveChart.SeriesCollection(1).Name = "=""Emperical Variogram"""
        ActiveChart.SeriesCollection(1).XValues = "=Krig_Dump!$B$2:$B$" &
test_diff_v_counter + 1
        ActiveChart.SeriesCollection(1).Values = "=Krig_Dump!$D$2:$D$" &
test_diff_v_counter + 1

'**************************************************************************

'**************************************************************************
        'Step 5 and 6 - Calculate the fitted variogram model
        'This module calculates values of C0, C1, and a for the Spherical,
Expon, Linear, and Guassian semi-variogram models and returns recommendations
        'based on minimizing the distances between the obeservations and
samples.
        'C0 - nugget
        'C1 - Sil
        'a - range
        'Calculate C0 (nugget) through linear extrapolation
        'Eventually look into more advanced models of extrapolation which
means this needs to be calculate after the fitted
        'model has been calculated
        Dim nugget As Double
        'This uses the 1st and 2nd actual numbers in the exp_semiv i.e. it
excludes the first number which is always 0
        'nugget = emp_semiv(3) + ((0 - 2) / (1 - 2)) * (emp_semiv(2) -
emp_semiv(3))
        nugget = 0
        If nugget < 0 Then nugget = 0
        'Calculate models values with sil and range beginning from .1 up to
the max emperical value with a step of .1
        'The max emperical value is used since no fitted model calculation
will exceed that (prove this)
        'Sil and range variables
        Dim sill As Double
        Dim range_1 As Double
        'Fitted Model Variables
        Dim s_gamma() As Double
        ReDim s_gamma(1 To 1)
        Dim e_gamma() As Double
        ReDim e_gamma(1 To 1)
        Dim l_gamma() As Double
        ReDim l_gamma(1 To 1)
        'Model variables for determining the minimum value between the
exp_semivariogram and the fitted models
        Dim s_diff() As Double
        ReDim s_diff(1 To 1)
        Dim e_diff() As Double
        ReDim e_diff(1 To 1)
```

```
        Dim l_diff() As Double
        ReDim l_diff(1 To 1)
        Dim s_diff_sum As Double
        Dim e_diff_sum As Double
        Dim l_diff_sum As Double
        'Comp gets the min of the least squares fit of each model
        Dim comp, comp_1 As Double
        Dim i As Double
        'This is the counter where we will loop from 1 to the total number of
emp_semi elements
        Dim pair_incr As Integer
        'There variables store the optimal sill and range
        Dim record_c1 As Double
        Dim record_a As Double
        comp_1 = -1 ' Arbitrary number
        Dim elements As Long
        elements = UBound(emp_semiv) 'Or this can be set to
test_diff_v_counter. Will always be the same.
        'Matrix counter
        Dim mat_count As Long
        mat_count = 1
        SemiForm.prog_label.Caption = "Tweaking Parameters"
        DoEvents
        'Increment sill and range by some step_val and determine the
resulting minimal difference between the fitted and the experimental
calculations
        'The fitted model that results in the lowest minimal difference
between it and the experimental semi-var is the selected model to use
        Dim fit_model As Integer
        For fit_model = elements To Int(elements * 0.75) Step -1
            range_1 = diff_array(fit_model)
            sill = emp_semiv(fit_model)
            For pair_incr = 1 To elements
                'Standard variogram calculations for spherical, exponential,
gamma, and linear models
                If diff_array(pair_incr) <= range_1 Then
                    s_gamma(mat_count) = nugget + sill * ((1.5 *
(diff_array(pair_incr) / range_1)) - (0.5 * (diff_array(pair_incr) / range_1)
^ 3))
                Else
                    s_gamma(mat_count) = sill
                End If
                e_gamma(mat_count) = nugget + sill * (1 - Exp((-3 *
diff_array(pair_incr)) / range_1))
                If diff_array(pair_incr) <= range_1 Then
                    l_gamma(mat_count) = nugget + (diff_array(pair_incr) *
(sill / range_1))
                Else
                    l_gamma(mat_count) = sill
                End If
```

```
                'Distance between calculated model point and exp_semi
                s_diff(mat_count) = Abs(s_gamma(mat_count) -
emp_semiv(pair_incr)) ^ 2
                e_diff(mat_count) = Abs(e_gamma(mat_count) -
emp_semiv(pair_incr)) ^ 2
                l_diff(mat_count) = Abs(l_gamma(mat_count) -
emp_semiv(pair_incr)) ^ 2
                s_diff_sum = s_diff_sum + s_diff(mat_count)
                e_diff_sum = e_diff_sum + e_diff(mat_count)
                l_diff_sum = l_diff_sum + l_diff(mat_count)
            Next
            'Determine minimal distance
            comp = Application.WorksheetFunction.Min(s_diff_sum, e_diff_sum,
l_diff_sum)
            'Whichever fitted model has the lowest minimal distance then add
that to the record_c1 and record_a variables
            If comp_1 > comp Or comp_1 = -1 Then
                If s_diff_sum < e_diff_sum And s_diff_sum < l_diff_sum Then
                    model_name = "Spherical"
                    record_c1 = sill
                    record_a = range_1
                ElseIf e_diff_sum < s_diff_sum And e_diff_sum < l_diff_sum
Then
                    model_name = "Exponential"
                    record_c1 = sill
                    record_a = range_1
                Else
                    model_name = "Linear"
                    record_c1 = sill
                    record_a = range_1
                End If
                'This sets the minimum difference number to compare against
each time through the loop
                comp_1 = comp
            End If
            'Reset variables
            s_diff_sum = 0
            e_diff_sum = 0
            l_diff_sum = 0
            mat_count = mat_count + 1
            ReDim Preserve s_gamma(1 To mat_count)
            ReDim Preserve e_gamma(1 To mat_count)
            ReDim Preserve l_gamma(1 To mat_count)
            ReDim Preserve s_diff(1 To mat_count)
            ReDim Preserve e_diff(1 To mat_count)
            ReDim Preserve l_diff(1 To mat_count)
        Next
        '*************************************
        'Puts the exerpimental semivariogram parameters on the Krig_Dump
sheet
```

```
        det_dim_s = Chr(69) & 1
        Worksheets("Krig_Dump").Range(det_dim_s).Value = "Experimental
Semivariogram (Best Estimate: " & model_name & ") C0, C1, a"
        Worksheets("Krig_Dump").Range(det_dim_s).Offset(1, 0) = nugget
        Worksheets("Krig_Dump").Range(det_dim_s).Offset(2, 0) = record_c1
        Worksheets("Krig_Dump").Range(det_dim_s).Offset(3, 0) = record_a
        '*************************************
        '***Sort the lag and exp semi data**
        Range("A2:D" & test_diff_v_counter + 1).Select
        ActiveWorkbook.Worksheets("Krig_Dump").Sort.SortFields.Clear
        ActiveWorkbook.Worksheets("Krig_Dump").Sort.SortFields.Add
Key:=Range("B2"), _
            SortOn:=xlSortOnValues, Order:=xlAscending,
DataOption:=xlSortNormal
        With ActiveWorkbook.Worksheets("Krig_Dump").Sort
            .SetRange Range("A2:D" & test_diff_v_counter + 1)
            .Header = xlNo
            .MatchCase = False
            .Orientation = xlTopToBottom
            .SortMethod = xlPinYin
            .Apply
        End With
        'Free up memory of unused data
        Erase s_gamma
        Erase e_gamma
        Erase l_gamma
        Erase s_diff
        Erase e_diff
        Erase l_diff
        Erase emp_semiv
    ElseIf SemiForm.exp_var_select.Text = "Standard Model" Then
        'Regress(inpyrng, [inpxrng], [constant], [labels], [confid],
[soutrng], [residuals], [sresiduals], [rplots], [lplots], [routrng],
[nplots], [poutrng])
        Dim r_sqrd As Double
        Dim NewBook_1 As New Worksheet
        Dim past_data_reg As Integer
        Dim past_d_r_1 As Integer
        Dim coef_store(1 To 8) As Double
        Dim coef_store_counter As Integer
        For coef_store_counter = 1 To 8
            coef_store(coef_store_counter) = 0
        Next coef_store_counter
        'Step 4 - Calculate exp semi gamma values using 1/2n(h)*SUM(1 to
n(h))[var(difference in values)]
        'Set the emp_semiv array to the number of elements in the diff_array
because for each individual difference there will be 1 exp-semi point
        ReDim emp_semiv(1 To UBound(diff_array, 1))
        'This variable is the SUM(1 to n(h))[var(difference in values)] part
of the exp-semi equation
```

```
        'Counters
        'Start with 2 instead of 1 because 1 is simply the 0 difference array
and all semi-variogram values would be calculated as 0.
        ii = 2
        j = 0
        Dim valid_var As Boolean
        Dim emp_var_count As Integer
        emp_var_count = 0
        Dim reg_stop_crit As Integer
        Do
            det_dim_s = Chr(68) & (ii + 1)
            'Reset the summation variable for individual difference
            summation_var = 0
            'SUM(1 to n(h))[var(difference in values)]
            For j = 1 To num_diff_array(ii)
                summation_var = summation_var + (out_array(ii, j)) ^ 2
            Next
            'This is the 1/2n(h)* summation_var part of the equation
            emp_semiv(ii) = (1 / (2 * num_diff_array(ii))) * summation_var
            If ii = 2 Then Worksheets("Krig_Dump").Range("D2").Value = 0
            Worksheets("Krig_Dump").Range(det_dim_s).Value = emp_semiv(ii)
            ii = ii + 1
        Loop While ii <= test_diff_v_counter
        '***********************************
        '***Sort the lag and exp semi data**
        Range("A2:D" & test_diff_v_counter + 1).Select
        ActiveWorkbook.Worksheets("Krig_Dump").Sort.SortFields.Clear
        ActiveWorkbook.Worksheets("Krig_Dump").Sort.SortFields.Add
Key:=Range("B2"), _
            SortOn:=xlSortOnValues, Order:=xlAscending,
DataOption:=xlSortNormal
        With ActiveWorkbook.Worksheets("Krig_Dump").Sort
            .SetRange Range("A2:D" & test_diff_v_counter + 1)
            .Header = xlNo
            .MatchCase = False
            .Orientation = xlTopToBottom
            .SortMethod = xlPinYin
            .Apply
        End With
        Dim vario_check As Integer
        For vario_check = 1 To test_diff_v_counter
            If Worksheets("Krig_Dump").Range("D1").Offset(vario_check + 1, 0)
>= Worksheets("Krig_Dump").Range("D1").Offset(vario_check, 0) Then
                emp_var_count = emp_var_count + 1
            End If
        Next vario_check
        If emp_var_count + 1 = test_diff_v_counter Then
            valid_var = True
        Else
            valid_var = False
```

```
        End If
        record_c1 = Worksheets("Krig_Dump").Cells(test_diff_v_counter + 1,
4).Value
        record_a = Worksheets("Krig_Dump").Cells(test_diff_v_counter + 1,
2).Value
        'Puts the exerpimental semivariogram calculations on the Krig_Dump
sheet
        det_dim_s = Chr(69) & 1
        Worksheets("Krig_Dump").Range("D1").Value = "Exp-Var"
        Worksheets("Krig_Dump").Range("E1").Value = "C0, C1, a"
        Worksheets("Krig_Dump").Range(det_dim_s).Offset(1, 0) = nugget
        record_c1 = WorksheetFunction.Max(emp_semiv)
        Worksheets("Krig_Dump").Range(det_dim_s).Offset(2, 0) = record_c1
        Worksheets("Krig_Dump").Range(det_dim_s).Offset(3, 0) = record_a
        '**************************************
        'Find a reasonable model but not one with r^2 of 100%. GRG will not
work properly with a model of r^2 100%
        If valid_var = True Then
            'Set to arbitray number just to enter loop to perform regression
once
            reg_stop_crit = 0
        Else
            reg_stop_crit = cnt_reg + 2
        End If
        Do While test_diff_v_counter <> reg_stop_crit
            SemiForm.prog_label.Caption = "Regression"
            DoEvents
            cnt_reg = cnt_reg + 1
            reg_stop_crit = reg_stop_crit + 1
            'Perform Regression
            If cnt_reg = 1 Then
                Application.Run "ATPVBAEN.XLAM!Regress",
Worksheets("Krig_Dump").Range("D2:D" & test_diff_v_counter + 1) _
                    , Worksheets("Krig_Dump").Range("B2:B" &
test_diff_v_counter + 1), False, False, , "Kriging_Regression", False _
                    , False, False, False, , False
                r_sqrd = Worksheets("Kriging_Regression").Range("B5").Value
                If r_sqrd <> 1 Then
                    curve_fit
                End If
                'If the regression iteration didn't produce a better result
then stop
                If ss_res_comp =
Worksheets("Kriging_Regression").Range("J2").Value Then
                    For coef_store_counter = 1 To cnt_reg
                        Worksheets("Kriging_Regression").Range("B" & 17 +
coef_store_counter).Value = coef_store(coef_store_counter)
                    Next coef_store_counter
                    Exit Do
                Else
```

```
                    For coef_store_counter = 1 To cnt_reg
                        coef_store(coef_store_counter) =
Worksheets("Kriging_Regression").Range("B" & 17 + coef_store_counter).Value
                    Next coef_store_counter
                    ss_res_comp =
Worksheets("Kriging_Regression").Range("J2").Value
                End If
                'If the model is adequate, and there are a minimum of three
regressors, then stop
                If r_sqrd >= 0.8 And cnt_reg >= 3 Then Exit Do
                If valid_var = True Then Exit Do
            ElseIf cnt_reg = 2 Then
                Application.DisplayAlerts = False
                On Error Resume Next
                Worksheets("Kriging_Regression").Delete
                Application.DisplayAlerts = True
                Worksheets("Krig_Dump").Columns("C:C").Select
                Selection.Insert Shift:=xlToRight,
CopyOrigin:=xlFormatFromLeftOrAbove
                Worksheets("Krig_Dump").Range("C1").Value = "Lag11"
                For past_d_r_1 = 1 To test_diff_v_counter
                    Worksheets("Krig_Dump").Range("C" & past_d_r_1 + 1).Value
= (Worksheets("Krig_Dump").Range("B" & past_d_r_1 + 1).Value) ^ 2
                Next past_d_r_1
                Application.Run "ATPVBAEN.XLAM!Regress",
Worksheets("Krig_Dump").Range("E2:E" & test_diff_v_counter + 1) _
                    , Worksheets("Krig_Dump").Range("B2:C" &
test_diff_v_counter + 1), False, False, , "Kriging_Regression", False _
                    , False, False, False, , False
                r_sqrd = Worksheets("Kriging_Regression").Range("B5").Value
                If r_sqrd <> 1 Then
                    curve_fit
                End If
                'If the regression iteration didn't produce a better result
then stop
                If Round(ss_res_comp, 5) <=
Round(Worksheets("Kriging_Regression").Range("J2").Value, 5) Then
                    For coef_store_counter = 1 To cnt_reg
                        Worksheets("Kriging_Regression").Range("B" & 17 +
coef_store_counter).Value = coef_store(coef_store_counter)
                    Next coef_store_counter
                    Exit Do
                Else
                    For coef_store_counter = 1 To cnt_reg
                        coef_store(coef_store_counter) =
Worksheets("Kriging_Regression").Range("B" & 17 + coef_store_counter).Value
                    Next coef_store_counter
                    ss_res_comp =
Worksheets("Kriging_Regression").Range("J2").Value
                End If
```

```
                'If the model is adequate, and there are a minimum of three
regressors, then stop
                If r_sqrd >= 0.8 And cnt_reg >= 3 Then Exit Do
            ElseIf cnt_reg = 3 Then
                Application.DisplayAlerts = False
                On Error Resume Next
                Worksheets("Kriging_Regression").Delete
                Application.DisplayAlerts = True
                Worksheets("Krig_Dump").Columns("D:D").Select
                Selection.Insert Shift:=xlToRight,
CopyOrigin:=xlFormatFromLeftOrAbove
                Worksheets("Krig_Dump").Range("D1").Value = "Lag111"
                For past_d_r_1 = 1 To test_diff_v_counter
                    Worksheets("Krig_Dump").Range("D" & past_d_r_1 + 1).Value
= (Worksheets("Krig_Dump").Range("B" & past_d_r_1 + 1).Value) ^ 3
                Next past_d_r_1
                Application.Run "ATPVBAEN.XLAM!Regress",
Worksheets("Krig_Dump").Range("F2:F" & test_diff_v_counter + 1) _
                    , Worksheets("Krig_Dump").Range("B2:D" &
test_diff_v_counter + 1), False, False, , "Kriging_Regression", False _
                    , False, False, False, , False
                r_sqrd = Worksheets("Kriging_Regression").Range("B5").Value
                If r_sqrd <> 1 Then
                    curve_fit
                End If
                'If the regression iteration didn't produce a better result
then stop
                 If Round(ss_res_comp, 5) <=
Round(Worksheets("Kriging_Regression").Range("J2").Value, 5) Then
                    For coef_store_counter = 1 To cnt_reg
                        Worksheets("Kriging_Regression").Range("B" & 17 +
coef_store_counter).Value = coef_store(coef_store_counter)
                    Next coef_store_counter
                    Exit Do
                Else
                    For coef_store_counter = 1 To cnt_reg
                        coef_store(coef_store_counter) =
Worksheets("Kriging_Regression").Range("B" & 17 + coef_store_counter).Value
                    Next coef_store_counter
                    ss_res_comp =
Worksheets("Kriging_Regression").Range("J2").Value
                End If
                'If the model is adequate, and there are a minimum of three
regressors, then stop
                If r_sqrd >= 0.8 And cnt_reg >= 3 Then Exit Do
            ElseIf cnt_reg = 4 Then
                Application.DisplayAlerts = False
                On Error Resume Next
                Worksheets("Kriging_Regression").Delete
                Application.DisplayAlerts = True
```

```
                Worksheets("Krig_Dump").Columns("E:E").Select
                Selection.Insert Shift:=xlToRight,
CopyOrigin:=xlFormatFromLeftOrAbove
                Worksheets("Krig_Dump").Range("E1").Value = "Lag1111"
                For past_d_r_1 = 1 To test_diff_v_counter
                    Worksheets("Krig_Dump").Range("E" & past_d_r_1 + 1).Value
= (Worksheets("Krig_Dump").Range("B" & past_d_r_1 + 1).Value) ^ 4
                Next past_d_r_1
                Application.Run "ATPVBAEN.XLAM!Regress",
Worksheets("Krig_Dump").Range("G2:G" & test_diff_v_counter + 1) _
                    , Worksheets("Krig_Dump").Range("B2:E" &
test_diff_v_counter + 1), False, False, , "Kriging_Regression", False _
                    , False, False, False, , False
                r_sqrd = Worksheets("Kriging_Regression").Range("B5").Value
                If r_sqrd <> 1 Then
                    curve_fit
                End If
                'If the regression iteration didn't produce a better result
then stop
                 If Round(ss_res_comp, 5) <=
Round(Worksheets("Kriging_Regression").Range("J2").Value, 5) Then
                    For coef_store_counter = 1 To cnt_reg
                        Worksheets("Kriging_Regression").Range("B" & 17 +
coef_store_counter).Value = coef_store(coef_store_counter)
                    Next coef_store_counter
                    Exit Do
                Else
                    For coef_store_counter = 1 To cnt_reg
                        coef_store(coef_store_counter) =
Worksheets("Kriging_Regression").Range("B" & 17 + coef_store_counter).Value
                    Next coef_store_counter
                    ss_res_comp =
Worksheets("Kriging_Regression").Range("J2").Value
                End If
                'If the model is adequate, and there are a minimum of three
regressors, then stop
                If r_sqrd >= 0.8 And cnt_reg >= 3 Then Exit Do
            ElseIf cnt_reg = 5 Then
                Application.DisplayAlerts = False
                On Error Resume Next
                Worksheets("Kriging_Regression").Delete
                Application.DisplayAlerts = True
                Worksheets("Krig_Dump").Columns("F:F").Select
                Selection.Insert Shift:=xlToRight,
CopyOrigin:=xlFormatFromLeftOrAbove
                Worksheets("Krig_Dump").Range("F1").Value = "Lag11111"
                For past_d_r_1 = 1 To test_diff_v_counter
                    Worksheets("Krig_Dump").Range("F" & past_d_r_1 + 1).Value
= (Worksheets("Krig_Dump").Range("B" & past_d_r_1 + 1).Value) ^ 5
                Next past_d_r_1
```

```
                Application.Run "ATPVBAEN.XLAM!Regress",
Worksheets("Krig_Dump").Range("H2:H" & test_diff_v_counter + 1) _
                    , Worksheets("Krig_Dump").Range("B2:F" &
test_diff_v_counter + 1), False, False, , "Kriging_Regression", False _
                    , False, False, False, , False
                r_sqrd = Worksheets("Kriging_Regression").Range("B5").Value
                If r_sqrd <> 1 Then
                    curve_fit
                End If
                'If the regression iteration didn't produce a better result
then stop
                If Round(ss_res_comp, 5) <=
Round(Worksheets("Kriging_Regression").Range("J2").Value, 5) Then
                    For coef_store_counter = 1 To cnt_reg
                        Worksheets("Kriging_Regression").Range("B" & 17 +
coef_store_counter).Value = coef_store(coef_store_counter)
                    Next coef_store_counter
                    Exit Do
                Else
                    For coef_store_counter = 1 To cnt_reg
                        coef_store(coef_store_counter) =
Worksheets("Kriging_Regression").Range("B" & 17 + coef_store_counter).Value
                    Next coef_store_counter
                    ss_res_comp =
Worksheets("Kriging_Regression").Range("J2").Value
                End If
                'If the model is adequate, and there are a minimum of three
regressors, then stop
                If r_sqrd >= 0.8 And cnt_reg >= 3 Then Exit Do
            ElseIf cnt_reg = 6 Then
                Application.DisplayAlerts = False
                On Error Resume Next
                Worksheets("Kriging_Regression").Delete
                Application.DisplayAlerts = True
                Worksheets("Krig_Dump").Columns("G:G").Select
                Selection.Insert Shift:=xlToRight,
CopyOrigin:=xlFormatFromLeftOrAbove
                Worksheets("Krig_Dump").Range("G1").Value = "Lag111111"
                For past_d_r_1 = 1 To test_diff_v_counter
                    Worksheets("Krig_Dump").Range("G" & past_d_r_1 + 1).Value
= (Worksheets("Krig_Dump").Range("B" & past_d_r_1 + 1).Value) ^ 6
                Next past_d_r_1
                Application.Run "ATPVBAEN.XLAM!Regress",
Worksheets("Krig_Dump").Range("I2:I" & test_diff_v_counter + 1) _
                    , Worksheets("Krig_Dump").Range("B2:G" &
test_diff_v_counter + 1), False, False, , "Kriging_Regression", False _
                    , False, False, False, , False
                r_sqrd = Worksheets("Kriging_Regression").Range("B5").Value
                If r_sqrd <> 1 Then
                    curve_fit
```

```
                End If
                'If the regression iteration didn't produce a better result
then stop
                If Round(ss_res_comp, 5) <=
Round(Worksheets("Kriging_Regression").Range("J2").Value, 5) Then
                    For coef_store_counter = 1 To cnt_reg
                        Worksheets("Kriging_Regression").Range("B" & 17 +
coef_store_counter).Value = coef_store(coef_store_counter)
                    Next coef_store_counter
                    Exit Do
                Else
                    For coef_store_counter = 1 To cnt_reg
                        coef_store(coef_store_counter) =
Worksheets("Kriging_Regression").Range("B" & 17 + coef_store_counter).Value
                    Next coef_store_counter
                    ss_res_comp =
Worksheets("Kriging_Regression").Range("J2").Value
                End If
                'If the model is adequate, and there are a minimum of three
regressors, then stop
                If r_sqrd >= 0.8 And cnt_reg >= 3 Then Exit Do
            ElseIf cnt_reg = 7 Then
                Application.DisplayAlerts = False
                On Error Resume Next
                Worksheets("Kriging_Regression").Delete
                Application.DisplayAlerts = True
                Worksheets("Krig_Dump").Columns("H:H").Select
                Selection.Insert Shift:=xlToRight,
CopyOrigin:=xlFormatFromLeftOrAbove
                Worksheets("Krig_Dump").Range("H1").Value = "Lag1111111"
                For past_d_r_1 = 1 To test_diff_v_counter
                    Worksheets("Krig_Dump").Range("H" & past_d_r_1 + 1).Value
= (Worksheets("Krig_Dump").Range("B" & past_d_r_1 + 1).Value) ^ 7
                Next past_d_r_1
                Application.Run "ATPVBAEN.XLAM!Regress",
Worksheets("Krig_Dump").Range("J2:J" & test_diff_v_counter + 1) _
                    , Worksheets("Krig_Dump").Range("B2:H" &
test_diff_v_counter + 1), False, False, , "Kriging_Regression", False _
                    , False, False, False, , False
                r_sqrd = Worksheets("Kriging_Regression").Range("B5").Value
                If r_sqrd <> 1 Then
                    curve_fit
                End If
                'If the regression iteration didn't produce a better result
then stop
                If Round(ss_res_comp, 5) <=
Round(Worksheets("Kriging_Regression").Range("J2").Value, 5) Then
                    For coef_store_counter = 1 To cnt_reg
                        Worksheets("Kriging_Regression").Range("B" & 17 +
coef_store_counter).Value = coef_store(coef_store_counter)
```

```
                    Next coef_store_counter
                    Exit Do
                Else
                    For coef_store_counter = 1 To cnt_reg
                        coef_store(coef_store_counter) =
Worksheets("Kriging_Regression").Range("B" & 17 + coef_store_counter).Value
                    Next coef_store_counter
                    ss_res_comp =
Worksheets("Kriging_Regression").Range("J2").Value
                End If
                'If the model is adequate, and there are a minimum of three
regressors, then stop
                If r_sqrd >= 0.8 And cnt_reg >= 3 Then Exit Do
            ElseIf cnt_reg = 8 Then
                Application.DisplayAlerts = False
                On Error Resume Next
                Worksheets("Kriging_Regression").Delete
                Application.DisplayAlerts = True
                Worksheets("Krig_Dump").Columns("I:I").Select
                Selection.Insert Shift:=xlToRight,
CopyOrigin:=xlFormatFromLeftOrAbove
                Worksheets("Krig_Dump").Range("I1").Value = "Lag1111111"
                For past_d_r_1 = 1 To test_diff_v_counter
                    Worksheets("Krig_Dump").Range("I" & past_d_r_1 + 1).Value
= (Worksheets("Krig_Dump").Range("B" & past_d_r_1 + 1).Value) ^ 8
                Next past_d_r_1
                Application.Run "ATPVBAEN.XLAM!Regress",
Worksheets("Krig_Dump").Range("K2:K" & test_diff_v_counter + 1) _
                    , Worksheets("Krig_Dump").Range("B2:I" &
test_diff_v_counter + 1), False, False, , "Kriging_Regression", False _
                    , False, False, False, , False
                r_sqrd = Worksheets("Kriging_Regression").Range("B5").Value
                If r_sqrd <> 1 Then
                    curve_fit
                End If
                'If the regression iteration didn't produce a better result
then stop
                If Round(ss_res_comp, 5) <=
Round(Worksheets("Kriging_Regression").Range("J2").Value, 5) Then
                    For coef_store_counter = 1 To cnt_reg
                        Worksheets("Kriging_Regression").Range("B" & 17 +
coef_store_counter).Value = coef_store(coef_store_counter)
                    Next coef_store_counter
                    Exit Do
                Else
                    For coef_store_counter = 1 To cnt_reg
                        coef_store(coef_store_counter) =
Worksheets("Kriging_Regression").Range("B" & 17 + coef_store_counter).Value
                    Next coef_store_counter
```

```
                    ss_res_comp =
Worksheets("Kriging_Regression").Range("J2").Value
                End If
                'If the model is adequate, and there are a minimum of three
regressors, then stop
                 If r_sqrd >= 0.8 And cnt_reg >= 3 Then Exit Do
            End If
            If cnt_reg > 8 Then
                err_msg_1 = MsgBox("The regression model could not find an
acceptable solution. Please choose another variogram option and run the
program again.", vbOKOnly, "Error Handler")
                Exit Sub
            End If
        Loop
        '************Add Variogram
Chart*****************************************
        Dim chart_title As String
        Dim chart_title_count As Integer
        For chart_title_count = 1 To cnt_reg
            If chart_title_count = cnt_reg Then
                chart_title = chart_title & "(" &
Round(Worksheets("Kriging_Regression").Range("B" & 17 +
chart_title_count).Value, 4) & ")*Lag^" & chart_title_count
            Else
                chart_title = chart_title & "(" &
Round(Worksheets("Kriging_Regression").Range("B" & 17 +
chart_title_count).Value, 4) & ")*Lag^" & chart_title_count & " + "
            End If
        Next chart_title_count
        Application.Sheets("Krig_Dump").Activate
        Range("A" & test_diff_v_counter + 3).Select
        ActiveSheet.Shapes.AddChart.Select
        ActiveChart.ChartType = xlXYScatterSmoothNoMarkers
        ActiveChart.HasTitle = True
        ActiveChart.ChartTitle.Text = chart_title
        ActiveChart.SeriesCollection.NewSeries
        ActiveChart.SeriesCollection(1).Name = "=""Emperical Variogram"""
        ActiveChart.SeriesCollection(1).XValues = "=Krig_Dump!$B$2:$B$" &
test_diff_v_counter + 1
        ActiveChart.SeriesCollection(1).Values = "=Krig_Dump!" & Chr(67 +
cnt_reg) & "$2:$" & Chr(67 + cnt_reg) & "$" & test_diff_v_counter + 1
        ActiveChart.SeriesCollection.NewSeries
        ActiveChart.SeriesCollection(2).Name = "=""Standard Model"""
        ActiveChart.SeriesCollection(2).XValues = "=Krig_Dump!$B$2:$B$" &
test_diff_v_counter + 1
        ActiveChart.SeriesCollection(2).Values =
"=Kriging_Regression!$B$28:$B$" & 27 + test_diff_v_counter

'*************************************************************************
    Else
```

```
        SemiForm.prog_label.Caption = "User Defined Variogram"
        DoEvents
        'SemiForm.Repaint
        'Error Handling
        If SemiForm.nug.Value = "" Or CDbl(SemiForm.nug.Value) < 0 Then
            err_msg_1 = MsgBox("The nugget value can not be null or below
zero if you have selected a specific experimental semivariogram. The program
will exit now.", vbOKOnly, "Error Handler")
            Exit Sub
        End If
        If SemiForm.sil.Value = "" Or CDbl(SemiForm.sil.Value) < 0 Then
            err_msg_1 = MsgBox("The sill value can not be null or below zero
if you have selected a specific experimental semivariogram. The program will
exit now.", vbOKOnly, "Error Handler")
            Exit Sub
        End If
        If SemiForm.ran.Value = "" Or CDbl(SemiForm.ran.Value) < 0 Then
            err_msg_1 = MsgBox("The range value can not be null or below zero
if you have selected a specific experimental semivariogram. The program will
exit now.", vbOKOnly, "Error Handler")
            Exit Sub
        End If
        'End Error Handling
        'This data is gathered from the user input if the preferred
calculation methods are not used.
        model_name = SemiForm.exp_var_select.Text
        nugget = CDbl(SemiForm.nug.Value)
        record_c1 = CDbl(SemiForm.sil.Value)
        record_a = CDbl(SemiForm.ran.Value)
        '*************************************
        'det_dim_s = Chr(68) & 1
        Worksheets("Krig_Dump").Range(Chr(69) & 1).Value = "Selected
Semivariogram (" & model_name & ") Values (c0, c1, a)"
        Worksheets("Krig_Dump").Range(Chr(69) & 1).Offset(1, 1) = nugget
        Worksheets("Krig_Dump").Range(Chr(69) & 1).Offset(2, 1) = record_c1
        Worksheets("Krig_Dump").Range(Chr(69) & 1).Offset(3, 1) = record_a
        '*************************************
    End If
    'Dim for 1 additional element to take into account the lagrangian
multiplier
    Dim Final_Gamma() As Double
    ReDim Final_Gamma(1 To (max_inp_out_array_size + 1), 1 To
(max_inp_out_array_size + 1))
    'Calculate final gamma values
    Dim inp_inc_1 As Integer
    Dim inp_inc_2 As Integer
    Dim matrix_start_position As Integer
    Dim matrix_value As Double
    Dim fin_gamma_reg As Double
    matrix_start_position = 2
```

```
    Dim msg As String
    For inp_inc_2 = 1 To max_inp_out_array_size
        'Lagrangian multiplier
        Final_Gamma(inp_inc_2, (max_inp_out_array_size + 1)) = 1
        Final_Gamma((max_inp_out_array_size + 1), inp_inc_2) = 1
        Final_Gamma(inp_inc_2, inp_inc_2) = 0
        For inp_inc_1 = matrix_start_position To (max_inp_out_array_size)
            msg = ""
            matrix_value = euc_values(inp_inc_2, inp_inc_1, det_dim)
            If model_name = "Spherical" Then
                'Spherical calculation
                If matrix_value <= record_a Then
                    Final_Gamma(inp_inc_2, inp_inc_1) = nugget + record_c1 *
((1.5 * (matrix_value / record_a)) - (0.5 * (matrix_value) ^ 3))
                Else
                    Final_Gamma(inp_inc_2, inp_inc_1) = record_c1
                End If
            ElseIf model_name = "Exponential" Then
                'Exponential calculation
                Final_Gamma(inp_inc_2, inp_inc_1) = nugget + record_c1 * (1 -
Exp((-3 * matrix_value) / record_a))
            ElseIf model_name = "Gaussian" Then
                'Gaussian Calculation
                If matrix_value <= record_a Then
                    Final_Gamma(inp_inc_2, inp_inc_1) = nugget + record_c1 *
(1 - Exp((-3 * (matrix_value ^ 2)) / (record_a ^ 2)))
                Else
                    Final_Gamma(inp_inc_2, inp_inc_1) = record_c1
                End If
            ElseIf model_name = "Linear" Then
                'Linear Calculation
                If matrix_value <= record_a Then
                    Final_Gamma(inp_inc_2, inp_inc_1) = nugget +
(matrix_value * (record_c1 / record_a))
                Else
                    Final_Gamma(inp_inc_2, inp_inc_1) = record_c1
                End If
            Else
                'This option is if the standard model is used
                'The model depends on the order of the regression model
                For fin_gamma_reg = 1 To cnt_reg
                    Final_Gamma(inp_inc_2, inp_inc_1) =
Final_Gamma(inp_inc_2, inp_inc_1) + ((matrix_value ^ fin_gamma_reg) *
Worksheets("Kriging_Regression").Range("B" & 17 + fin_gamma_reg).Value)
                Next fin_gamma_reg
            End If
            Final_Gamma(inp_inc_1, inp_inc_2) = Final_Gamma(inp_inc_2,
inp_inc_1)
        Next
        matrix_start_position = matrix_start_position + 1
```

```
    Next
    'Lagrangian multiplier
    Final_Gamma((max_inp_out_array_size + 1), (max_inp_out_array_size + 1)) =
0
    Dim f_msg() As String 'Use this to populate a text box in the results GUI
    ReDim f_msg(1 To max_inp_out_array_size)
    'ReDim f_msg(1 To UBound(diff_array, 1))
    'This variable is to aide in displaying results
    '***********************************
    det_dim_s = Chr(70 + cnt_reg) & 1
    Worksheets("Krig_Dump").Range(det_dim_s).Value = "Final LHS Gamma
Calculations"
    Dim ofset_cal As Integer
    '***********************************
    Dim f_msg_inp As String
    For inp_inc_2 = 1 To max_inp_out_array_size
        msg = ""
        For inp_inc_1 = 1 To max_inp_out_array_size
            '***********************************
            ofset_cal = ofset_cal + 1
            '***********************************
            'Populate output to user
            If inp_inc_1 < max_inp_out_array_size Then
                msg = msg & Round(Final_Gamma(inp_inc_2, inp_inc_1), 3) & ",
"
            Else
                msg = msg & Round(Final_Gamma(inp_inc_2, inp_inc_1), 3)
            End If
            '**************************************
            If inp_inc_1 = max_inp_out_array_size Then
                Worksheets("Krig_Dump").Range(det_dim_s).Offset(ofset_cal, 0)
= Final_Gamma(inp_inc_2, inp_inc_1)
                ofset_cal = ofset_cal + 1
                Worksheets("Krig_Dump").Range(det_dim_s).Offset(ofset_cal, 0)
= ""
            Else
                Worksheets("Krig_Dump").Range(det_dim_s).Offset(ofset_cal, 0)
= Final_Gamma(inp_inc_2, inp_inc_1)
            End If
            '**************************************
        Next
        'Fill the final gamma GUI text box
        f_msg_inp = ""
        For dim_count = 1 To det_dim
            'If (det_dim = 1 And inp_inc_2 <> max_inp_out_array_size) Or
(det_dim <> 1 And inp_inc_2 = max_inp_out_array_size) Then
            If (det_dim = 1) Or (det_dim <> 1 And dim_count = det_dim) Then
                f_msg_inp = f_msg_inp &
Range(SemiForm.RefEditOriginalSampleRange.Value).Cells(inp_inc_2,
dim_count).Value
```

```
            'ElseIf det_dim <> 1 And inp_inc_2 <> max_inp_out_array_size Then
            ElseIf (det_dim <> 1) And (dim_count <> det_dim) Then
                f_msg_inp = f_msg_inp &
Range(SemiForm.RefEditOriginalSampleRange.Value).Cells(inp_inc_2,
dim_count).Value & ","
            End If
        Next
        f_msg(inp_inc_2) = "Input " & f_msg_inp & ":: " & msg & Chr(13) &
Chr(13)
        'Populate Gamma Coeff results box
        Results_Form.Results_1.Text = Results_Form.Results_1.Text &
f_msg(inp_inc_2)
    Next
    'Populate fitted model details
    If model_name <> "Standard Model" Then
        Results_Form.Results_2.Text = "Your fitted model is " & model_name &
"." & Chr(13) & Chr(13) & "Co is = " & Round(nugget, 3) & ", c1 is = " &
Round(record_c1, 3) & ", and a is = " & Round(record_a, 3) & "." _
            & Chr(13) & Chr(13) & "The resulting minimal difference in
samples versues observations was " & Round(comp_1, 3) & "."
    Else
        Results_Form.Results_2.Text = "Your fitted model is " & model_name &
"." & Chr(13) & "The resulting minimal difference in samples versues
observations was " & Worksheets("Kriging_Regression").Range("J2").Value & "."
    End If

'**************************************************************************
    'Step 7 and 8 - Construct the system of equations
    'Perform Matrix Transformation
    'This variable is a flag to indicate whether to perform multiple kriging
or a single point prediction. More less a dummy flag.
    Dim dimension_count As Integer
    Dim ArrInv() As Variant
    Dim ArrAns() As Variant
    Dim multi_dim_exit As Integer
    Dim msg1 As String
    If SemiForm.Mult_Pred.Value = "" Then
        Dim krig_inp_point As Double
    Else
        Dim krig_inp_point_2() As Double
        ReDim krig_inp_point_2(1 To det_dim)
    End If
    Dim krig_inp_point_1 As Double
    Dim point_est() As Double
    Dim matrix_inv As Integer
    Dim matrix_inv_1 As Integer
    Dim matrix_count As Integer
    Dim Z As Double
    Dim krig_count As Integer
    Dim krig_var As Double
```

```
    Dim krig_std As Double
    Dim krig_conf As Double
    Dim krig_conf_l As Double
    Dim krig_conf_h As Double
    Dim arr_count As Integer
    Dim msg_2 As String
    Dim pred_value_lp As Integer
    Dim krig_result_point As Variant
    Dim MD_array_counter As Integer
    Dim col As Integer
    Dim reset_dim As Integer
    SemiForm.prog_label.Caption = "Weight Calculations"
    DoEvents
    'RHS gamma values
    MD_array_counter = det_dim
    dimension_count = 1
    resolution = 0
    'determine the minimum and maximum number in each dimension to construct
the MD stopping points
    If SemiForm.Mult_Pred.Value <> "" And det_dim > 1 Then
        Dim max_dim_var() As Integer
        Dim min_dim_var() As Integer
        ReDim max_dim_var(1 To det_dim)
        ReDim min_dim_var(1 To det_dim)
        For col = 1 To det_dim
            max_dim_var(col) =
Application.WorksheetFunction.Max(Range(SemiForm.RefEditOriginalSampleRange.V
alue).Columns(col))
            min_dim_var(col) =
Application.WorksheetFunction.Min(Range(SemiForm.RefEditOriginalSampleRange.V
alue).Columns(col))
            If col <> det_dim Then
                krig_inp_point_2(col) =
Range(SemiForm.Predict_Input.Value).Cells(1, col).Value
            Else
                krig_inp_point_2(col) =
Range(SemiForm.Predict_Input.Value).Cells(1, col).Value -
Range(SemiForm.Mult_Pred.Value)
            End If
        Next
    End If
    Dim disp_count As Integer
    'This loops based until all the values from the original point plus the
resolution value are covered
    Dim get_weight As Interior
    Dim max_min_chk As Integer
    Dim MD_exit_chk As Integer
    Dim MD_array_counter_1 As Integer
    Dim MD_array_counter_2 As Integer
    MD_array_counter_2 = 1
```

```
    Dim dimen_offset As Integer
    dimen_offset = 1
    get_weight = 0
    Do
        get_weight = get_weight + 1
        'Need to calculate the RHS gamma values
        'This is the input variable selected by the user
        'SemiForm.Mult_Pred.Value = "" means single point kriging prediction
else then some resoltion was given.
        If SemiForm.Mult_Pred.Value <> "" Then
            'det_dim > 1 means there are more than 1 dimension to the problem
            If det_dim = 1 Then
                krig_inp_point = Range(SemiForm.Predict_Input.Value) +
resolution
                '***Exit loop once the prediction point exceeds the maximum
input value i.e. to prevent extrapolation***
                If krig_inp_point >
Range(SemiForm.RefEditOriginalSampleRange.Value).Cells(Range(SemiForm.RefEdit
OriginalSampleRange.Value).Rows.Count, 1) Then Exit Do
            Else
                'Need to increment each bound until the max, then reset it to
the min value and start incrementing the next dimension
                'Create a counter as we go from one dimension to the next.
Once the counter exceeds the last dimension then exit the loop
                'If krig_inp_point_2(MD_array_counter) +
Range(SemiForm.Mult_Pred.Value) > max_dim_var(MD_array_counter) And
dimension_count = 1 Then
                MD_array_counter_1 = 0
                For MD_exit_chk = 1 To det_dim
                    If krig_inp_point_2(MD_exit_chk) >=
Range(SemiForm.Predict_Input.Value).Cells(Range(SemiForm.RefEditOriginalSampl
eRange.Value).Rows.Count, MD_exit_chk).Value Then
                        MD_array_counter_1 = MD_array_counter_1 + 1
                    End If
                    If MD_array_counter_1 = det_dim Then
                        Exit Do
                    End If
                Next MD_exit_chk
                'ElseIf krig_inp_point_2(MD_array_counter) +
Range(SemiForm.Mult_Pred.Value) > max_dim_var(MD_array_counter) And
dimension_count >= 1 Then
                If krig_inp_point_2(MD_array_counter) +
Range(SemiForm.Mult_Pred.Value) > max_dim_var(MD_array_counter) And
dimension_count >= 1 Then
                    'The prior statement captures when you hit a maximum
point
                    krig_inp_point_2(MD_array_counter) =
min_dim_var(MD_array_counter)
                    For max_min_chk = (det_dim - 1) To 1 Step -1
```

```
                              krig_inp_point_2(max_min_chk) =
krig_inp_point_2(max_min_chk) + Range(SemiForm.Mult_Pred.Value)
                              If krig_inp_point_2(max_min_chk) >
max_dim_var(max_min_chk) Then
                                  krig_inp_point_2(max_min_chk) =
min_dim_var(max_min_chk)
                              Else
                                  Exit For
                              End If
                          Next max_min_chk
                      Else
                           krig_inp_point_2(MD_array_counter) =
krig_inp_point_2(MD_array_counter) + Range(SemiForm.Mult_Pred.Value)
                      End If
                  End If
              End If
              ReDim point_est(1 To (max_inp_out_array_size + 1))
              '**************************************
              det_dim_s = Chr(71 + cnt_reg) & 1
              Worksheets("Krig_Dump").Range(det_dim_s).Value = "Final Right Hand
Side Gamma Calculations"
              '**************************************
              'Loop through each input value and calculate the distance between the
input point and the kriging point along with the RHS gamma calculation
              For pair_incr = 1 To max_inp_out_array_size
                  'If det_dim = 1 Then
                  If SemiForm.Mult_Pred.Value = "" Then
                      krig_inp_point_1 = euc_values_rhs(pair_incr, 1, det_dim)
                  ElseIf SemiForm.Mult_Pred.Value <> "" And det_dim = 1 Then
                      krig_inp_point_1 = euc_values_rhs(pair_incr, 1, det_dim)
                  Else
                      krig_inp_point_1 = euc_values_rhs_MD_w_RES(pair_incr, 1,
det_dim, krig_inp_point_2)
                  End If
                  If model_name = "Spherical" Then
                      'RHS gamma calculations
                      If krig_inp_point_1 = 0 Then
                          point_est(pair_incr) = 0
                      Else
                          If Abs(krig_inp_point_1) <= record_a Then
                              point_est(pair_incr) = nugget + record_c1 * ((1.5 *
(Abs(krig_inp_point_1) / record_a)) - (0.5 * (Abs(krig_inp_point_1) /
record_a) ^ 3))
                          Else
                              point_est(pair_incr) = record_c1
                          End If
                      End If
                  ElseIf model_name = "Exponential" Then
                      'RHS gamma calculations
                      If krig_inp_point_1 = 0 Then
```

```
                point_est(pair_incr) = 0
            Else
                point_est(pair_incr) = nugget + record_c1 * (1 - Exp((-3
* Abs(krig_inp_point_1)) / record_a))
            End If
        ElseIf model_name = "Gaussian" Then
            'RHS gamma calculations
            If krig_inp_point_1 = 0 Then
                point_est(pair_incr) = 0
            Else
                If Abs(krig_inp_point_1) <= record_a Then
                    point_est(pair_incr) = nugget + record_c1 * (1 -
Exp(-3 * (krig_inp_point_1 ^ 2) / (record_a ^ 2)))
                Else
                    point_est(pair_incr) = record_c1
                End If
            End If
        ElseIf model_name = "Linear" Then
            'RHS gamma calculations
            If krig_inp_point_1 = 0 Then
                point_est(pair_incr) = 0
            Else
                If Abs(krig_inp_point_1) <= record_a Then
                    point_est(pair_incr) = nugget + Abs(krig_inp_point_1)
* (record_c1 / record_a)
                Else
                    point_est(pair_incr) = record_c1
                End If
            End If
        Else
            'RHS gamma calculations
            If krig_inp_point_1 = 0 Then
                point_est(pair_incr) = 0
            Else
                For fin_gamma_reg = 1 To cnt_reg
                    point_est(pair_incr) = point_est(pair_incr) +
(Abs(krig_inp_point_1 ^ fin_gamma_reg) *
Worksheets("Kriging_Regression").Range("B" & 17 + fin_gamma_reg).Value)
                Next fin_gamma_reg
            End If
        End If
        'Put RHS Gamma Values in Spreadsheet
        '**************************************
        If SemiForm.Mult_Pred.Value = "" Then
            Worksheets("Krig_Dump").Range(det_dim_s).Offset(pair_incr, 0)
= point_est(pair_incr)
        Else
            Worksheets("Krig_Dump").Range(det_dim_s).Offset(1, 0) = "N/A"
        End If
        '**************************************
```

```
            If pair_incr <> max_inp_out_array_size Then
                msg1 = msg1 & Round(point_est(pair_incr), 3) & Chr(13)
            Else
                msg1 = msg1 & Round(point_est(pair_incr), 3) & Chr(13) &
Chr(13)
            End If
        Next
        'RHS Lagrangian multiplier
        point_est(max_inp_out_array_size + 1) = 1
        'Populate RHS gamma calculation results
        Results_Form.Results_4.Text = msg1
        '*************************************
        Dim det_dim_s_weights As String
        det_dim_s_weights = Chr(72 + cnt_reg) & 1
        Worksheets("Krig_Dump").Range(det_dim_s_weights).Value = "Final
Weights"
        'Calculate weights
        'Only need to get the weights once
        If get_weight = 1 Then
            ArrInv() = Application.WorksheetFunction.MInverse(Final_Gamma)
            If ArrInv(1) = "" Then
                ArrInv =
Excel.Application.WorksheetFunction.MInverse(Final_Gamma)
            End If
        End If
        ArrAns = Excel.Application.WorksheetFunction.MMult(ArrInv,
Excel.Application.WorksheetFunction.Transpose(point_est))
        If ArrAns(1) = "" Then
            ArrAns = Excel.Application.WorksheetFunction.MMult(ArrInv,
Excel.Application.WorksheetFunction.Transpose(point_est))
        End If
        '*************************************
        'Put Gamma Inversion Values in Krig_Dump Spreadsheet
        If SemiForm.Mult_Pred.Value = "" Then
            Worksheets("Krig_Dump").Range(det_dim_s_weights).Offset(1,
0).Resize(UBound(ArrAns), 1) = ArrAns
        Else
            Worksheets("Krig_Dump").Range(det_dim_s_weights).Offset(1, 0) =
"N/A"
            Worksheets("Krig_Dump").Range(det_dim_s).Offset(1, 0) = "N/A"
        End If
        '**************************************
        arr_count = 1
        'Populate the weights to be displayed to the user
        Do
            If arr_count <> UBound(ArrAns, 1) Then
                msg_2 = msg_2 & "Lamda (" & arr_count & "): " &
Round(ArrAns(arr_count, 1), 3) & Chr(13)
            Else
```

```
                msg_2 = msg_2 & "Mu (1): " & Round(ArrAns(arr_count, 1), 3) &
Chr(13) & Chr(13)
            End If
            arr_count = arr_count + 1
        Loop While arr_count <= UBound(ArrAns, 1)
        Results_Form.Weight.Text = msg_2
        'http://microsoft.allfaq.org/forums/t/155165.aspx

'*************************************************************************
        'Step 9 is to perform Ordinay Kriging to estimate point
        'Reset the prediction value to 0 each time.
        'This variable is called Z but it actually represents the t
distribution, not the z-score of the standard normal
        Z = 0
        krig_var = 0
        krig_std = 0
        'Perform the kriging algorithm. Z* = SUM(weights*original output
values)
        For krig_count = 1 To max_inp_out_array_size
            Z = Z + (ArrAns(krig_count, 1) * inp_out_data(krig_count, 2))
        Next
        For krig_count = 1 To (max_inp_out_array_size + 1)
            krig_var = krig_var + ArrAns(krig_count, 1) *
point_est(krig_count)
        Next
        If krig_var < 0 Then krig_var = 0
        krig_std = Sqr(Abs(krig_var))
        krig_conf = 0
        krig_conf = Application.WorksheetFunction.TInv((1 -
CDbl(SemiForm.CI.Value)), max_inp_out_array_size - 1) * krig_std /
(Sqr(max_inp_out_array_size))
        krig_conf_l = Z - krig_conf
        krig_conf_h = Z + krig_conf
        '*************************************
        disp_count = disp_count + 1
        det_dim_s = Chr(73 + cnt_reg) & disp_count
        If disp_count = 1 Then
            Worksheets("Krig_Dump").Range(det_dim_s).Value = "Kriging
Prediction"
        End If
        Worksheets("Krig_Dump").Range(det_dim_s).Offset(dimension_count, 0) =
Z
        det_dim_s = Chr(74 + cnt_reg) & disp_count
        If disp_count = 1 Then
            Worksheets("Krig_Dump").Range(det_dim_s).Value = "Prediction
Error"
        End If
        Worksheets("Krig_Dump").Range(det_dim_s).Offset(dimension_count, 0) =
krig_var
        det_dim_s = Chr(75 + cnt_reg) & disp_count
```

```
        If disp_count = 1 Then
            Worksheets("Krig_Dump").Range(det_dim_s).Value = "Error Standard
Deviation"
        End If
        Worksheets("Krig_Dump").Range(det_dim_s).Offset(dimension_count, 0) =
krig_std
        det_dim_s = Chr(76 + cnt_reg) & disp_count
        If disp_count = 1 Then
            Worksheets("Krig_Dump").Range(det_dim_s).Value = "FLC"
        End If
        Dim FLC_wks_outpt As String
        Dim FLC_dim_count As Integer
        If SemiForm.Mult_Pred.Value = "" Then
            FLC_wks_outpt = ""
            If det_dim = 1 Then
                FLC_wks_outpt = Range(SemiForm.Predict_Input.Value)
            Else
                For FLC_dim_count = 1 To det_dim
                    If FLC_dim_count <> det_dim Then
                        FLC_wks_outpt = FLC_wks_outpt &
Range(SemiForm.Predict_Input.Value).Cells(1, FLC_dim_count).Value & ","
                    Else
                        FLC_wks_outpt = FLC_wks_outpt &
Range(SemiForm.Predict_Input.Value).Cells(1, FLC_dim_count).Value
                    End If
                Next
            End If
        Else
            FLC_wks_outpt = ""
            If det_dim = 1 Then
                FLC_wks_outpt = krig_inp_point
            Else
                For FLC_dim_count = 1 To det_dim
                    If FLC_dim_count <> det_dim Then
                        FLC_wks_outpt = FLC_wks_outpt &
krig_inp_point_2(FLC_dim_count) & ","
                    Else
                        FLC_wks_outpt = FLC_wks_outpt &
krig_inp_point_2(FLC_dim_count)
                    End If
                Next
            End If
        End If
        Worksheets("Krig_Dump").Range(det_dim_s).Offset(dimension_count, 0) =
FLC_wks_outpt
        '***************************************

'***************************************************************************
        'Populate Kriging results box.
```

```
        'SemiForm.Mult_Pred.Value = "" means single point kriging prediction
else then some resoltion was given.
        pred_value_lp = 0
        If SemiForm.Mult_Pred.Value = "" Then
            If det_dim = 1 Then
                Results_Form.Results_3.Text = "The predictive value at point
" & Round(Range(SemiForm.Predict_Input.Value).Cells(1, 1).Value, 3) & " is: "
& Round(Z, 3) & " with a Variance of " & Round(krig_var, 3) & " and a Std Dev
of " & Round(krig_std, 3) & ". The 95% HCIL (+/- " & Round(krig_conf, 3) & ")
is " & Round(krig_conf_l, 3) & "|" & Round(krig_conf_h, 3) & "."
            Else
                For pred_value_lp = 1 To det_dim
                    If pred_value_lp = 1 Then
                        krig_result_point = krig_result_point &
Range(SemiForm.Predict_Input.Value).Cells(1, pred_value_lp).Value
                    Else
                        krig_result_point = krig_result_point & ", " &
Range(SemiForm.Predict_Input.Value).Cells(1, pred_value_lp).Value
                    End If
                Next
                Results_Form.Results_3.Text = "The predictive value at point
" & krig_result_point & " is: " & Round(Z, 3) & " with a Variance of " &
Round(krig_var, 3) & " and a Std Dev of " & Round(krig_std, 3) & ". The 95%
HCIL (+/- " & Round(krig_conf, 3) & ") is " & Round(krig_conf_l, 3) & "|" &
Round(krig_conf_h, 3) & "."
            End If
        Else
            If det_dim = 1 Then
                Results_Form.Results_3.Text = Results_Form.Results_3.Text &
"The predictive value at point " & Round(krig_inp_point, 3) & " is: " &
Round(Z, 3) & " Variance: " & Round(krig_var, 3) & " Std Dev: " &
Round(krig_std, 3) & ". The 95% HCIL (+/- " & Round(krig_conf, 3) & ") is " &
Round(krig_conf_l, 3) & "|" & Round(krig_conf_h, 3) & "." & Chr(13)
            Else
                krig_result_point = ""
                For pred_value_lp = 1 To det_dim
                    If pred_value_lp = 1 Then
                         krig_result_point = krig_result_point &
krig_inp_point_2(pred_value_lp)
                    Else
                        krig_result_point = krig_result_point & ", " &
krig_inp_point_2(pred_value_lp)
                    End If
                Next
                Results_Form.Results_3.Text = Results_Form.Results_3.Text &
"The predictive value at point " & krig_result_point & " is: " & Round(Z, 3)
& " with a Variance of " & Round(krig_var, 3) & " and a Std Dev of " &
Round(krig_std, 3) & ". The 95% CI (z +/- " & Round(krig_conf, 3) & ") is " &
Round(krig_conf_l, 3) & "|" & Round(krig_conf_h, 3) & "." & Chr(13)
            End If
```

```
        End If
        If SemiForm.Mult_Pred.Value <> "" Then
            resolution = resolution + Range(SemiForm.Mult_Pred.Value)
        End If
    Loop While SemiForm.Mult_Pred.Value <> ""
    '**********************************
    'Autofit the Krig_Dump sheet
    Worksheets("Krig_Dump").Columns("A:Z").EntireColumn.AutoFit
    '**********************************
    Sheets("Krig_Dump").Activate
    If SemiForm.Mult_Pred.Value <> "" Then
        '************Add Response
Curve*********************************************
        Dim last_used_cell As Variant
        last_used_cell = Range(Chr(73 + cnt_reg) & "65536").End(xlUp).Row + 1
        Range("A" & test_diff_v_counter + 3).Select
        ActiveSheet.Shapes.AddChart.Select
        ActiveChart.ChartType = xlXYScatterSmoothNoMarkers
        ActiveChart.SeriesCollection.NewSeries
        ActiveChart.SeriesCollection(1).Name = "=""Response Curve"""
        If det_dim = 1 Then
            ActiveChart.SeriesCollection(1).XValues = "=Krig_Dump!$" & Chr(76
+ cnt_reg) & "$2:$" & Chr(76 + cnt_reg) & "$" & last_used_cell - 1
        End If
        ActiveChart.SeriesCollection(1).Values = "=Krig_Dump!$" & Chr(73 +
cnt_reg) & "$2:$" & Chr(73 + cnt_reg) & "$" & last_used_cell - 1

'**************************************************************************
    End If
    Range("A1").Select
    'Determine end time
    sngEnd = Timer                                  ' Get end time.
    'Determine time elapsed
    sngElapsed = Format(sngEnd - sngStart, "Fixed") ' Elapsed time.
    Results_Form.E_Time_Txt.Text = sngElapsed
    Unload SemiForm
    Results_Form.Show
End Sub
Function Last(choice As Long, rng As Range)
'Ron de Bruin, 5 May 2008
' 1 = last row
' 2 = last column
' 3 = last cell
    Dim lrw As Long
    Dim lcol As Long
    Select Case choice
    Case 1:
        On Error Resume Next
        Last = rng.Find(What:="*", _
                        After:=rng.Cells(1), _
```

```
                                Lookat:=xlPart, _
                                LookIn:=xlFormulas, _
                                SearchOrder:=xlByRows, _
                                SearchDirection:=xlPrevious, _
                                MatchCase:=False).Row
            On Error GoTo 0
        Case 2:
            On Error Resume Next
            Last = rng.Find(What:="*", _
                                After:=rng.Cells(1), _
                                Lookat:=xlPart, _
                                LookIn:=xlFormulas, _
                                SearchOrder:=xlByColumns, _
                                SearchDirection:=xlPrevious, _
                                MatchCase:=False).Column
            On Error GoTo 0
        Case 3:
            On Error Resume Next
            lrw = rng.Find(What:="*", _
                               After:=rng.Cells(1), _
                               Lookat:=xlPart, _
                               LookIn:=xlFormulas, _
                               SearchOrder:=xlByRows, _
                               SearchDirection:=xlPrevious, _
                               MatchCase:=False).Row
            On Error GoTo 0
            On Error Resume Next
            lcol = rng.Find(What:="*", _
                                After:=rng.Cells(1), _
                                Lookat:=xlPart, _
                                LookIn:=xlFormulas, _
                                SearchOrder:=xlByColumns, _
                                SearchDirection:=xlPrevious, _
                                MatchCase:=False).Column
            On Error GoTo 0
            On Error Resume Next
            Last = rng.Parent.Cells(lrw, lcol).Address(False, False)
            If Err.Number > 0 Then
                Last = rng.Cells(1).Address(False, False)
                Err.Clear
            End If
            On Error GoTo 0
    End Select
End Function
Function euc_values(f_a_count As Integer, f_b_count As Integer, f_det_dim As
Integer)
    'Reset before calculating each time
    Dim f_euc_cal_0 As Double
    Dim f_dim_count_0 As Integer
    f_euc_cal_0 = 0
```

```
    For f_dim_count_0 = 1 To f_det_dim
        f_euc_cal_0 = f_euc_cal_0 +
(Range(SemiForm.RefEditOriginalSampleRange.Value).Cells(f_a_count,
f_dim_count_0).Value -
Range(SemiForm.RefEditOriginalSampleRange.Value).Cells(f_b_count,
f_dim_count_0).Value) ^ 2
    Next
    euc_values = Sqr(f_euc_cal_0)
End Function
Function euc_values_rhs(f_a_count As Integer, f_b_count As Integer, f_det_dim
As Integer)
    'Reset before calculating each time
    Dim f_euc_cal_1 As Double
    Dim f_dim_count_1 As Integer
    f_euc_cal_1 = 0
    For f_dim_count_1 = 1 To f_det_dim
        f_euc_cal_1 = f_euc_cal_1 +
(Range(SemiForm.RefEditOriginalSampleRange.Value).Cells(f_a_count,
f_dim_count_1).Value - (Range(SemiForm.Predict_Input.Value).Cells(f_b_count,
f_dim_count_1).Value + resolution)) ^ 2
    Next
    euc_values_rhs = Sqr(f_euc_cal_1)
End Function
Function euc_values_rhs_MD_w_RES(f_a_count As Integer, f_b_count As Integer,
f_det_dim As Integer, arr() As Double)
    'Reset before calculating each time
    Dim f_euc_cal_2 As Double
    Dim f_dim_count_2 As Integer
    f_euc_cal_2 = 0
    For f_dim_count_2 = 1 To f_det_dim
        f_euc_cal_2 = f_euc_cal_2 +
(Range(SemiForm.RefEditOriginalSampleRange.Value).Cells(f_a_count,
f_dim_count_2).Value - arr(f_dim_count_2)) ^ 2
    Next
    euc_values_rhs_MD_w_RES = Sqr(f_euc_cal_2)
End Function
Function CheckSolver() As Boolean
    'Adjusted for Application.Run() to avoid Reference problems with Solver
    ' Peltier Technical Services, Inc., Copyright  2007. All rights reserved.
    ' Returns True if Solver can be used, False if not.
    Dim bSolverInstalled As Boolean
    Dim bSolverInstalled_1 As Boolean
    Dim bSolverInstalled_2 As Boolean
    ' Assume true unless otherwise
    CheckSolver = True
    On Error Resume Next
    ' check whether Solver is installed
    bSolverInstalled = Application.AddIns("Solver Add-In").Installed
    bSolverInstalled_1 = Application.AddIns("Analysis ToolPak").Installed
```

```
    bSolverInstalled_2 = Application.AddIns("Analysis ToolPak -
VBA").Installed
    Err.Clear
    If Not bSolverInstalled Then
        ' (re)install Solver
        Application.AddIns("Solver Add-In").Installed = True
        bSolverInstalled = Application.AddIns("Solver Add-In").Installed
    End If
    If Not bSolverInstalled_1 Then
        Application.AddIns("Analysis ToolPak").Installed = True
        bSolverInstalled_1 = Application.AddIns("Analysis ToolPak").Installed
    End If
    If Not bSolverInstalled_2 Then
        Application.AddIns("Analysis ToolPak - VBA").Installed = True
        bSolverInstalled_2 = Application.AddIns("Analysis ToolPak -
VBA").Installed
    End If
    If Not bSolverInstalled Or Not bSolverInstalled_1 Or Not
bSolverInstalled_2 Then
        MsgBox "The required data analysis tools are not found. Please load
the Analysis ToolPak, Analysis ToolPak - VBA, and Solver Add-In.", vbCritical
        CheckSolver = False
    End If
    On Error GoTo 0
End Function
Function curve_fit()
    SemiForm.prog_label.Caption = "GRG"
    DoEvents
    ' reset
    Dim yhat_reg As Integer
    Dim yhat As String
    Dim res_reg As String
    Dim sum_sq_res As Integer
    Dim past_d_r_2 As Integer
    Dim err_msg_2
    'After Regression we need to calculate yhat, eis, and SSres
    'yhat
    Range("B28").Select
    For sum_sq_res = 1 To (test_diff_v_counter)
        For yhat_reg = 1 To cnt_reg + 1
            If yhat_reg = 1 Then
                yhat = yhat + "=R[" & -10 - sum_sq_res & "]C"
            Else
                yhat = yhat + "+R[" & -11 - sum_sq_res + yhat_reg &
"]C*Krig_Dump!R[-26]C[" & -2 + yhat_reg & "]"
            End If
        Next yhat_reg
        Worksheets("Kriging_Regression").Range("B" & 27 +
sum_sq_res).FormulaR1C1 = yhat
        yhat = ""
```

```
    Next sum_sq_res
    Range("C28").Select
    'eis
    For sum_sq_res = 1 To (test_diff_v_counter)
        res_reg = "=Krig_Dump!R[-26]C[" & cnt_reg & "]-RC[-1]"
        Worksheets("Kriging_Regression").Range("C" & 27 +
sum_sq_res).FormulaR1C1 = res_reg
        res_reg = ""
    Next sum_sq_res
    'SSres
    For sum_sq_res = 1 To (test_diff_v_counter)
        Worksheets("Kriging_Regression").Range("D" & 27 + sum_sq_res).Formula
= "=C" & 27 + sum_sq_res & "^ 2"
    Next sum_sq_res
    Worksheets("Kriging_Regression").Range("L1").Value = "Max Gamma Value"
    Worksheets("Kriging_Regression").Range("L2").Value =
Worksheets("Krig_Dump").Cells(test_diff_v_counter + 1, cnt_reg + 3).Value
    Worksheets("Kriging_Regression").Range("J1").Value = "SS(Res)"
    Worksheets("Kriging_Regression").Range("J2").Formula = "=sum(D" & 28 &
":D" & 27 + test_diff_v_counter & ")"
    Application.Run "Solver.xlam!SolverReset"
    Dim sol_by_chang As String
    Dim last_const As String
    sol_by_chang = "$B$17:$B$" & 17 + (cnt_reg)
    Application.Run "Solver.xlam!SolverOk", "$J$2", 2, 0, sol_by_chang, 1,
"GRG Nonlinear"
    Application.Run "Solver.xlam!SolverAdd", "$B$17", 2, "0"
    For past_d_r_2 = 1 To (test_diff_v_counter - 1)
        Application.Run "Solver.xlam!SolverAdd", "$B$" & 27 + past_d_r_2, 1,
"$B$" & 28 + past_d_r_2
    Next past_d_r_2
    ' run the analysis
    Dim result As Integer
    result = Application.Run("Solver.xlam!SolverSolve", True)
    ' finish the analysis
    Application.Run "Solver.xlam!SolverFinish"
    ' report on success of analysis
    If result >= 4 Or (Round(ss_res_comp, 5) <
Round(Worksheets("Kriging_Regression").Range("J2").Value, 5) And ss_res_comp
<> 0) Then
        SemiForm.prog_label.Caption = "Switching to EA"
        DoEvents
        'Resolve using EA engine
        Application.Run "Solver.xlam!SolverOk", "$J$2", 2, 0, sol_by_chang,
1, "Evolutionary"
        Dim result_1 As Integer
        ' run the analysis
        result_1 = Application.Run("Solver.xlam!SolverSolve", True)
        ' finish the analysis
        Application.Run "Solver.xlam!SolverFinish"
```

```
    End If
End Function
```

Appendix A.2: Test Planning Module

```
Public sum_var As Double
'*************************************
'Simple load and unload form section
Private Sub CommandButton3_Click()
    ' Put away the form
    Unload pd_analysis
    Exit Sub
End Sub
Private Sub CommandButton4_Click()
    ' Put away the form
    Unload Me
    'Load form
    ShowSemiForm
End Sub
'*************************************
'*****************START HELP SECTION OF THE FORM***************************
Private Sub Image11_Click()
    Krig_Help.TextBox1.Text = "Enter the integer starting (min) value for the
inputs. This can not be a negative number."
    Krig_Help.Show
End Sub
Private Sub Image12_Click()
    Krig_Help.TextBox1.Text = "Enter the integer ending (max) value for the
inputs. This can not be a negative number and must be greter than the minimum
number."
    Krig_Help.Show
End Sub
Private Sub Image13_Click()
    Krig_Help.TextBox1.Text = "Enter the total test budget allocated to
collecting sample data."
    Krig_Help.Show
End Sub
Private Sub Image14_Click()
    Krig_Help.TextBox1.Text = "If a total test budget is entered, the user
may the cost test test."
    Krig_Help.Show
End Sub
Private Sub Image15_Click()
    Krig_Help.TextBox1.Text = "This box allows the user to linearly scale the
cost per test up or down by a factor between -9.9 to 9.9."
    Krig_Help.Show
End Sub
Private Sub Image16_Click()
```

```
    Krig_Help.TextBox1.Text = "This list box shows the recommended FLCs that
are associated with each sample size."
    Krig_Help.Show
End Sub
Private Sub Image18_Click()
    Krig_Help.TextBox1.Text = "This allows the user to select how much of the
data should be posted into the worksheet. The default of 0 is to paste the
recommnded pilot design into the worksheet. If altered, this value must be an
integer greater than 1 and less than or equal to the available problem space
sample size. Changing this value from the default results in an increase in
processing time."
    Krig_Help.Show
End Sub
Private Sub Image19_Click()
    Krig_Help.TextBox1.Text = "This allows the user to enter a nugget value.
The default is to 0 and must not be set to negative."
    Krig_Help.Show
End Sub
Private Sub Image4_Click()
    Krig_Help.TextBox1.Text = "The user can specify the increment size. The
default covers all unique lags in the problem space."
    Krig_Help.Show
End Sub
Private Sub Image5_Click()
    Krig_Help.TextBox1.Text = "Enter a numerical integer that is greater than
zero. Note: each input/dimension must be equal in size. The max is 8
dimensions"
    Krig_Help.Show
End Sub
Private Sub Image9_Click()
    Krig_Help.TextBox1.Text = "The list box below displays critical
information depending on sample size. This information is used to allow the
user to select a starting sample size."
    Krig_Help.Show
End Sub
'******************END HELP SECTION OF THE FORM****************************
Private Sub exp_var_confirm_Click()
    'Determine start time
    Dim sngStart As Single, sngEnd As Single
    Dim sngElapsed As Single
    sngStart = Timer ' Get start time.
    Me.CommandButton1.Visible = False
    '******************START FORM ERROR
CHECKING******************************
    If CDbl(pd_analysis.num_dim.Value) < 1 Or pd_analysis.num_dim.Value = ""
Or CDbl(pd_analysis.num_dim.Value) - Int(CDbl(pd_analysis.num_dim.Value)) <>
0 Or CDbl(pd_analysis.num_dim.Value) > 8 Then
        err_msg_1 = MsgBox("Required. The number of dimensions must be an
integer greater than 0 but less than 8.", vbOKOnly, "Error Handler")
        Exit Sub
```

```
    End If
    If CDbl(pd_analysis.min_val.Value) < 0 Or pd_analysis.min_val.Value = ""
Or CDbl(pd_analysis.min_val.Value) - Int(CDbl(pd_analysis.min_val.Value)) <>
0 Then
        err_msg_1 = MsgBox("Required. The minimum starting value must be an
integer greater than or equal to 0.", vbOKOnly, "Error Handler")
        Exit Sub
    End If
    'Currently disabled with plans to enable listed as part of future
research. The only lag selection is the default
    'which is the minimum increment of 1 for this research.
    'If CDbl(pd_analysis.ComboBoxlag.Value) <= 0 Or
pd_analysis.ComboBoxlag.Value <> "(Default)" Then
    '    err_msg_1 = MsgBox("Required. The minimum lag distance must be
greater than 0. The program will exit now.", vbOKOnly, "Error Handler")
    '    Exit Sub
    'End If
    If CDbl(pd_analysis.max_val.Value) <= CDbl(pd_analysis.min_val.Value) Or
pd_analysis.max_val.Value = "" Or CDbl(pd_analysis.max_val.Value) -
Int(CDbl(pd_analysis.max_val.Value)) <> 0 Then
        err_msg_1 = MsgBox("Required. The maximum starting value must be an
integer greater than the minimum value.", vbOKOnly, "Error Handler")
        Exit Sub
    End If
    If CDbl(pd_analysis.nug.Value) < 0 Or pd_analysis.nug.Value = "" Then
        err_msg_1 = MsgBox("Required. The nugget value must be greater than
0.", vbOKOnly, "Error Handler")
        Exit Sub
    End If
    If CDbl(pd_analysis.budget.Value) <= 0 Or pd_analysis.budget.Value = ""
Then
        err_msg_1 = MsgBox("Required. The budget must be greater than 0.",
vbOKOnly, "Error Handler")
        Exit Sub
    End If
    If CDbl(pd_analysis.cost_p_test.Value) > CDbl(pd_analysis.budget.Value)
Or CDbl(pd_analysis.cost_p_test.Value) <= 0 Or pd_analysis.cost_p_test.Value
= "" Then
        err_msg_1 = MsgBox("Required. The cost per test must be greater than
0 and less than the total budget and must not be blank.", vbOKOnly, "Error
Handler")
        Exit Sub
    End If
    If CDbl(pd_analysis.lin_cost.Value) > 9.9 Or
CDbl(pd_analysis.lin_cost.Value) < -9.9 Then
        err_msg_1 = MsgBox("Optional unless a budget and cost per test is
specified. The value must range between -9.9 to 9.9. This box must remain
blank if no budget is specified.", vbOKOnly, "Error Handler")
        Exit Sub
    End If
```

```
    If (CDbl(pd_analysis.max_val) ^ CDbl(num_dim)) > 65000 Then
        err_msg_1 = MsgBox("The number of FLCs exceeds the amount allowed by
Excel. Please adjust (lower) the Max Value and/or the Number of Dimensions.",
vbOKOnly, "Error Handler")
        Exit Sub
    End If
    '******************END FORM ERROR CHECKING****************************
    '******************Create new sheet to display data*****************
        'Create a new sheet for data dumps and calculation results
        Dim NewBook As New Worksheet
        Application.DisplayAlerts = False
        On Error Resume Next
        Sheets("Test_Planning").Delete
        Application.DisplayAlerts = True
        Set NewBook = Worksheets.Add
        NewBook.Name = "Test_Planning"
    '*********************************************************************

    '**************Display title and user input on new worksheet***********
    Worksheets("Test_Planning").Range("A1").Value = "Test Planning Worksheet"
    With Worksheets("Test_Planning").Range("A1").Font
        .Name = "Calibri"
        .Size = 24
        .Strikethrough = False
        .Superscript = False
        .Subscript = False
        .OutlineFont = False
        .Shadow = False
        .Underline = xlUnderlineStyleNone
        .ThemeColor = xlThemeColorLight1
        .TintAndShade = 0
        .ThemeFont = xlThemeFontMinor
    End With
    For hghl = 2 To 3
        For hghl_1 = 2 To 6 Step 2
            With Worksheets("Test_Planning").Cells(hghl, hghl_1).Interior
                .Pattern = xlSolid
                .PatternColorIndex = xlAutomatic
                .ThemeColor = xlThemeColorLight2
                .TintAndShade = 0.799981688894314
                .PatternTintAndShade = 0
            End With
        Next hghl_1
    Next hghl
    With Worksheets("Test_Planning").Range("A1").Font.Bold = True
        Worksheets("Test_Planning").Range("A2").Value = "# Dim ="
        Worksheets("Test_Planning").Range("B2").Value = CDbl(num_dim)
        Worksheets("Test_Planning").Range("C2").Value = "Min Value ="
        Worksheets("Test_Planning").Range("D2").Value = CDbl(min_val)
        Worksheets("Test_Planning").Range("E2").Value = "Max Value ="
```

```
        Worksheets("Test_Planning").Range("F2").Value = CDbl(max_val)
        Worksheets("Test_Planning").Range("A3").Value = "Budget ="
        Worksheets("Test_Planning").Range("B3").Value =
CDbl(pd_analysis.budget)
        Worksheets("Test_Planning").Range("C3").Value = "Cost/Test ="
        Worksheets("Test_Planning").Range("D3").Value =
CDbl(pd_analysis.cost_p_test)
        Worksheets("Test_Planning").Range("E3").Value = "Cost Scale ="
        Worksheets("Test_Planning").Range("F3").Value =
CDbl(pd_analysis.lin_cost)
        Worksheets("Test_Planning").Range("A4").Value = "FLCs"
    End With
    Columns("C:C").EntireColumn.AutoFit
    Columns("E:E").EntireColumn.AutoFit
    With Worksheets("Test_Planning").Range("A4").Font
        .Name = "Calibri"
        .Size = 18
        .Strikethrough = False
        .Superscript = False
        .Subscript = False
        .OutlineFont = False
        .Shadow = False
        .Underline = xlUnderlineStyleNone
        .ThemeColor = xlThemeColorLight1
        .TintAndShade = 0
        .ThemeFont = xlThemeFontMinor
    End With
    '***********************************************************************

    '*****Convert Input into equally spaced array with dimension Xsupd*****
    'Step 1 - Create Problem Space Array
    'This routine creates a 2D array. The 1st dimension is the unique
identifier and the second dimension houses
    'the n-dimensional values of the problem space.
    Dim problem_space_array() As Integer
    ReDim problem_space_array(1 To CDbl(pd_analysis.max_val) ^ CDbl(num_dim),
1 To CDbl(num_dim))
    Dim x_count As Integer
    Dim lag_inc As Integer
    Dim lag_inc_1 As Integer
    Dim x_count1 As Integer
    For x_count = 1 To CDbl(num_dim)
        lag_inc_1 = 0
        If x_count = 1 Then
            lag_inc_count = 1
            lag_inc = 0
            dim_x_count = x_count
        Else
            lag_inc = 1
            lag_inc_count = 1
```

```
            dim_x_count = x_count - 1
        End If
        For x1_count = 1 To CDbl(pd_analysis.max_val) ^ CDbl(num_dim)
            lag_inc_1 = lag_inc_1 + 1
            If CDbl(pd_analysis.max_val) ^ dim_x_count >= lag_inc_count Then
                If x_count = 1 Then
                    lag_inc = lag_inc + 1
                End If
            Else
                If x_count = 1 Then
                    lag_inc = 1
                    lag_inc_count = 1
                Else
                    If lag_inc = CDbl(pd_analysis.max_val) Then
                        lag_inc = 1
                    Else
                        lag_inc = lag_inc + 1
                    End If
                    lag_inc_count = 1
                End If
            End If
            lag_inc_count = lag_inc_count + 1
            problem_space_array(lag_inc_1, (CDbl(num_dim) - x_count + 1)) =
CDbl(pd_analysis.min_val) + (lag_inc - 1)
            'The following two with statements provide highlighting for the
FLCs
            With Worksheets("Test_Planning").Cells(x1_count + 4,
CDbl(num_dim) - (x_count - 1)).Interior
                .Pattern = xlSolid
                .PatternColorIndex = xlAutomatic
                .ThemeColor = xlThemeColorLight1
                .TintAndShade = 0.799981688894314
                .PatternTintAndShade = 0
            End With
        Next x1_count
    Next x_count
    [a5].Resize(UBound(problem_space_array), CDbl(num_dim)) =
problem_space_array
    '******************End Input Arrray
Creation****************************************

    '******************Step 2 - Augmented simulated annealing
process*****************
    'This process consists of multiple steps that are defined by comment as
required
    'This appraoch, by nature, is required to be integrated together with
most of the remaining code of this section.
    'This is the reason why step 2 is so large in terms of lines of code.
    '***The first and second action is to take the min and max value of Xsupd
and create a design and COV matrix***
```

```
    'Distance Matrix Creation
    Dim dist_matrix() As Double
    '2^num_dim will capture all the end points of the factor space
    ReDim dist_matrix(1 To 2 ^ num_dim, 1 To 2 ^ num_dim)
    'Need to determine all FLCs to be added to pilot design
    'Assign the FLCs for the pilot design and determine the Euclidean
distance
    'Assigns the diagonal of zeros
    Dim init_dsgn As Integer
    Dim init_dsgn_1 As Integer
    For init_dsgn = 1 To 2 ^ num_dim
        dist_matrix(init_dsgn, init_dsgn) = 0
    Next init_dsgn
    'Reset before calculating each time
    Dim f_euc_cal_0 As Double
    Dim f_dim_count_0 As Integer
    'Need to determine the lags for each value in the permutation
    Dim per_check As Integer
    per_check = 0
    Dim perm_matrix() As Variant
    ReDim perm_matrix(1 To 2 ^ num_dim, 1 To num_dim)
    Dim per_count As Integer
    per_count = 0
    Dim lb_1 As MSForms.ListBox
    Set lb_1 = Me.sam_location
    lb_1.Clear ' clear the listbox content
    lb_1.ColumnCount = CDbl(num_dim) + 1
    For init_dsgn = 1 To CDbl(pd_analysis.max_val) ^ CDbl(num_dim)
        'Only get distance if the current FLC passes the permutation test
        For f_dim_count_0 = 1 To CDbl(num_dim)
            'Need to determine which FLCs in the worksheet have a mix of all
high or low values
            If Worksheets("Test_Planning").Cells(init_dsgn + 4,
f_dim_count_0) = CDbl(pd_analysis.max_val) Or
Worksheets("Test_Planning").Cells(init_dsgn + 4, f_dim_count_0) =
CDbl(pd_analysis.min_val) Then
                per_check = per_check + 1
            End If
        Next f_dim_count_0
        If per_check = CDbl(num_dim) Then
            per_count = per_count + 1
            For f_dim_count_0 = 1 To CDbl(num_dim)
                perm_matrix(per_count, f_dim_count_0) =
Worksheets("Test_Planning").Cells(init_dsgn + 4, f_dim_count_0)
                'Store the FLC ID Number
            Next f_dim_count_0
            lb_1.AddItem init_dsgn
        End If
        per_check = 0
    Next init_dsgn
```

```
    'After permutation matrix is developed, now we need the lag matrix
    For init_dsgn = 2 To 2 ^ num_dim 'rows
        For init_dsgn_1 = 1 To (init_dsgn - 1) 'columns
            f_euc_cal_0 = 0
            For f_dim_count_0 = 1 To CDbl(num_dim)
                f_euc_cal_0 = f_euc_cal_0 + (perm_matrix(init_dsgn,
f_dim_count_0) - perm_matrix(init_dsgn_1, f_dim_count_0)) ^ 2
            Next f_dim_count_0
            'Due to Symmetry
            dist_matrix(init_dsgn_1, init_dsgn) = Sqr(f_euc_cal_0)
            dist_matrix(init_dsgn, init_dsgn_1) = dist_matrix(init_dsgn_1,
init_dsgn)
        Next init_dsgn_1
    Next init_dsgn
    'Create the initial COV matrix
    'The 1st dimension will be redimmed as the pilot design grows and is the
ith index of the COV matrix
    'The 2nd dimension will be redimmed as the pilot design grows and is the
jth index of the COV matrix
    Dim cov_matrix() As Double
    ReDim cov_matrix(1 To (2 ^ num_dim) + 1, 1 To (2 ^ num_dim) + 1)
    Dim nugget As Double
    Dim c1 As Double
    Dim a As Double
    'Set the range equal to the maximum lag
    a = dist_matrix(2 ^ num_dim, 1)
    'Set c1 = 1. c1 or the sil is the variance portion of the graph.
    'It is reasonable to set the sill equal to the sample variance since the
data is evenly distributed.
    'Under the assumption of normality, the data can be transformed into Z
(standard normal) thus requiring
    'the population variance = 1
    c1 = 1
    nugget = CDbl(pd_analysis.nug)
    For init_dsgn = 1 To (2 ^ num_dim) + 1
        cov_matrix(init_dsgn, init_dsgn) = 0
    Next init_dsgn
    For init_dsgn = 2 To 2 ^ num_dim 'row
        For init_dsgn_1 = 1 To (init_dsgn - 1) 'column
            cov_matrix(init_dsgn, init_dsgn_1) = nugget + c1 * (1 - Exp((-3 *
dist_matrix(init_dsgn, init_dsgn_1)) / a))
            'Due to symmetry
            cov_matrix(init_dsgn_1, init_dsgn) = cov_matrix(init_dsgn,
init_dsgn_1)
        Next init_dsgn_1
    Next init_dsgn
    For init_dsgn = 1 To (2 ^ num_dim) 'row
        '****Lagrangian Multiplier*****
        cov_matrix(init_dsgn, ((2 ^ num_dim)) + 1) = 1
        cov_matrix(((2 ^ num_dim)) + 1, init_dsgn) = 1
```

```
        '******************************
    Next init_dsgn
    '*****END building and defining the initial matricies***

    '*****Get cost data************************************
    Dim tot_budget As Double
    Dim cpt As Double
    Dim lin_scale As Double
    tot_budget = CDbl(pd_analysis.budget)
    cpt = CDbl(pd_analysis.cost_p_test)
    lin_scale = CDbl(pd_analysis.lin_cost)
    '*****End Get cost data************************************

    '***********Start search algorithm*************
    'set expected improvement, which will be used as the stopping criterion
    Dim ei As Double
    Dim ei_weight As Double
    'Set EI to a starting value of 1
    Dim flag As Integer
    'This flag determines the search direction. 0 searches left and 1
searches right
    Dim dir_flag As Integer
    flag = 0
    Dim index_count As Integer
    'set the starting search point to the indexed value just below the max
    index_count = CDbl(pd_analysis.max_val) - 1
    'Identify candidate input
    'Need to store the input index and distance
    Dim RHS_lag_marix() As Integer
    ReDim RHS_lag_matrix(1 To 1, 1 To CDbl(num_dim))
    Dim RHS_dis_matrix() As Double
    ReDim RHS_dis_matrix(1 To 2 ^ num_dim, 1 To 1)
    Dim lag_pointer As Integer
    Dim cand_input() As Double
    ReDim cand_input(1 To (2 ^ num_dim) + 1)
    lag_pointer =
Application.WorksheetFunction.RoundUp(UBound(problem_space_array) / 2, 0)
    'After center point is selected, calculate the RHS lag and COV matrix
    For f_dim_count_0 = 1 To CDbl(num_dim)
        RHS_lag_matrix(1, f_dim_count_0) = problem_space_array(lag_pointer,
f_dim_count_0)
    Next f_dim_count_0
    For init_dsgn = 1 To (2 ^ num_dim) 'row
        f_euc_cal_0 = 0
        For f_dim_count_0 = 1 To CDbl(num_dim)
            f_euc_cal_0 = f_euc_cal_0 + (perm_matrix(init_dsgn,
f_dim_count_0) - RHS_lag_matrix(1, f_dim_count_0)) ^ 2
        Next
        RHS_dis_matrix(init_dsgn, 1) = Sqr(f_euc_cal_0)
```

```
        cand_input(init_dsgn) = nugget + c1 * (1 - Exp((-3 *
RHS_dis_matrix(init_dsgn, 1)) / a))
    Next init_dsgn
    'Lagrange multiplier
    cand_input(init_dsgn) = 1
    'Perform search algorithm until either an acceptable expected improvement
is reached or no better solutions are found
    'Need to first calculate the initial variance based on the 2 point
initial FLC. This is the starting point i.e. worst variance possible.
    'This requires one additional point. (max+min)/2 will be used to
calculate the worst variance.
    Dim ArrInv() As Variant
    Dim ArrAns() As Variant
    'Calculate weights
    ArrInv() = Application.MInverse(cov_matrix)
    ArrAns() = Application.MMult(ArrInv, Application.Transpose(cand_input))
    'Calculate initial variance
    Dim krig_var As Double
    Dim var_best As Double
    krig_var = 0
    For krig_count = 1 To CDbl(2 ^ num_dim) + 1
        krig_var = krig_var + ArrAns(krig_count, 1) * cand_input(krig_count)
    Next
    'The search direction is 1d based on distances (left -0 and right - 1)
    dir_flag = 1
    'Set the initial variance equal to the best candidate and therefore add
the third FLC to the design
    'var_best = krig_var
    var_best = 0
    Dim krig_var_old As Double
    krig_var_old = krig_var
    'Fill the list boxes with the initial data
    Dim lb As MSForms.ListBox
    Set lb = Me.sam_selection
    Dim ini_samp_size As Integer
    Dim ini_samp_size_1 As Integer
    ini_samp_size = 0
    'Fill the sample selection list box with the initial two samples
    Dim cost_tot_new As Integer
    'ei = 100
    With lb
        .Clear ' clear the listbox content
        .ColumnCount = 4
        cost_tot = CDbl(pd_analysis.cost_p_test)
        bud_remain = CDbl(pd_analysis.budget)
        For ini_samp_size = 1 To (2 ^ num_dim) + 1
            .AddItem ini_samp_size
            If ini_samp_size <> CDbl(2 ^ num_dim) + 1 Then
                .List(ini_samp_size - 1, 1) = "-"
            Else
```

```
                .List(ini_samp_size - 1, 1) = krig_var
            End If
            If CDbl(pd_analysis.lin_cost) = 0 Then
                bud_remain = bud_remain - cost_tot
            Else
                If ini_samp_size = 1 Then
                    bud_remain = bud_remain - cost_tot
                    'This variable increases each time by a linear amount.
This variable tracks that.
                    cost_tot_new = (CDbl(pd_analysis.lin_cost) * cost_tot)
                Else
                    bud_remain = bud_remain - (CDbl(pd_analysis.lin_cost) *
cost_tot_new)
                    cost_tot_new = (CDbl(pd_analysis.lin_cost) *
cost_tot_new)
                End If
            End If
           .List(ini_samp_size - 1, 2) = Round(bud_remain, 2)
        Next ini_samp_size
        sum_var = sum_var + krig_var
        .ListIndex = 0 'select the first item
    End With
    'Fill the sample location list box with the initial samples
    With lb_1
        Dim col_wid_num As String
        col_wid_num = "0;"
        Dim col_wid As Integer
        For col_wid = 1 To CDbl(num_dim)
            col_wid_num = col_wid_num & "20;"
            .ColumnWidths = col_wid_num
        Next col_wid
        Dim fill_FLC_box As Integer
        fill_FLC_box = 0
        For ini_samp_size = 1 To 2 ^ CDbl(num_dim)
            For fill_FLC_box = 1 To pd_analysis.num_dim
                .List(ini_samp_size - 1, fill_FLC_box) =
perm_matrix(ini_samp_size, fill_FLC_box)
            Next fill_FLC_box
            flc = ""
        Next ini_samp_size
    End With
    If bud_remain <= 0 Then
        err_msg_1 = MsgBox("The budget entered is not sufficient to begin
sampling. The program will terminate.", vbOKOnly, "Error Handler")
        Exit Sub
    End If
    Erase perm_matrix
    'Fill the sample location/selection list box with the initial two samples
    'Also add location and COV to the matrix of data
    'Counters
```

```
    Dim m As Integer
    Dim n As Integer
    Dim array_inc_count As Integer
    array_inc_count = 0
    Dim comp_count As Integer
    'Set comp_count = 1 from the beginning since we will never insert a new
line before the lower problem bound
    comp_count = 1
    Dim iTemp() As Double  'Temporary array for distance matrix
    Dim iTemp_1() As Double  'Temporary array for COV matrix
    Dim find_inp_index As Integer
    Dim fill_next_can_inp_1 As Integer
    Dim dup As Integer
    Dim det_loc_1 As Integer
    Dim fill_FLC_box_1 As Integer
    'This is the location counter
    Dim loc_int As Integer
    Dim RHS_lag_matrix_TEMP() As Variant
    ReDim RHS_lag_matrix_TEMP(1 To 1, 1 To CDbl(num_dim))
    'This variable is used to find the maximin difference after the variances
don't differ for a given design matrix
    Dim ind_diff As Integer
    '*****This array tracks the order of the array
    Dim fp_inp_index_ordered() As Integer
    ReDim fp_inp_index_ordered(1 To 2 ^ num_dim)
    For init_desgn = 1 To (2 ^ num_dim)
        fp_inp_index_ordered(init_desgn) = CDbl(lb_1.Column(0, init_desgn -
1))
    Next init_desgn
    Dim iloop As Integer
    Dim iloop2 As Integer
    Dim str1 As Integer
    Dim str2 As Integer
    Dim dup_found As Integer
    Dim hit_end As Integer
    Dim move_counter As Integer
    Dim past_data_ind As Integer
    Dim tot_design_check As Integer
    Dim design_check_1 As Integer
    design_check_1 = 0
    Me.neighborhood.Visible = False
    Me.Frame10.Visible = False
    Me.Frame11.Visible = False
    Me.Frame12.Visible = False
    Me.Frame13.Visible = False
    Me.Frame18.Visible = False
    Me.iter.ZOrder (0)
    Me.iter.Visible = True
    Me.outof.Value = UBound(problem_space_array)
    Do While CDbl(lb.ListCount) < UBound(problem_space_array)
```

```
        Me.strt.Value = lb.ListCount
        'Determine end time
        sngEnd = Timer                                  ' Get end time.
        'Determine time elapsed
        sngElapsed = Format(sngEnd - sngStart, "Fixed") ' Elapsed time.
        Me.time_elap.Text = sngElapsed
        DoEvents
        'Me.Repaint
        'Need to perform dynamic array slicing to insert the new values while
rearranging the current values in the dist and COV matrices
        'Temp storage idea taken from
http://www.xtremevbtalk.com/showthread.php?t=82476
        ReDim iTemp(1 To CDbl(2 ^ num_dim) + array_inc_count, 1 To CDbl(2 ^
num_dim) + array_inc_count)
        ReDim iTemp_1(1 To ((CDbl(2 ^ num_dim) + 1) + array_inc_count), 1 To
((CDbl(2 ^ num_dim) + 1) + array_inc_count))
        'Copy original array into temp array:
        For m = 1 To (CDbl(2 ^ num_dim) + array_inc_count)     'Loop for 1st
row
            For n = 1 To (CDbl(2 ^ num_dim) + array_inc_count)  'Loop for
column
                If n <> (CDbl(2 ^ num_dim) + array_inc_count) Then
                    iTemp(m, n) = dist_matrix(m, n)
                End If
                iTemp_1(m, n) = cov_matrix(m, n)
            Next n
        Next m
        'Put values back from temporary array while adding the new design
candidate in the necessary position and adjusting the matrices accordingly
        array_inc_count = array_inc_count + 1
        'For each candidate that gets added during the annealing process, the
matrix will continue to grow
        ReDim dist_matrix(1 To CDbl(2 ^ num_dim) + array_inc_count, 1 To
CDbl(2 ^ num_dim) + array_inc_count)
        ReDim cov_matrix(1 To ((CDbl(2 ^ num_dim) + 1) + array_inc_count), 1
To ((CDbl(2 ^ num_dim) + 1) + array_inc_count))
        'Loop through the current list of FLCs to determine where to insert
the candidate FLC
        'Have to determine which row the cand_input must be inserted into
        'After that, the appropriate distances and COV must be calculated
relative to every other distance in the pilot design
        'First determine which indexed input the candidate input is equal to
        If Me.end_cal = True Then
            If bud_remain <= 0 Then
                With Me.amt_to_paste
                    For past_data_ind = CDbl(lb_1.ListCount) To 1 Step -1
                        .AddItem past_data_ind, 0
                    Next past_data_ind
                End With
                For ei_pop = 2 To CDbl(lb.ListCount)
```

```
                    lb.Column(3, ei_pop) = "-"
                Next ei_pop
                Me.CommandButton1.Visible = True
                'Determine end time
                sngEnd = Timer                                  ' Get end
time.
                'Determine time elapsed
                sngElapsed = Format(sngEnd - sngStart, "Fixed") ' Elapsed
time.
                Me.time_elap.Text = sngElapsed
                Me.iter.Visible = False
                Me.neighborhood.Visible = True
                Me.Frame10.Visible = True
                Me.Frame11.Visible = True
                Me.Frame12.Visible = True
                Me.Frame13.Visible = True
                Me.Frame18.Visible = True
                Exit Sub
            End If
        End If
        For find_inp_index = 1 To CDbl(pd_analysis.max_val) ^ CDbl(num_dim)
            comp_count = 1
            For det_loc_1 = 1 To CDbl(num_dim)
                If RHS_lag_matrix(1, det_loc_1) <=
problem_space_array(find_inp_index, det_loc_1) And RHS_lag_matrix(1,
det_loc_1) >= problem_space_array(find_inp_index, det_loc_1) Then
                    comp_count = comp_count + 1
                End If
            Next det_loc_1
            If comp_count = det_loc_1 Then
                lb_1.AddItem find_inp_index
                For fill_FLC_box_1 = 1 To CDbl(pd_analysis.num_dim)
                    With lb_1
                        .List(lb_1.ListCount - 1, fill_FLC_box_1) =
problem_space_array(find_inp_index, fill_FLC_box_1)
                    End With
                Next fill_FLC_box_1
                Exit For
            End If
        Next find_inp_index
        If design_check_1 = 2 Then
            If CDbl(lb_1.Column(0, lb_1.ListCount - 1)) <=
UBound(problem_space_array) / 2 Then
                dir_flag = 1
            Else
                dir_flag = 0
            End If
        End If
        'The purpose of this is to set the proper direction if the design
search was previously exhausted
```

```
        design_check_1 = 0
        ReDim Preserve fp_inp_index_ordered(1 To lb_1.ListCount)
        fp_inp_index_ordered(lb_1.ListCount) = find_inp_index
        'Sort the array (http://www.ozgrid.com/VBA/sort-array.htm)
        For lLoop = 1 To UBound(fp_inp_index_ordered)
           For lLoop2 = lLoop To UBound(fp_inp_index_ordered)
               If fp_inp_index_ordered(lLoop2) < fp_inp_index_ordered(lLoop)
Then
                   str1 = fp_inp_index_ordered(lLoop)
                   str2 = fp_inp_index_ordered(lLoop2)
                   fp_inp_index_ordered(lLoop) = str2
                   fp_inp_index_ordered(lLoop2) = str1
               End If
            Next lLoop2
        Next lLoop
        'Now the new input has been added to the location list box
        'Next add the new elements to the dist and cov matrix
        m = 0
        n = 0
        move_counter = 0
        For m = LBound(iTemp_1, 1) To (UBound(iTemp_1, 1) - 1)  'Loop through
rows
            For n = 1 To m  'Loop through columns. This only loops through
step-wise half of the matrix. After that, the rest can be filled through
symmetry
                'Need to accomplish two things. One move original elements to
new locations and also need to calculate new data into the inserted elements
                'This calculates new data elements
                'And m <> n ignores the diagonal elements since the
difference in distance between one position and itself is 0
                'If m=n then do nothing as the elements are already 0
                If (n < m) And fp_inp_index_ordered(m) < find_inp_index Then
                    'Nothing moves. Put back into the original array
                    dist_matrix(m, n) = iTemp(m, n)
                    cov_matrix(m, n) = iTemp_1(m, n)
                    dist_matrix(n, m) = dist_matrix(m, n)
                    cov_matrix(n, m) = cov_matrix(m, n)
                ElseIf (n < m) And fp_inp_index_ordered(m) <
fp_inp_index_ordered(m + 1) Then
                    'The reason for this redundacy is Excel will accept a
false True for the above statement when n+1 exceeds the last element of the
list
                    If (n < m) Then
                        'Move original data
                        'Move data down
                        If move_counter = 0 Then move_counter = m
                        If move_counter > n Then
                            dist_matrix(m + 1, n) = iTemp(m, n)
                            cov_matrix(m + 1, n) = iTemp_1(m, n)
                            dist_matrix(n, m + 1) = dist_matrix(m + 1, n)
```

```
                            cov_matrix(n, m + 1) = cov_matrix(m + 1, n)
                        'Move data down and across
                        ElseIf move_counter <= n And move_counter <> 0 Then
                            dist_matrix(m + 1, n + 1) = iTemp(m, n)
                            cov_matrix(m + 1, n + 1) = iTemp_1(m, n)
                            dist_matrix(n + 1, m + 1) = dist_matrix(m + 1, n
+ 1)

                            cov_matrix(n + 1, m + 1) = cov_matrix(m + 1, n +
1)
                        End If
                    End If
                End If
            Next n
        Next m
        'After all elements have been moved, then calculate all new elements.
The location is determined if the new element's current value is 0 and it is
a non-diagonal
        For m = LBound(iTemp_1, 1) To UBound(iTemp_1, 1)      'Loop through
rows
            For n = 1 To m  'Loop through columns. This only loops through
step-wise half of the matrix. After that, the rest can be filled
                If n <> m And dist_matrix(m, n) = 0 And find_inp_index >=
fp_inp_index_ordered(m) Then
                    'This calculates and adds new data elements
                    f_euc_cal_0 = 0
                    For f_dim_count_0 = 1 To CDbl(num_dim)
                        f_euc_cal_0 = f_euc_cal_0 +
(problem_space_array(find_inp_index, f_dim_count_0) -
problem_space_array(fp_inp_index_ordered(n), f_dim_count_0)) ^ 2
                    Next
                    dist_matrix(m, n) = Sqr(f_euc_cal_0)
                    'Due to Symmetry
                    dist_matrix(n, m) = Sqr(f_euc_cal_0)
                    'Reset range to the max value
                    a = WorksheetFunction.Max(dist_matrix)
                    cov_matrix(m, n) = nugget + c1 * (1 - Exp((-3 *
dist_matrix(m, n)) / a))
                    cov_matrix(n, m) = nugget + c1 * (1 - Exp((-3 *
dist_matrix(m, n)) / a))
                    'End new data element calculation
                ElseIf n <> m And dist_matrix(m, n) = 0 And find_inp_index <
fp_inp_index_ordered(m) Then
                    'This calculates and adds new data elements
                    f_euc_cal_0 = 0
                    For f_dim_count_0 = 1 To CDbl(num_dim)
                        f_euc_cal_0 = f_euc_cal_0 +
(problem_space_array(find_inp_index, f_dim_count_0) -
problem_space_array(fp_inp_index_ordered(m), f_dim_count_0)) ^ 2
                    Next
                    dist_matrix(m, n) = Sqr(f_euc_cal_0)
```

```
                    'Due to Symmetry
                    dist_matrix(n, m) = Sqr(f_euc_cal_0)
                    'Reset range to the max value
                    a = WorksheetFunction.Max(dist_matrix)
                    cov_matrix(m, n) = nugget + c1 * (1 - Exp((-3 *
dist_matrix(m, n)) / a))
                    cov_matrix(n, m) = nugget + c1 * (1 - Exp((-3 *
dist_matrix(m, n)) / a))
                    'End new data element calculation
                End If
                'Lagrangian multiplier
                If (n = m) Then
                    cov_matrix(m, UBound(iTemp_1, 1) + 1) = 1
                    cov_matrix(UBound(iTemp_1, 1) + 1, m) = 1
                End If
            Next n
        Next m
        'Reset the location counter to offset by one each time
        loc_int = 1
        ReDim cand_input(1 To CDbl(lb_1.ListCount) + 1)
        ReDim RHS_lag_matrix(1 To 1, 1 To CDbl(num_dim))
        ReDim RHS_dis_matrix(1 To CDbl(lb_1.ListCount), 1 To 1)
        'Start searching for the worst input

'**************************************************************************
        'Set initial var_new variable
        var_new = 0
        'Search for a better solution. If found add to the pilot design,
redirect the directional flag and continue by exiting loop
        'Stop looking on the last FLC and directly add that to the list.
        Do While (CDbl(lb.ListCount) <> UBound(problem_space_array))
            dup_found = 1
            hit_end = 0
            'search by using dist_mat then set it to RHS
            For fill_next_can_inp = 1 To CDbl(lb_1.ListCount)
                If dir_flag = 0 Then
                    f_euc_cal_0 = 0
                    For f_dim_count_0 = 1 To CDbl(num_dim)
                        RHS_lag_matrix(1, f_dim_count_0) =
problem_space_array(find_inp_index - loc_int, f_dim_count_0)
                        f_euc_cal_0 = f_euc_cal_0 +
(problem_space_array(fp_inp_index_ordered(fill_next_can_inp), f_dim_count_0)
- RHS_lag_matrix(1, f_dim_count_0)) ^ 2
                    Next
                    RHS_dis_matrix(fill_next_can_inp, 1) = Sqr(f_euc_cal_0)
                    'Need to check if RHS lag has already been used, if so
increment loc_int
                    'The IF statement only checks the candidate input against
the list once to see if it is a duplicate
                    If dup_found = 1 Then
```

```
                        For fill_next_can_inp_1 = 1 To CDbl(lb_1.ListCount)
                            dup = 0
                            For f_dim_count_0 = 1 To CDbl(num_dim)
                                If RHS_lag_matrix(1, f_dim_count_0) =
CDbl(lb_1.Column(f_dim_count_0, fill_next_can_inp_1 - 1)) Then
                                    dup = dup + 1
                                End If
                            Next f_dim_count_0
                            If dup = CDbl(num_dim) Then
                                If CDbl(lb_1.Column(0, fill_next_can_inp_1 -
1)) = LBound(problem_space_array) Or (CDbl(lb_1.Column(0, fill_next_can_inp_1
- 1) = UBound(problem_space_array))) Then
                                    If (krig_var = var_best And
CDbl(lb_1.ListCount) <> UBound(problem_space_array)) Or (var_best = 0 And
CDbl(lb_1.ListCount) <> UBound(problem_space_array)) Then
                                        'Reset back to starting point
                                        For f_dim_count_0 = 1 To
CDbl(num_dim)
                                            RHS_lag_matrix(1, f_dim_count_0)
= CDbl(lb_1.Column(f_dim_count_0, CDbl(lb_1.ListCount - 1)))
                                        Next
                                        loc_int = 0
                                        'If no better variance is found by
searching the entire direction
                                        tot_design_check = tot_design_check +
1
                                        'Change the direction
                                        dir_flag = 1
                                    Else
                                        hit_end = 1
                                        fill_next_can_inp =
CDbl(lb_1.ListCount)
                                        loc_int = 1
                                    End If
                                    RHS_dis_matrix(1, 1) = temp_rhs_var
                                End If
                                loc_int = loc_int + 1
                                'If the input is found to be a duplicate,
then start the process over
                                If hit_end = 0 And tot_design_check <> 2 Then
                                    fill_next_can_inp = 0
                                Else
                                    fill_next_can_inp = CDbl(lb_1.ListCount)
                                End If
                                dup_found = 1
                                Exit For
                            Else
                                dup_found = 0
                            End If
                        Next fill_next_can_inp_1
```

```
                    End If
                ElseIf dir_flag = 1 Then
                    f_euc_cal_0 = 0
                    For f_dim_count_0 = 1 To CDbl(num_dim)
                        RHS_lag_matrix(1, f_dim_count_0) =
problem_space_array(find_inp_index + loc_int, f_dim_count_0)
                        f_euc_cal_0 = f_euc_cal_0 +
(problem_space_array(fp_inp_index_ordered(fill_next_can_inp), f_dim_count_0)
- RHS_lag_matrix(1, f_dim_count_0)) ^ 2
                    Next f_dim_count_0
                    RHS_dis_matrix(fill_next_can_inp, 1) = Sqr(f_euc_cal_0)
                    'Need to check if RHS lag has already been used, if so
increment loc_int
                    'The IF statement only checks the candidate input against
the list once to see if it is a duplicate
                    If dup_found = 1 Then
                        For fill_next_can_inp_1 = 1 To CDbl(lb_1.ListCount)
                            dup = 0
                            For f_dim_count_0 = 1 To CDbl(num_dim)
                                If RHS_lag_matrix(1, f_dim_count_0) =
CDbl(lb_1.Column(f_dim_count_0, fill_next_can_inp_1 - 1)) Then
                                    dup = dup + 1
                                End If
                            Next
                            If dup = CDbl(num_dim) Then
                                If CDbl(lb_1.Column(0, fill_next_can_inp_1 -
1)) = LBound(problem_space_array) Or CDbl(lb_1.Column(0, fill_next_can_inp_1
- 1)) = UBound(problem_space_array) Then
                                    'Need to add provision for hitting the
end of the design and not having found a better solution. Have to reverse the
direction and start over from the previous starting point.
                                    If (krig_var = var_best And
CDbl(lb_1.ListCount) <> UBound(problem_space_array)) Or (var_best = 0 And
CDbl(lb_1.ListCount) <> UBound(problem_space_array)) Then
                                        'Reset back to starting point
                                        For f_dim_count_0 = 1 To
CDbl(num_dim)
                                            RHS_lag_matrix(1, f_dim_count_0)
= CDbl(lb_1.Column(f_dim_count_0, CDbl(lb_1.ListCount - 1)))
                                        Next
                                        loc_int = 0
                                        'If no better variance is found by
searching the entire direction
                                        tot_design_check = tot_design_check +
1
                                        'Change the direction
                                        dir_flag = 0
                                    Else
                                        hit_end = 1
```

```
                                        fill_next_can_inp =
CDbl(lb_1.ListCount)
                                        loc_int = 1
                                    End If
                                    RHS_dis_matrix(1, 1) = temp_rhs_var
                                End If
                                loc_int = loc_int + 1
                                'If the input is found to be a duplicate,
then start the process over unless it is at the beginning or ending number
and we have not searched the whole problem space
                                If hit_end = 0 And tot_design_check <> 2 Then
                                    fill_next_can_inp = 0
                                Else
                                    fill_next_can_inp = CDbl(lb_1.ListCount)
                                End If
                                dup_found = 1
                                Exit For
                            Else
                                dup_found = 0
                            End If
                        Next fill_next_can_inp_1
                    End If
                End If
            Next fill_next_can_inp
            For fill_next_can_inp = 1 To CDbl(lb_1.ListCount)
                cand_input(fill_next_can_inp) = nugget + c1 * (1 - Exp((-3 *
RHS_dis_matrix(fill_next_can_inp, 1)) / a))
            Next fill_next_can_inp
            cand_input(fill_next_can_inp) = 1
            krig_var = 0
            'Reset Design Check
            'Calculate weights
            ArrInv() = Application.MInverse(cov_matrix)
            ArrAns() = Application.MMult(ArrInv,
Application.Transpose(cand_input))
            For krig_count = 1 To lb_1.ListCount + 1
                krig_var = krig_var + ArrAns(krig_count, 1) *
cand_input(krig_count)
            Next
            loc_int = loc_int + 1
            'The first option contines to track the RHS_lag_matrix while
better solutions are found
            If hit_end = 0 And tot_design_check <> 2 Then
                If (Round(krig_var, 7) >= Round(var_best, 7) And krig_var_old
>= var_best) Then
                    var_best = krig_var
                    For f_dim_count_0 = 1 To CDbl(num_dim)
                        RHS_lag_matrix_TEMP(1, f_dim_count_0) =
RHS_lag_matrix(1, f_dim_count_0)
                    Next
```

```
                End If
                'This temp variable is used to restore the 1st RHS value
after the program hits the beginning or end while searching for better
solutions
                temp_rhs_var = RHS_dis_matrix(1, 1)
            Else
                If dir_flag = 0 Then
                    dir_flag = 1
                ElseIf dir_flag = 1 Then
                    dir_flag = 0
                End If
                design_check_1 = tot_design_check
                tot_design_check = 0
                If CDbl(pd_analysis.lin_cost) = 0 Then
                    bud_remain = bud_remain - cost_tot
                Else
                    bud_remain = bud_remain - (CDbl(pd_analysis.lin_cost) *
cost_tot_new)
                    cost_tot_new = (CDbl(pd_analysis.lin_cost) *
cost_tot_new)
                End If
                sum_var = sum_var + var_best
                With lb
                    .AddItem (.ListCount + 1)
                    .Column(1, .ListCount - 1) = Round(var_best, 7)
                    .Column(2, .ListCount - 1) = bud_remain
                    If bud_remain <= 0 And Me.amt_to_paste.Value = "" Then
                        Me.amt_to_paste.Value = .ListCount
                    End If
                End With
                For f_dim_count_0 = 1 To CDbl(num_dim)
                    RHS_lag_matrix(1, f_dim_count_0) = RHS_lag_matrix_TEMP(1,
f_dim_count_0)
                Next
                'This temp variable is used to restore the RHS value after
the program hits the beginning or end while searching for better solutions
                temp_rhs_var = RHS_dis_matrix(1, 1)
                krig_var_old = var_best
                var_best = 0
                loc_int = 1
                Exit Do
            End If
        Loop
        'This code is to paste the final FLC into the lists
        If (CDbl(lb.ListCount) = UBound(problem_space_array)) Then
            '1st have to determine which FLC is the last
            For fill_next_can_inp = 1 To (UBound(fp_inp_index_ordered) - 1)
                final_FLC = final_FLC + 1
                If Abs(fp_inp_index_ordered(fill_next_can_inp) -
fp_inp_index_ordered(fill_next_can_inp + 1)) > 1 Then
```

```
                    For f_dim_count_0 = 1 To CDbl(num_dim)
                        RHS_lag_matrix(1, f_dim_count_0) =
problem_space_array(fp_inp_index_ordered(fill_next_can_inp) + 1,
f_dim_count_0)
                    Next f_dim_count_0
                    final_FLC = final_FLC + 1
                    Exit For
                End If
            Next fill_next_can_inp
            For fill_next_can_inp = 1 To CDbl(lb_1.ListCount)
                f_euc_cal_0 = 0
                For f_dim_count_0 = 1 To CDbl(num_dim)
                    f_euc_cal_0 = f_euc_cal_0 +
(problem_space_array(fp_inp_index_ordered(fill_next_can_inp), f_dim_count_0)
- RHS_lag_matrix(1, f_dim_count_0)) ^ 2
                Next
                RHS_dis_matrix(fill_next_can_inp, 1) = Sqr(f_euc_cal_0)
            Next fill_next_can_inp
            For fill_next_can_inp = 1 To CDbl(lb_1.ListCount)
                cand_input(fill_next_can_inp) = nugget + c1 * (1 - Exp((-3 *
RHS_dis_matrix(fill_next_can_inp, 1)) / a))
            Next fill_next_can_inp
            cand_input(fill_next_can_inp) = 1
            krig_var = 0
            'Calculate weights
            ArrInv() = Application.MInverse(cov_matrix)
            ArrAns() = Application.MMult(ArrInv,
Application.Transpose(cand_input))
            For krig_count = 1 To lb_1.ListCount + 1
                krig_var = krig_var + ArrAns(krig_count, 1) *
cand_input(krig_count)
                var_best = krig_var
            Next
            If CDbl(pd_analysis.lin_cost) = 0 Then
                bud_remain = bud_remain - cost_tot
            Else
                bud_remain = bud_remain - (CDbl(pd_analysis.lin_cost) *
cost_tot_new)
                cost_tot_new = (CDbl(pd_analysis.lin_cost) * cost_tot_new)
            End If
            With lb
                .AddItem (.ListCount + 1)
                .Column(1, .ListCount - 1) = Round(var_best, 7)
                .Column(2, .ListCount - 1) = bud_remain
                If bud_remain <= 0 And Me.amt_to_paste.Value = "" Then
                    Me.amt_to_paste.Value = .ListCount
                End If
                sum_var = sum_var + var_best
            End With
            With lb_1
```

```
                .AddItem (final_FLC)
                For f_dim_count_0 = 1 To CDbl(num_dim)
                    .Column(f_dim_count_0, .ListCount - 1) =
RHS_lag_matrix(1, f_dim_count_0)
                Next f_dim_count_0
            End With
        End If
    Loop
    'This statements purges the original inp_out array since it is not used
in the code anymore.
    Erase iTemp
    Erase iTemp_1
    '***********End augmented simulated annealing algorithm************
    With Me.amt_to_paste
        For past_data_ind = CDbl(lb_1.ListCount) To 1 Step -1
            .AddItem past_data_ind, 0
        Next past_data_ind
    End With
    For ei_pop = CDbl(2 ^ num_dim) To CDbl(lb.ListCount)
        lb.Column(3, ei_pop) = Round((CDbl(lb.Column(1, ei_pop)) / sum_var) *
100, 2)
    Next ei_pop
    'Determine end time
    sngEnd = Timer                                ' Get end time.
    'Determine time elapsed
    sngElapsed = Format(sngEnd - sngStart, "Fixed") ' Elapsed time.
    Me.time_elap.Text = sngElapsed
    Me.iter.Visible = False
    Me.neighborhood.Visible = True
    Me.Frame10.Visible = True
    Me.Frame11.Visible = True
    Me.Frame12.Visible = True
    Me.Frame13.Visible = True
    Me.Frame18.Visible = True
    Me.CommandButton1.Visible = True
End Sub
'Sort code from postman2000 at
http://www.ozgrid.com/forum/showthread.php?t=71509
Function SortListBox(oLb As MSForms.ListBox, sCol As Integer, sType As
Integer, sDir As Integer)
    Dim vaItems As Variant
    Dim i As Long, j As Long
    Dim c As Integer
    Dim vTemp As Variant
     'Put the items in a variant array
    vaItems = oLb.List
     'Sort the Array Alphabetically(1)
    If sType = 1 Then
        For i = LBound(vaItems, 1) To UBound(vaItems, 1) - 1
            For j = i + 1 To UBound(vaItems, 1)
```

```
                'Sort Ascending (1)
                If sDir = 1 Then
                    If vaItems(i, sCol) > vaItems(j, sCol) Then
                        For c = 0 To oLb.ColumnCount - 1 'Allows sorting of
multi-column ListBoxes
                            vTemp = vaItems(i, c)
                            vaItems(i, c) = vaItems(j, c)
                            vaItems(j, c) = vTemp
                        Next c
                    End If
                    'Sort Descending (2)
                ElseIf sDir = 2 Then
                    If vaItems(i, sCol) < vaItems(j, sCol) Then
                        For c = 0 To oLb.ColumnCount - 1 'Allows sorting of
multi-column ListBoxes
                            vTemp = vaItems(i, c)
                            vaItems(i, c) = vaItems(j, c)
                            vaItems(j, c) = vTemp
                        Next c
                    End If
                End If
            Next j
        Next i
         'Sort the Array Numerically(2)
         '(Substitute CInt with another conversion type (CLng, CDec, etc.)
depending on type of numbers in the column)
    ElseIf sType = 2 Then
        For i = LBound(vaItems, 1) To UBound(vaItems, 1) - 1
            For j = i + 1 To UBound(vaItems, 1)
                'Sort Ascending (1)
                If sDir = 1 Then
                    If CInt(vaItems(i, sCol)) > CInt(vaItems(j, sCol)) Then
                        For c = 0 To oLb.ColumnCount - 1 'Allows sorting of
multi-column ListBoxes
                            vTemp = vaItems(i, c)
                            vaItems(i, c) = vaItems(j, c)
                            vaItems(j, c) = vTemp
                        Next c
                    End If
                    'Sort Descending (2)
                ElseIf sDir = 2 Then
                    If CInt(vaItems(i, sCol)) < CInt(vaItems(j, sCol)) Then
                        For c = 0 To oLb.ColumnCount - 1 'Allows sorting of
multi-column ListBoxes
                            vTemp = vaItems(i, c)
                            vaItems(i, c) = vaItems(j, c)
                            vaItems(j, c) = vTemp
                        Next c
                    End If
                End If
```

```
            Next j
        Next i
    End If
     'Set the list to the array
    oLb.List = vaItems
End Function
'Post data to sheet
Private Sub CommandButton1_Click()
    If Me.sam_location.ListCount = 0 Then
        err_msg_1 = MsgBox("You must run the test planning section above
prior to pasting data into the worksheet.", vbOKOnly, "Error Handler")
        Exit Sub
    End If
    If pd_analysis.amt_to_paste.Value <> "" Then
        If CDbl(pd_analysis.amt_to_paste.Value) < 1 Then
            err_msg_1 = MsgBox("The sample size to paste must be an integer
greater than 1.", vbOKOnly, "Error Handler")
            Exit Sub
        End If
    End If
    'Calculate the amount to paste
    Dim past_counter As Integer
    'Dim sum_var_1 As Double
    If Me.var_neg = True Then
        For ei_pop = 1 To (CDbl(Me.sam_selection.ListCount) - 1)
            sum_var_1 = sum_var_1 + CDbl(Me.sam_selection.Column(3, ei_pop +
2 ^ CDbl(Me.num_dim) - 1))
            past_counter = past_counter + 1
            If sum_var_1 > CDbl(Me.ComboBox1.Value) Then
                Me.amt_to_paste.Value = CDbl(Me.amt_to_paste.Value) + 2 ^
CDbl(Me.num_dim)
                Exit For
            End If
            Me.amt_to_paste.Value = past_counter
        Next ei_pop
    End If
    Dim past_data As Integer
    Worksheets("Test_Planning").Range("J1").Value = "n="
    Worksheets("Test_Planning").Range("K1").Value =
CDbl(pd_analysis.amt_to_paste.Value)
    Worksheets("Test_Planning").Range("J" & 3 +
CDbl(Me.amt_to_paste.Value)).Value = "Pilot Design FLCs"
    Worksheets("Test_Planning").Range("J2").Value = "Sample Size"
    Worksheets("Test_Planning").Range("K2").Value = "Kriging Variance"
    Worksheets("Test_Planning").Range("L2").Value = "Balance($)"
    Worksheets("Test_Planning").Range("M2").Value = "EI"
    For remov_data = CDbl(Me.sam_selection.ListCount - 1) To
(CDbl(Me.amt_to_paste.Value) + 1) Step -1
        Me.sam_location.RemoveItem (Me.sam_location.ListCount - 1)
    Next remov_data
```

```
    'Sort by the 1st column in the ListBox Numerically in Ascending Order
    Dim lb_1 As MSForms.ListBox
    Set lb_1 = Me.sam_location
    Evaluate SortListBox(lb_1, 0, 2, 1)
    Dim amount_pasted As Integer
    amount_pasted = CDbl(Me.amt_to_paste.Value)
    With Worksheets("Test_Planning")
        For past_data = 1 To amount_pasted
            With Worksheets("Test_Planning").Cells(2 + past_data,
10).Interior
                .Pattern = xlSolid
                .PatternColorIndex = xlAutomatic
                .ThemeColor = xlThemeColorDark2
                .TintAndShade = -9.99786370433668E-02
                .PatternTintAndShade = 0
            End With
            With Worksheets("Test_Planning").Cells(2 + past_data,
11).Interior
                .Pattern = xlSolid
                .PatternColorIndex = xlAutomatic
                .ThemeColor = xlThemeColorDark2
                .TintAndShade = -9.99786370433668E-02
                .PatternTintAndShade = 0
            End With
            With Worksheets("Test_Planning").Cells(2 + past_data,
12).Interior
                .Pattern = xlSolid
                .PatternColorIndex = xlAutomatic
                .ThemeColor = xlThemeColorDark2
                .TintAndShade = -9.99786370433668E-02
                .PatternTintAndShade = 0
            End With
            With Worksheets("Test_Planning").Cells(2 + past_data,
13).Interior
                .Pattern = xlSolid
                .PatternColorIndex = xlAutomatic
                .ThemeColor = xlThemeColorDark2
                .TintAndShade = -9.99786370433668E-02
                .PatternTintAndShade = 0
            End With
            'First list box items
            .Range("J" & 2 + past_data).Value = Me.sam_selection.Column(0,
past_data - 1)
            .Range("K" & 2 + past_data).Value = Me.sam_selection.Column(1,
past_data - 1)
            .Range("L" & 2 + past_data).Value = Me.sam_selection.Column(2,
past_data - 1)
            .Range("M" & 2 + past_data).Value = Me.sam_selection.Column(3,
past_data - 1)
            For f_dim_count_0 = 1 To CDbl(num_dim)
```

```
                With Worksheets("Test_Planning").Cells(5 + amount_pasted +
past_data - 1, 9 + f_dim_count_0).Interior
                    .Pattern = xlSolid
                    .PatternColorIndex = xlAutomatic
                    .ThemeColor = xlThemeColorAccent6
                    .TintAndShade = 0.799981688894314
                    .PatternTintAndShade = 0
                End With
                .Range((Chr(73 + f_dim_count_0)) & 5 + amount_pasted +
past_data - 1).Value = CDbl(Me.sam_location.Column(f_dim_count_0, past_data -
1))
            Next f_dim_count_0
        Next past_data
        For f_dim_count_0 = 1 To CDbl(num_dim)
            .Range((Chr(73 + f_dim_count_0)) & 4 + amount_pasted).Value = "X"
& f_dim_count_0
        Next f_dim_count_0
        .Range((Chr(73 + f_dim_count_0)) & 4 + amount_pasted).Value = "Y"
    End With
    Columns("J:J").EntireColumn.AutoFit
    Columns("K:K").EntireColumn.AutoFit
    Columns("L:L").EntireColumn.AutoFit
    Columns("M:M").EntireColumn.AutoFit
    Unload pd_analysis
End Sub
```

Appendix A.3: Main DFK Kriging Form

```
Option Explicit
Private Sub CheckBox1_Click()
    If SemiForm.Label17.Enabled = False Then
        SemiForm.Label17.Enabled = True
        SemiForm.ComboBox1.Enabled = True
    Else
        SemiForm.Label17.Enabled = False
        SemiForm.ComboBox1.Enabled = False
    End If
End Sub

Private Sub exp_var_confirm_Click()
    If SemiForm.exp_var_select <> "Best Estimate (Recommended)" Then
        SemiForm.nug.Enabled = False
        SemiForm.sil.Enabled = True
        SemiForm.ran.Enabled = True
    Else
        SemiForm.nug.Enabled = False
        SemiForm.sil.Enabled = False
        SemiForm.ran.Enabled = False
    End If
```

```
End Sub
'*******Help information**********************************
Private Sub Image3_Click()
    Krig_Help.TextBox1.Text = "This is where the user selects which inputs to
use. The selection is to be made by columns. If the selected inputs are a
single column " & _
        "then it is considered a single dimensional problem. If multiple
columns are selected then it is considered a multidimensional problem. Each
row is considered " & _
        "a seperate input value."
    Krig_Help.Show
End Sub
Private Sub Image4_Click()
    Krig_Help.TextBox1.Text = "This is the lag or also known as binning as it
takes the calculated differences and puts them into predetermined bins. This
speeds up " & _
        "calculation time. The lag aides in properly calculating the
experimental semivariogram. It generally will reduce the number of
experimental semivariogram points therefore causing the potential for using a
fitted model that yields " & _
        "a less accurate prediction value. Selecting (Default) will allow the
software to try to properly determine the lag/bin size. It is recommended
unless a certain lag is required or even preferred."
    Krig_Help.Show
End Sub
Private Sub Image5_Click()
    Krig_Help.TextBox1.Text = "This is the neighborhood. The purpose here is
to set bounds for data where the data may not have a strong correlation due
to " & _
        "the large distances apart. For example, it is hard to say that a
sample of soil data taken in the United States is correlated with a soil
sample " & _
        "taken in Turkey. So, this option allows you to rule out those types
of values."
    Krig_Help.Show
End Sub
Private Sub Image6_Click()
    Krig_Help.TextBox1.Text = "This selection is the output. It should
directly correspond to the input values, have an equal number of " & _
        "values as the inputs, and is limited to a single column."
    Krig_Help.Show
End Sub
Private Sub Image7_Click()
    Krig_Help.TextBox1.Text = "Here is where the user can select specific
fitted models to use. If a specific model is selected then " & _
        "the user must specify a nugget, sill, and range value all of which
must be greater than or equal to zero. It is recommended to let " & _
        "the software determine the best fitted model. It accomplishes this
through two methods. One method regressors the variogram and then performs "
& _
```

```
        "GRG or EA. The second method perfroms multisampling of various
posssible values and selects " & _
        "whichever model results in the lowest MSE between the experimental
semivariogram and the fitted models. NOTE: If EA is required the software may
require " & _
        "more CPU time to process the results."
    Krig_Help.Show
End Sub
Private Sub Image8_Click()
    Krig_Help.TextBox1.Text = "Here is where the user can select whichever
Kriging type best suits the need of the problem at hand. " & _
        "The assumptions should give some further insight into the proper
selection although Ordinary Kriging is the generally the most common."
    Krig_Help.Show
End Sub
Private Sub Image9_Click()
    Krig_Help.TextBox1.Text = "Here is where the user can input the nugget,
sill, and range if a specific fitted model has been chosen. " & _
        "The nugget value is the y-intercept value when looking at the fitted
models on a graph. If this value is non zero (discontinous) " & _
        "then noise may be present or potential measurement errors may exist
(Chiles, 1999). The sill is the limited value of the variogram. " & _
        "It states the distance where Z(h) and Z(x+h) start to become
uncorrelated (Chiles, 1999). The range is the x-axis distance value that " &
_
        "corresponds to the sill value."
    Krig_Help.Show
End Sub
Private Sub Image10_Click()
    Krig_Help.TextBox1.Text = "This is essentially the point at which to be
predicted. A single column selection corresponds with a single dimension " &
_
        "and a multicoulmn selection corresponds to a multi-dimensionsal
point. If the resolution is entered then this value is the starting value
prediction value of the series."
    Krig_Help.Show
End Sub
Private Sub Image11_Click()
    Krig_Help.TextBox1.Text = "This is the resolution in which to perform
multiple Kriging predictions at one time. " & _
        "The predictions will begin at beginning FLC and end once the
prediction value is outside " & _
        "the upper bound of the original input data set. The increment
counter will be set to the value of the first dimension of the first FLC."
    Krig_Help.Show
End Sub
'*******End Help information*********************************
Private Sub krigtype_AfterUpdate()
    'Populate the Kriging Assumptions in the GUI
    If SemiForm.KrigType = "Ordinary Kriging" Then
```

```
        SemiForm.KrigAssum.Text = "-Data is spatially correlated in Euclydian
space i.e. the closer the distance between them the more correlated." &
Chr(13) & "-The number of values per input variable has a large range." &
Chr(13) & "-The predictions have the exact same value as the observed
observation output."
    'ElseIf SemiForm.KrigType = "Universal Kriging" Then
    '    SemiForm.KrigAssum.Text = "Test."
    End If
End Sub
Private Sub CommandButton1_Click()
    Dim OriginalSampleRange, OutSample As String
    Dim err_msg_1
    'Get user input data
    'Get exact location of the cell as a string which can be used in a Range
Object with all usual properties
    'Err Handling
    If SemiForm.RefEditOriginalSampleRange.Value = "" Then
        err_msg_1 = MsgBox("You did not select any inputs. The program will
exit now.", vbOKOnly, "Error Handler")
        Exit Sub
    End If
    If SemiForm.Refoutdata.Value = "" Then
        err_msg_1 = MsgBox("You did not select any inputs. The program will
exit now.", vbOKOnly, "Error Handler")
        Exit Sub
    End If
    'End Err Handling
    OriginalSampleRange =
Range(SemiForm.RefEditOriginalSampleRange.Value).Address(external:=True)
    ' Turn off screen updating
    Application.ScreenUpdating = True
    'Call the main module to do all the calculations
    calc_semiv_model_value (OriginalSampleRange)
    Unload Me
End Sub
Private Sub CommandButton2_Click()
    ' Put away the form
    Unload SemiForm
    Exit Sub
End Sub
```

Appendix A.4: DFK Help Form

```
Private Sub CommandButton1_Click()
    Unload Me
End Sub
```

Appendix A.5: Introduction Form

```
Private Sub CommandButton1_Click()
    ' Put away the form
    Unload Me
    'Load form
    pd_analyze_start
End Sub
Private Sub CommandButton2_Click()
    ' Put away the form
    Unload Me
    'Load form
    ShowSemiForm
End Sub
Private Sub CommandButton3_Click()
    ' Put away the form
    Unload Kriging_Intro
    Exit Sub
End Sub
Private Sub CommandButton5_Click()
    Krig_Help.TextBox1.Text = "There are several forms of Kriging, all with
the intent to estimate a continuous, spatial attribute at an unsampled site.
Kriging is a form of generalized linear regression for the formulation of an
optimal estimator in a minimum mean square error sense.  Simple Kriging
provides a gateway into more detailed methods of Kriging.  Simple Kriging is
limited due to its simplicity and embedded assumptions.  Ordinary Kriging,
the method used in this research, is the most widely used Kriging method and
is based off of many of the principles founded in simple Kriging.  The
acronym B.L.U.E is associated with ordinary Kriging.  The acronym stands for
best linear unbiased estimator.  Ordinary Kriging is linear since it
estimates weighted linear combinations of data.  It is unbiased since it
tries to have the mean residual equal to 0." _
    & "Finally, ordinary Kriging is considered best since it tries to
minimize the variance of the errors.  Practically speaking, the goal of
ordinary Kriging is unattainable as the mean error and variance are always
unknown.  This implies that it cannot guarantee the mean error is equal to 0
or that the variance is minimized.  The best attempt is to build a model of
the data that is available and work with the average error and the error
variance.  In ordinary Kriging, a probability model is used such that the
bias and error variance can both be calculated.  By choosing weights for
nearby samples this ensures that the average error for the model is exactly 0
and that the error variance is minimized."
    Krig_Help.Show
End Sub
```

Appendix A.6: Results Form

```
Private Sub CommandButton1_Click()
```

```
    Unload Results_Form
End Sub
```

Appendix A.7: Workbook Code to Add Ribbon Bar Menu

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' START ThisWorkbook Code Module
' Created By Chip Pearson, chip@cpearson.com
' Sample code for Creating An Add-In at
http://www.cpearson.com/Excel/CreateAddIn.aspx
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''
Option Explicit

Private Const C_TAG = "Kriging" ' C_TAG should be a string unique to this
add-in.
Private Const C_TOOLS_MENU_ID As Long = 30007&

Private Sub Workbook_Open()
''''''''''''''''''''''''''''''''''''''''''''''
' Workbook_Open
' Create a submenu on the Tools menu. The
' submenu has two controls on it.
''''''''''''''''''''''''''''''''''''''''''''''
Dim ToolsMenu As Office.CommandBarControl
Dim ToolsMenuItem As Office.CommandBarControl
Dim ToolsMenuControl As Office.CommandBarControl

''''''''''''''''''''''''''''''''''''''''''''''
' First delete any of our controls that
' may not have been properly deleted previously.
''''''''''''''''''''''''''''''''''''''''''''''
DeleteControls

'''''''''''''''''''''''''''''''''''''''''''''
' Get a reference to the Tools menu.
'''''''''''''''''''''''''''''''''''''''''''''
Set ToolsMenu = Application.CommandBars.FindControl(ID:=C_TOOLS_MENU_ID)
If ToolsMenu Is Nothing Then
    MsgBox "Unable to access Tools menu.", vbOKOnly
    Exit Sub
End If

'''''''''''''''''''''''''''''''''''''''''''''
' Create a item on the Tools menu.
'''''''''''''''''''''''''''''''''''''''''''''
```

```
Set ToolsMenuItem = ToolsMenu.Controls.Add(Type:=msoControlPopup,
temporary:=True)
If ToolsMenuItem Is Nothing Then
    MsgBox "Unable to add item to the Tools menu.", vbOKOnly
    Exit Sub
End If

With ToolsMenuItem
    .Caption = "&Kriging"
    .BeginGroup = True
    .Tag = C_TAG
    '.OnAction = "'" & ThisWorkbook.Name & "'!Kriging_start"
End With

''''''''''''''''''''''''''''''''''''''''''''''''
' Create the first control on the new item
' in the Tools menu.
''''''''''''''''''''''''''''''''''''''''''''''''
Set ToolsMenuControl = ToolsMenuItem.Controls.Add(Type:=msoControlButton,
temporary:=True)
If ToolsMenuControl Is Nothing Then
    MsgBox "Unable to add item to Tools menu item.", vbOKOnly
    Exit Sub
End If

With ToolsMenuControl
    ''''''''''''''''''''''''''''''''''''''
    ' Set the display caption and the
    ' procedure to run when clicked.
    ''''''''''''''''''''''''''''''''''''''
    .Caption = "Kriging Start"
    .OnAction = "'" & ThisWorkbook.Name & "'!Kriging_start"
    .Tag = C_TAG
End With

'With ToolsMenuControl
    ''''''''''''''''''''''''''''''''''''''
    ' Set the display caption and the
    ' procedure to run when clicked.
    ''''''''''''''''''''''''''''''''''''''
'    .Caption = "Get FLC"
'    .OnAction = "'" & ThisWorkbook.Name & "'!get_flc"
'    .Tag = C_TAG
'End With

''''''''''''''''''''''''''''''''''''''''''''''''
' Create the first control on the new item
' in the Tools menu.
''''''''''''''''''''''''''''''''''''''''''''''''
```

```
Set ToolsMenuControl = ToolsMenuItem.Controls.Add(Type:=msoControlButton,
temporary:=True)
If ToolsMenuControl Is Nothing Then
    MsgBox "Unable to add item to Tools menu item.", vbOKOnly
    Exit Sub
End If

With ToolsMenuControl
    ''''''''''''''''''''''''''''''''''''
    ' Set the display caption and the
    ' procedure to run when clicked.
    ''''''''''''''''''''''''''''''''''''
    .Caption = "Test Planning"
    .OnAction = "'" & ThisWorkbook.Name & "'!pd_analyze_start"
    .Tag = C_TAG
End With

Set ToolsMenuControl = ToolsMenuItem.Controls.Add(Type:=msoControlButton,
temporary:=True)
If ToolsMenuControl Is Nothing Then
    MsgBox "Unable to add item to Tools menu item.", vbOKOnly
    Exit Sub
End If

With ToolsMenuControl
    ''''''''''''''''''''''''''''''''''''
    ' Set the display caption and the
    ' procedure to run when clicked.
    ''''''''''''''''''''''''''''''''''''
    .Caption = "Sequential Kriging"
    .OnAction = "'" & ThisWorkbook.Name & "'!ShowSemiForm"
    .Tag = C_TAG
End With

End Sub


Private Sub Workbook_BeforeClose(Cancel As Boolean)
''''''''''''''''''''''''''''''''''''''''''''''''''''
' Workbook_BeforeClose
' Before closing the add-in, clean up our controls.
''''''''''''''''''''''''''''''''''''''''''''''''''''
    DeleteControls
End Sub


Private Sub DeleteControls()
''''''''''''''''''''''''''''''''''''
' Delete controls whose Tag is
' equal to C_TAG.
```

```
''''''''''''''''''''''''''''''''''''''''
Dim Ctrl As Office.CommandBarControl

On Error Resume Next
Set Ctrl = Application.CommandBars.FindControl(Tag:=C_TAG)

Do Until Ctrl Is Nothing
    Ctrl.Delete
    Set Ctrl = Application.CommandBars.FindControl(Tag:=C_TAG)
Loop

End Sub

'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
' END ThisWorkbook Code Module
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

Appendix B: Remaining Bootstrapped Data for Original DFK Process

This appendix presents the remaining bootstrapped data from Chapter 4. The bootstrap replicates 1 and 25 are displayed in Chapter 4. These replicates were omitted in the body of the paper as the two displayed replicates are enough for demonstration of the mathematical method involved in performing the bootstrapped calculations.

Bootstrap Replicate 2										$\bar{z}^*_{i:2}$
0.067	0.097	0.161	0.091	0.091	0.097	0.091	0.091	0.125	0.161	0.107
0.257	0.281	0.292	0.203	0.659	0.203	0.659	0.290	0.203	0.292	0.334
0.322	0.805	0.805	0.351	0.193	0.805	0.193	0.132	0.264	0.351	0.422
0.216	0.412	0.216	0.268	0.392	0.392	0.392	0.403	0.412	0.403	0.350
0.188	0.976	0.363	0.421	0.421	0.421	0.219	1.100	0.421	0.421	0.495

Bootstrap Replicate 3										$\bar{z}^*_{i:3}$
0.125	0.125	0.097	0.091	0.091	0.091	0.091	0.125	0.125	0.091	0.105
0.292	0.257	0.281	0.257	0.292	0.292	0.203	0.203	0.290	0.257	0.262
0.193	0.264	0.322	0.322	0.351	0.193	0.805	0.264	0.264	0.805	0.378
0.412	0.268	0.216	0.392	0.268	0.268	0.412	0.268	0.268	0.216	0.299
0.363	0.188	0.976	0.188	0.976	0.363	0.188	0.188	0.219	0.976	0.462

Bootstrap Replicate 4										$\bar{z}^*_{i:4}$
0.067	0.091	0.125	0.067	0.097	0.091	0.161	0.125	0.125	0.067	0.101
0.257	0.281	0.659	0.281	0.292	0.659	0.281	0.257	0.257	0.203	0.343
0.805	0.193	0.264	0.351	0.193	0.132	0.805	0.351	0.805	0.193	0.409
0.392	0.392	0.268	0.553	0.216	0.553	0.392	0.392	0.403	0.268	0.383
0.421	0.219	0.219	0.421	0.219	1.100	1.100	0.363	0.219	1.100	0.538

Bootstrap Replicate 5										$\bar{z}^*_{i:5}$
0.125	0.161	0.125	0.161	0.161	0.097	0.097	0.091	0.067	0.097	0.118
0.257	0.203	0.659	0.257	0.203	0.292	0.292	0.290	0.290	0.257	0.300
0.193	0.805	0.193	0.351	0.351	0.322	0.351	0.805	0.193	0.322	0.389
0.412	0.412	0.392	0.216	0.553	0.403	0.553	0.216	0.412	0.392	0.396
0.219	0.363	0.421	0.976	1.100	0.421	0.188	0.976	0.976	0.976	0.661

Bootstrap Replicate 6										$\bar{z}^*_{i:6}$
0.125	0.091	0.091	0.067	0.067	0.125	0.091	0.091	0.067	0.125	0.094
0.281	0.203	0.257	0.203	0.290	0.281	0.290	0.203	0.292	0.203	0.250
0.193	0.322	0.322	0.805	0.132	0.193	0.322	0.322	0.351	0.193	0.316
0.403	0.553	0.268	0.392	0.392	0.392	0.268	0.553	0.412	0.392	0.402
0.363	0.363	1.100	0.421	1.100	0.363	0.976	0.363	0.219	0.219	0.549

Bootstrap Replicate 7										$\bar{z}^*_{i:7}$
0.067	0.125	0.067	0.067	0.091	0.091	0.091	0.091	0.091	0.161	0.094
0.659	0.292	0.257	0.203	0.292	0.203	0.290	0.292	0.281	0.292	0.306
0.193	0.351	0.322	0.322	0.805	0.193	0.264	0.193	0.132	0.351	0.313
0.392	0.553	0.412	0.216	0.412	0.268	0.392	0.412	0.216	0.553	0.382
1.100	0.188	1.100	0.363	0.976	0.421	0.421	0.363	0.421	0.421	0.577

Bootstrap Replicate 8										$\bar{z}^*_{i:8}$
0.161	0.161	0.097	0.067	0.091	0.091	0.067	0.091	0.161	0.125	0.111
0.290	0.659	0.281	0.257	0.203	0.203	0.292	0.281	0.659	0.292	0.342
0.322	0.132	0.351	0.132	0.193	0.351	0.351	0.264	0.132	0.805	0.303
0.268	0.403	0.216	0.412	0.392	0.268	0.216	0.268	0.412	0.392	0.325
0.219	1.100	0.363	1.100	1.100	0.219	0.188	0.976	0.421	1.100	0.679

Bootstrap Replicate 9										$\bar{z}^*_{i:9}$
0.125	0.091	0.091	0.161	0.161	0.125	0.067	0.091	0.091	0.091	0.109
0.203	0.290	0.203	0.257	0.257	0.290	0.292	0.257	0.257	0.257	0.256
0.351	0.322	0.322	0.351	0.322	0.264	0.132	0.132	0.132	0.264	0.259
0.553	0.268	0.403	0.553	0.392	0.403	0.403	0.553	0.268	0.403	0.420
0.188	0.188	0.188	0.363	1.100	0.421	0.976	1.100	0.421	0.363	0.531

Bootstrap Replicate 10										$\bar{z}^*_{i:10}$
0.161	0.097	0.067	0.067	0.091	0.161	0.125	0.091	0.091	0.097	0.105
0.281	0.281	0.203	0.290	0.281	0.281	0.292	0.281	0.257	0.257	0.270
0.351	0.351	0.351	0.805	0.264	0.132	0.322	0.264	0.805	0.264	0.391
0.403	0.412	0.392	0.412	0.412	0.392	0.403	0.268	0.553	0.412	0.406
0.219	0.976	0.188	0.188	1.100	0.219	0.421	0.363	0.188	1.100	0.496

Bootstrap Replicate 11										$\bar{z}^*_{i:11}$
0.125	0.161	0.067	0.067	0.097	0.067	0.097	0.161	0.125	0.097	0.106
0.659	0.290	0.290	0.257	0.292	0.281	0.281	0.290	0.290	0.290	0.322
0.805	0.351	0.264	0.193	0.132	0.132	0.351	0.264	0.322	0.351	0.317
0.412	0.403	0.268	0.392	0.412	0.403	0.216	0.268	0.216	0.216	0.320
0.219	0.976	0.188	1.100	0.976	0.188	1.100	0.421	0.188	0.219	0.557

Bootstrap Replicate 12										$\bar{z}^*_{i:12}$
0.067	0.097	0.091	0.091	0.091	0.097	0.091	0.125	0.125	0.091	0.097
0.290	0.290	0.659	0.290	0.257	0.292	0.203	0.290	0.281	0.257	0.311
0.351	0.132	0.132	0.351	0.805	0.193	0.322	0.264	0.351	0.322	0.322
0.392	0.403	0.403	0.392	0.553	0.412	0.403	0.216	0.553	0.412	0.414
0.976	0.976	0.188	0.363	0.188	0.976	0.188	1.100	0.188	0.363	0.550

Bootstrap Replicate 13										$\bar{z}^*_{i:13}$
0.097	0.097	0.161	0.067	0.067	0.091	0.091	0.097	0.125	0.091	0.098
0.659	0.659	0.281	0.292	0.290	0.659	0.281	0.290	0.292	0.281	0.399
0.193	0.132	0.132	0.351	0.351	0.193	0.132	0.351	0.193	0.193	0.222
0.553	0.412	0.403	0.268	0.553	0.216	0.216	0.553	0.403	0.268	0.384
1.100	0.421	0.363	0.976	0.976	1.100	0.363	0.363	0.363	0.976	0.700

Bootstrap Replicate 14										$\bar{z}^*_{i:14}$
0.067	0.091	0.097	0.091	0.125	0.091	0.125	0.097	0.091	0.091	0.097
0.281	0.281	0.257	0.659	0.257	0.281	0.290	0.281	0.659	0.659	0.391
0.805	0.193	0.132	0.351	0.322	0.264	0.351	0.322	0.264	0.805	0.381
0.268	0.412	0.412	0.392	0.216	0.268	0.392	0.268	0.216	0.268	0.311
1.100	0.363	0.188	0.219	0.363	0.219	0.188	0.188	0.976	0.363	0.417

Bootstrap Replicate 15										$\bar{z}^*_{i:15}$
0.067	0.067	0.091	0.097	0.097	0.091	0.091	0.161	0.091	0.091	0.094
0.659	0.290	0.290	0.292	0.203	0.292	0.203	0.203	0.203	0.203	0.284
0.264	0.193	0.193	0.193	0.132	0.351	0.132	0.351	0.132	0.351	0.229
0.392	0.403	0.553	0.403	0.553	0.553	0.412	0.403	0.553	0.216	0.444
0.219	0.976	0.219	0.976	0.976	0.421	0.976	1.100	0.363	1.100	0.733

Bootstrap Replicate 16										$\bar{z}^*_{i:16}$
0.125	0.097	0.067	0.125	0.125	0.091	0.125	0.091	0.161	0.125	0.113
0.203	0.281	0.292	0.659	0.292	0.290	0.290	0.292	0.203	0.290	0.309
0.805	0.264	0.264	0.351	0.193	0.351	0.132	0.264	0.132	0.805	0.356
0.412	0.412	0.392	0.268	0.403	0.216	0.553	0.553	0.412	0.403	0.402
0.188	0.976	0.421	0.219	0.976	0.421	0.219	0.976	1.100	0.363	0.586

Bootstrap Replicate 17										$\bar{z}^*_{i:17}$
0.097	0.097	0.091	0.161	0.067	0.091	0.125	0.161	0.091	0.161	0.114
0.290	0.203	0.292	0.203	0.257	0.257	0.257	0.257	0.257	0.257	0.253
0.193	0.264	0.132	0.322	0.132	0.193	0.264	0.264	0.351	0.132	0.225
0.268	0.553	0.268	0.412	0.216	0.412	0.412	0.412	0.392	0.553	0.390
0.363	0.363	0.976	0.188	0.363	0.421	0.188	0.421	0.976	0.219	0.448

Bootstrap Replicate 18										$\bar{z}^*_{i:18}$
0.091	0.067	0.125	0.091	0.067	0.097	0.067	0.091	0.067	0.091	0.085
0.292	0.203	0.659	0.257	0.257	0.292	0.203	0.281	0.659	0.257	0.336
0.132	0.351	0.351	0.132	0.805	0.351	0.351	0.351	0.805	0.805	0.444
0.392	0.392	0.268	0.392	0.216	0.268	0.268	0.412	0.553	0.403	0.356
0.188	0.219	1.100	0.219	0.188	0.188	1.100	0.976	0.363	0.219	0.476

Bootstrap Replicate 19										$\bar{z}^*_{i:19}$
0.097	0.067	0.067	0.091	0.097	0.091	0.097	0.091	0.125	0.161	0.098
0.203	0.257	0.659	0.257	0.290	0.290	0.659	0.659	0.292	0.292	0.386
0.351	0.322	0.322	0.322	0.351	0.132	0.351	0.193	0.805	0.805	0.396
0.412	0.392	0.392	0.392	0.553	0.216	0.553	0.216	0.392	0.412	0.393
0.976	0.363	0.976	0.188	0.188	0.219	0.421	0.421	0.219	0.976	0.495

Bootstrap Replicate 20										$\bar{z}^*_{i:20}$
0.161	0.097	0.091	0.091	0.067	0.097	0.097	0.125	0.067	0.097	0.099
0.257	0.292	0.281	0.292	0.281	0.203	0.203	0.257	0.203	0.292	0.256
0.351	0.805	0.193	0.132	0.132	0.193	0.805	0.322	0.351	0.264	0.355
0.412	0.216	0.403	0.412	0.412	0.216	0.268	0.412	0.412	0.392	0.355
0.976	0.188	0.421	0.219	0.219	0.976	1.100	0.976	0.976	0.976	0.703

Bootstrap Replicate 21										$\bar{z}^*_{i:21}$
0.161	0.091	0.097	0.091	0.097	0.125	0.161	0.097	0.091	0.067	0.108
0.203	0.281	0.659	0.290	0.257	0.292	0.257	0.257	0.257	0.257	0.301
0.264	0.322	0.193	0.264	0.351	0.193	0.322	0.351	0.193	0.264	0.272
0.216	0.403	0.553	0.403	0.412	0.392	0.553	0.268	0.553	0.412	0.416
0.363	0.188	0.421	0.363	0.363	0.976	1.100	0.421	1.100	0.188	0.548

Bootstrap Replicate 22										$\bar{z}^*_{i:22}$
0.067	0.091	0.125	0.091	0.067	0.067	0.091	0.091	0.091	0.125	0.091
0.257	0.203	0.659	0.203	0.659	0.290	0.257	0.257	0.290	0.203	0.328
0.193	0.322	0.351	0.351	0.193	0.351	0.351	0.351	0.351	0.322	0.314
0.403	0.392	0.412	0.553	0.403	0.216	0.392	0.216	0.412	0.403	0.380
0.976	0.219	0.188	0.421	0.421	0.976	0.363	0.363	0.219	0.421	0.457

Bootstrap Replicate 23										$\bar{z}^*_{i:23}$
0.125	0.097	0.067	0.091	0.091	0.097	0.067	0.125	0.067	0.161	0.099
0.203	0.281	0.290	0.281	0.292	0.290	0.292	0.292	0.257	0.659	0.314
0.264	0.193	0.264	0.351	0.351	0.193	0.322	0.351	0.132	0.132	0.255
0.403	0.403	0.403	0.216	0.268	0.553	0.392	0.403	0.392	0.392	0.382
0.976	0.421	0.363	0.188	0.188	0.188	0.976	0.363	0.188	0.188	0.404

Bootstrap Replicate 24										$\bar{z}^*_{i:24}$
0.161	0.067	0.125	0.091	0.097	0.091	0.091	0.091	0.091	0.091	0.100
0.203	0.203	0.257	0.659	0.290	0.257	0.257	0.292	0.203	0.292	0.291
0.264	0.193	0.132	0.805	0.264	0.351	0.351	0.193	0.193	0.351	0.310
0.553	0.268	0.268	0.268	0.412	0.392	0.216	0.392	0.412	0.412	0.359
0.219	0.976	1.100	0.363	0.188	0.188	1.100	0.421	0.421	1.100	0.608

Appendix C: Varying Sill and Nugget Variogram Tables and Plots

This appendix further demonstrates the relationships in fitted variogram functions. The appendix is broken up into two sections. The first section demonstrates how the variogram functions vary when the sill varies up to the maximum lag. The results are visually presented through four plots at the end of the first section. The second section demonstrates the same as the first section with the exception that the nugget is varied while the range and sill remain constant.

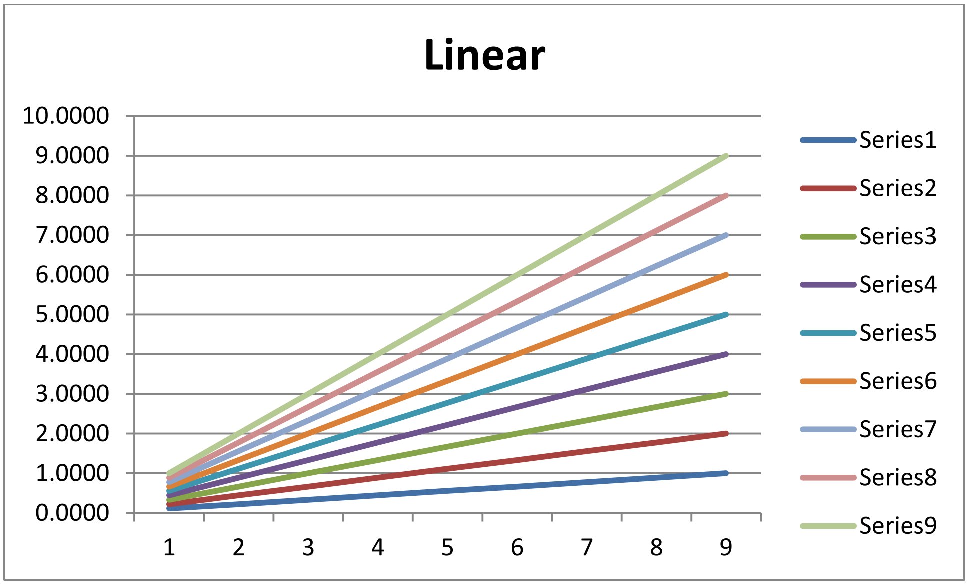
Appendix C.1: Varying Sill Variogram Tables and Plots

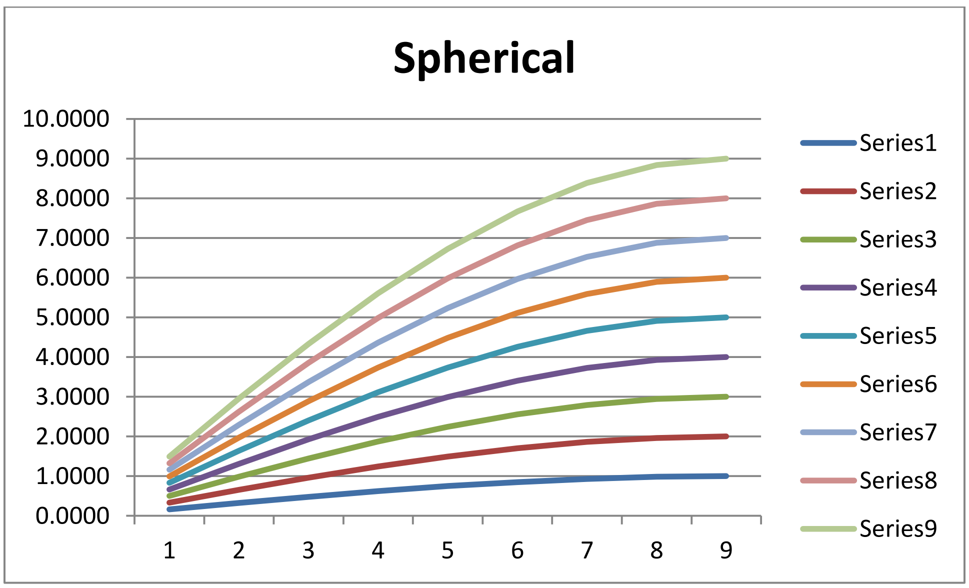
c_1	a	Linear $\{h: h = 1,2, \ldots, 9\}$								
1	9	0.1111	0.2222	0.3333	0.4444	0.5556	0.6667	0.7778	0.8889	1.0000
2	9	0.2222	0.4444	0.6667	0.8889	1.1111	1.3333	1.5556	1.7778	2.0000
3	9	0.3333	0.6667	1.0000	1.3333	1.6667	2.0000	2.3333	2.6667	3.0000
4	9	0.4444	0.8889	1.3333	1.7778	2.2222	2.6667	3.1111	3.5556	4.0000
5	9	0.5556	1.1111	1.6667	2.2222	2.7778	3.3333	3.8889	4.4444	5.0000
6	9	0.6667	1.3333	2.0000	2.6667	3.3333	4.0000	4.6667	5.3333	6.0000
7	9	0.7778	1.5556	2.3333	3.1111	3.8889	4.6667	5.4444	6.2222	7.0000
8	9	0.8889	1.7778	2.6667	3.5556	4.4444	5.3333	6.2222	7.1111	8.0000
9	9	1.0000	2.0000	3.0000	4.0000	5.0000	6.0000	7.0000	8.0000	9.0000

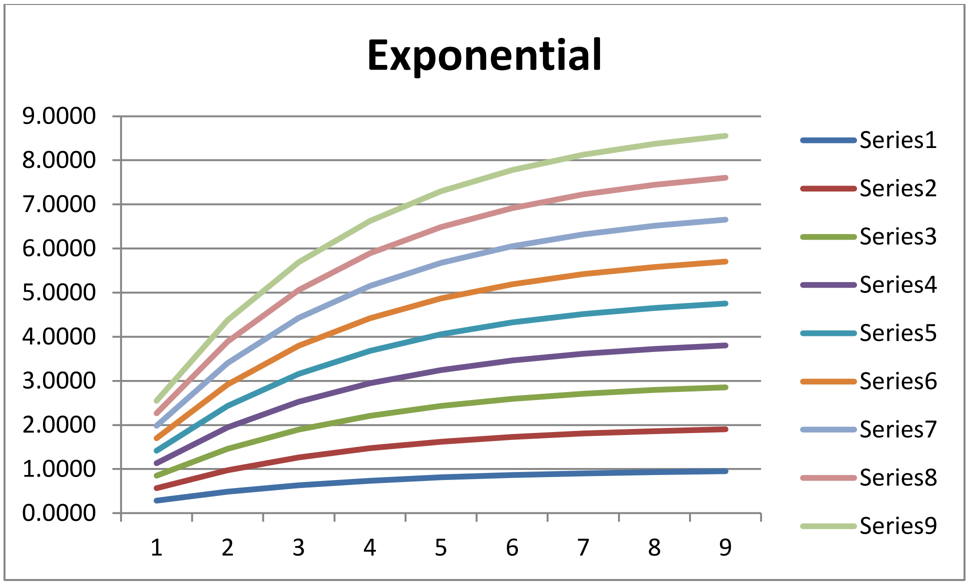
c_1	a	Spherical $\{h: h = 1,2, \dots, 9\}$								
1	9	0.1660	0.3278	0.4815	0.6228	0.7476	0.8519	0.9314	0.9822	0.1660
2	9	0.3320	0.6557	0.9630	1.2455	1.4952	1.7037	1.8628	1.9643	0.3320
3	9	0.4979	0.9835	1.4444	1.8683	2.2428	2.5556	2.7942	2.9465	0.4979
4	9	0.6639	1.3114	1.9259	2.4911	2.9904	3.4074	3.7257	3.9287	0.6639
5	9	0.8299	1.6392	2.4074	3.1139	3.7380	4.2593	4.6571	4.9108	0.8299
6	9	0.9959	1.9671	2.8889	3.7366	4.4856	5.1111	5.5885	5.8930	0.9959
7	9	1.1619	2.2949	3.3704	4.3594	5.2332	5.9630	6.5199	6.8752	1.1619
8	9	1.3278	2.6228	3.8519	4.9822	5.9808	6.8148	7.4513	7.8573	1.3278
9	9	1.4938	2.9506	4.3333	5.6049	6.7284	7.6667	8.3827	8.8395	1.4938

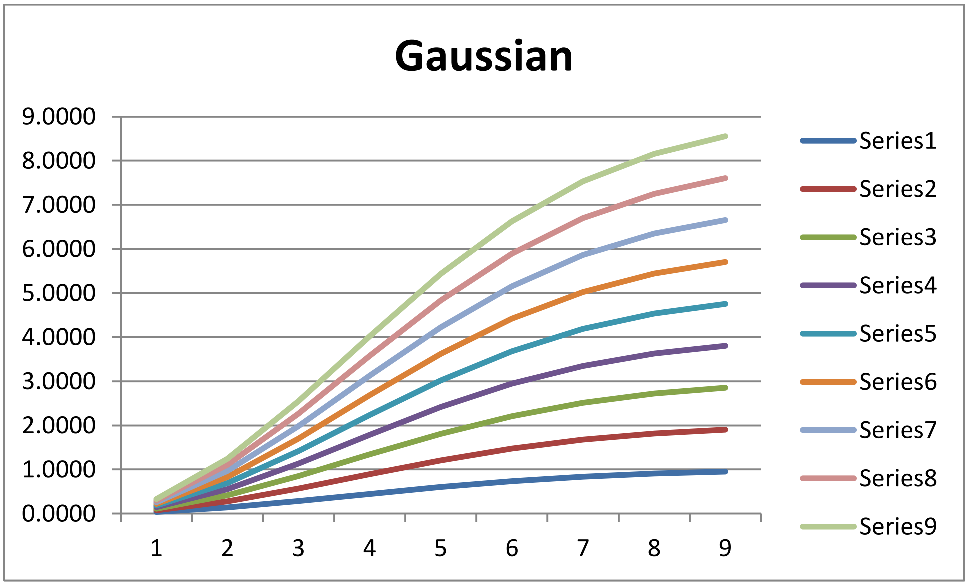
c_1	a	Exponential $\{h: h = 1,2, \dots, 9\}$								
1	9	0.2835	0.4866	0.6321	0.7364	0.8111	0.8647	0.9030	0.9305	0.9502
2	9	0.5669	0.9732	1.2642	1.4728	1.6222	1.7293	1.8061	1.8610	1.9004
3	9	0.8504	1.4597	1.8964	2.2092	2.4334	2.5940	2.7091	2.7915	2.8506
4	9	1.1339	1.9463	2.5285	2.9456	3.2445	3.4587	3.6121	3.7221	3.8009
5	9	1.4173	2.4329	3.1606	3.6820	4.0556	4.3233	4.5151	4.6526	4.7511
6	9	1.7008	2.9195	3.7927	4.4184	4.8667	5.1880	5.4182	5.5831	5.7013
7	9	1.9843	3.4061	4.4248	5.1548	5.6779	6.0527	6.3212	6.5136	6.6515
8	9	2.2677	3.8927	5.0570	5.8912	6.4890	6.9173	7.2242	7.4441	7.6017
9	9	2.5512	4.3792	5.6891	6.6276	7.3001	7.7820	8.1273	8.3746	8.5519

c_1	a	Gaussian $\{h: h = 1,2, \dots, 9\}$								
1	9	0.950213	0.999994	0.0364	0.1377	0.2835	0.4471	0.6038	0.7364	0.8371
2	9	1.900426	1.999988	0.0727	0.2754	0.5669	0.8942	1.2077	1.4728	1.6743
3	9	2.850639	2.999982	0.1091	0.4131	0.8504	1.3413	1.8115	2.2092	2.5114
4	9	3.800852	3.999975	0.1454	0.5508	1.1339	1.7884	2.4153	2.9456	3.3485
5	9	4.751065	4.999969	0.1818	0.6885	1.4173	2.2355	3.0192	3.6820	4.1857
6	9	5.701278	5.999963	0.2182	0.8262	1.7008	2.6826	3.6230	4.4184	5.0228
7	9	6.651491	6.999957	0.2545	0.9639	1.9843	3.1298	4.2268	5.1548	5.8599
8	9	7.601703	7.999951	0.2909	1.1016	2.2677	3.5769	4.8307	5.8912	6.6971
9	9	8.551916	8.999945	0.3272	1.2393	2.5512	4.0240	5.4345	6.6276	7.5342


Linear
10.0000
9.0000
8.0000
7.0000
6.0000
5.0000
4.0000
3.0000
2.0000
1.0000
0.0000
1 2 3 4 5 6 7 8 9
Series1
Series2
Series3
Series4
Series5
Series6
Series7
Series8
Series9


Spherical
10.0000
9.0000
8.0000
7.0000
6.0000
5.0000
4.0000
3.0000
2.0000
1.0000
0.0000
1 2 3 4 5 6 7 8 9
Series1
Series2
Series3
Series4
Series5
Series6
Series7
Series8
Series9


Exponential
9.0000
8.0000
7.0000
6.0000
5.0000
4.0000
3.0000
2.0000
1.0000
0.0000
1
2
3
4
5
6
7
8
9
Series1
Series2
Series3
Series4
Series5
Series6
Series7
Series8
Series9


Gaussian
9.0000
8.0000
7.0000
6.0000
5.0000
4.0000
3.0000
2.0000
1.0000
0.0000
1
2
3
4
5
6
7
8
9
Series1
Series2
Series3
Series4
Series5
Series6
Series7
Series8
Series9

Appendix C.2: Varying Nugget Variogram Tables and Plots

Nugget = 1

c_1	a	Spherical $\{h: h = 1,2,\ldots,9\}$								
9	1	10.0000	10.0000	10.0000	10.0000	10.0000	10.0000	10.000	10.000	10.000
9	2	7.1875	10.0000	10.0000	10.0000	10.0000	10.0000	10.000	10.000	10.000
9	3	5.3333	8.6667	10.0000	10.0000	10.0000	10.0000	10.000	10.000	10.000
9	4	4.3047	7.1875	9.2266	10.0000	10.0000	10.0000	10.000	10.000	10.000
9	5	3.6640	6.1120	8.1280	9.4960	10.0000	10.0000	10.000	10.000	10.000
9	6	3.2292	5.3333	7.1875	8.6667	9.6458	10.0000	10.000	10.000	10.000
9	7	2.9155	4.7522	6.4315	7.8746	9.0029	9.7376	10.000	10.000	10.000
9	8	2.6787	4.3047	5.8252	7.1875	8.3389	9.2266	9.798	10.000	10.000
9	9	2.4938	3.9506	5.3333	6.6049	7.7284	8.6667	9.383	9.84	10.000

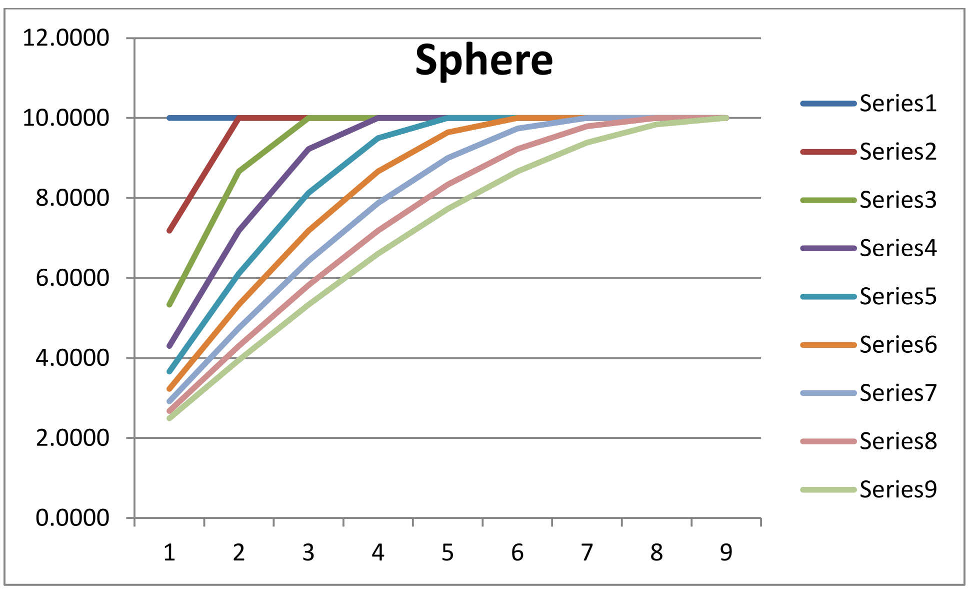
Nugget = 2

c_1	a	Spherical $\{h: h = 1,2,\ldots,9\}$								
9	1	11.0000	11.0000	11.0000	11.0000	11.0000	11.0000	11.000	11.000	11.000
9	2	8.1875	11.0000	11.0000	11.0000	11.0000	11.0000	11.000	11.000	11.000
9	3	6.3333	9.6667	11.0000	11.0000	11.0000	11.0000	11.000	11.000	11.000
9	4	5.3047	8.1875	10.2266	11.0000	11.0000	11.0000	11.000	11.000	11.000
9	5	4.6640	7.1120	9.1280	10.4960	11.0000	11.0000	11.000	11.000	11.000
9	6	4.2292	6.3333	8.1875	9.6667	10.6458	11.0000	11.000	11.000	11.000
9	7	3.9155	5.7522	7.4315	8.8746	10.0029	10.7376	11.000	11.000	11.000
9	8	3.6787	5.3047	6.8252	8.1875	9.3389	10.2266	10.798	11.000	11.000
9	9	3.4938	4.9506	6.3333	7.6049	8.7284	9.6667	10.383	10.84	11.000

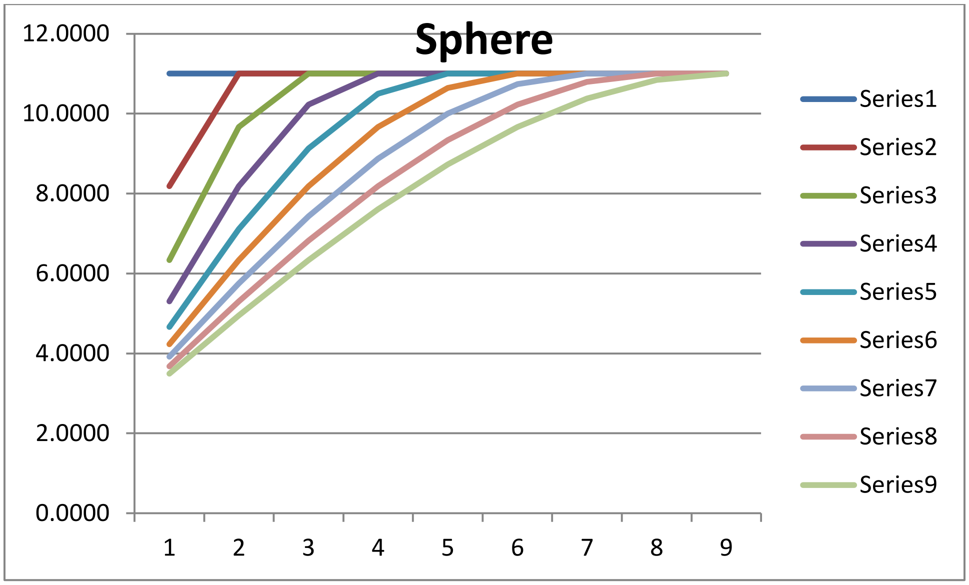
Nugget = 3

c_1	a	Spherical $\{h: h = 1,2,\dots,9\}$								
9	1	12.000	12.000	12.0000	12.0000	12.0000	12.0000	12.0000	12.0000	12.000
9	2	9.188	12.000	12.0000	12.0000	12.0000	12.0000	12.0000	12.0000	12.000
9	3	7.333	10.667	12.0000	12.0000	12.0000	12.0000	12.0000	12.0000	12.000
9	4	6.305	9.1875	11.2266	12.0000	12.0000	12.0000	12.0000	12.0000	12.000
9	5	5.664	8.1120	10.1280	11.4960	12.0000	12.0000	12.0000	12.0000	12.000
9	6	5.229	7.3333	9.1875	10.6667	11.6458	12.0000	12.0000	12.0000	12.000
9	7	4.916	6.7522	8.4315	9.8746	11.0029	11.7376	12.0000	12.0000	12.000
9	8	4.679	6.3047	7.8252	9.1875	10.3389	11.2266	11.7979	12.0000	12.000
9	9	4.494	5.9506	7.3333	8.6049	9.7284	10.6667	11.3827	11.8395	12.000

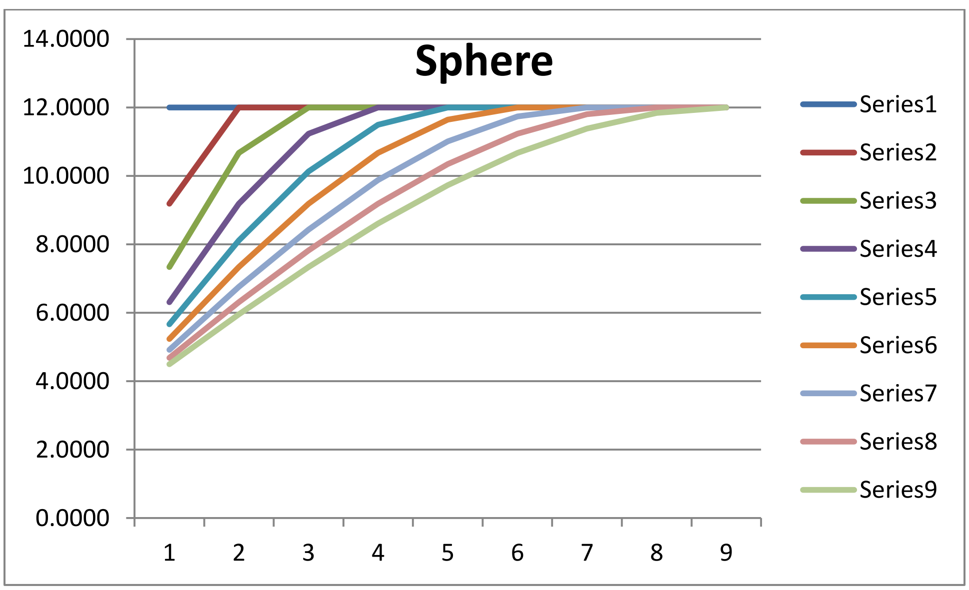
Nugget = 1



Nugget = 2



Nugget = 3



Appendix D: Supplemental Data from Application Area 2: Sample Size Selection

Table 50: Pilot Design Estimated Response Data

		Predicted Response		
	Candidate FLC	Smooth Polynomial Curve	Rough Polynomial Curve	Noisy Curve
D= 1; Total FLC= 25; Variance Reduction= 25%	5	0.47150	3.21660	51.74491
	23	2.02550	7.38003	52.72585
D= 1; Total FLC= 25; Variance Reduction= 33%	4	0.37850	2.83269	51.67268
	18	1.63550	6.90621	50.77602
D= 1; Total FLC= 25; Variance Reduction= 50%	8	0.76519	4.82349	49.85692
	21	1.87841	7.18280	50.63585
D= 1; Total FLC= 25; Variance Reduction= 66%	3	0.29500	3.94554	51.22746
	24	2.11100	7.45919	52.11113
D= 1; Total FLC= 25; Variance Reduction= 75%	6	0.58378	4.37253	49.81804
	18	1.64113	6.91673	51.04040

D= 1; Total FLC= 25; Variance Reduction= 90%	6	0.58150	4.46724	49.81804
	17	1.55500	6.69211	51.58242
D= 2; Total FLC= 25; Variance Reduction= 25%	(1, 2)	0.22933	2.87158	51.46236
	(4, 1)	1.47455	5.99412	50.84575
D= 2; Total FLC= 25; Variance Reduction= 33%	(2, 2)	0.62389	4.19965	51.65355
	(4, 3)	1.66150	6.74387	51.88809
D= 2; Total FLC= 25; Variance Reduction= 50%	(1, 4)	0.39688	5.99906	50.95169
	(5, 2)	1.96280	9.21019	51.06570
D= 2; Total FLC= 25; Variance Reduction= 66%	(2, 1)	0.55963	3.62093	50.67789
	(4, 5)	1.74794	6.76693	52.17126
D= 2; Total FLC= 25; Variance Reduction= 75%	(2, 3)	0.76288	5.15992	50.64519
	(3, 4)	1.30025	6.36026	51.17188
D= 2; Total FLC= 25; Variance Reduction= 90%	(3, 2)	1.12307	5.89098	50.98575
	(3, 4)	1.30473	6.38044	51.02156

D= 1; Total FLC= 125; Variance Reduction= 25%	6	0.51700	2.77739	51.37124
	93	4.96050	12.07827	49.81512
D= 1; Total FLC= 125; Variance Reduction= 33%	12	1.10642	5.24324	51.99270
	120	4.77128	15.84691	47.63722
D= 1; Total FLC= 125; Variance Reduction= 50%	12	1.10883	5.24395	51.99270
	120	4.79362	14.77820	47.67108
D= 1; Total FLC= 125; Variance Reduction= 66%	25	2.18600	7.71890	52.38430
	90	4.94400	11.92503	50.75688
D= 1; Total FLC= 125; Variance Reduction= 75%	21	1.87900	7.12575	50.46867
	80	4.79850	11.21554	51.80544
D= 1; Total FLC= 125; Variance Reduction= 90%	23	2.03500	7.38626	51.42548
	70	4.54950	10.61942	50.93005
D= 3; Total FLC= 125; Variance Reduction= 25%	(1, 1, 2)	0.21768	2.76975	52.00449
	(4, 4, 1)	4.85602	11.75360	49.76547

D= 3; Total FLC= 125; Variance Reduction= 33%	(2, 3, 2)	2.85392	9.40406	52.67096
	(5, 1, 2)	5.01229	13.19267	50.58011
D= 3; Total FLC= 125; Variance Reduction= 50%	(1, 1, 3)	0.44948	1.63525	50.83818
	(4, 5, 5)	4.80432	12.72027	51.94059
D= 3; Total FLC= 125; Variance Reduction= 66%	(2, 1, 2)	2.24137	7.15881	52.95494
	(3, 5, 4)	4.64795	10.65394	54.03372
D= 3; Total FLC= 125; Variance Reduction= 75%	(1, 2, 4)	0.88769	5.71143	50.69707
	(3, 4, 5)	4.53024	10.05423	52.91640
D= 3; Total FLC= 125; Variance Reduction= 90%	(2, 2, 3)	2.69963	8.12533	51.58729
	(4, 4, 2)	4.93471	12.30718	51.02052

Table 51: Pilot Design Estimated Variance Data

		Variance	
	Candidate FLC	Smooth Polynomial Curve	Rough Polynomial Curve

D= 1; Total FLC= 25; Variance Reduction= 25%	5	0.43620	3.62242
	23	0.27263	2.26401
D= 1; Total FLC= 25; Variance Reduction= 33%	4	0.34674	3.11598
	18	0.12842	1.15407
D= 1; Total FLC= 25; Variance Reduction= 50%	8	0.12027	0.18242
	21	0.09622	0.15382
D= 1; Total FLC= 25; Variance Reduction= 66%	3	0.07175	0.14951
	24	0.09335	0.15862
D= 1; Total FLC= 25; Variance Reduction= 75%	6	0.07001	0.12058
	18	0.00579	0.15862
D= 1; Total FLC= 25; Variance Reduction= 90%	6	0.06965	0.08869
	17	0.06965	0.08790
D= 2; Total FLC= 25; Variance Reduction= 25%	(1, 2)	0.25801	0.27119
	(4, 1)	0.36180	0.38278

D= 2; Total FLC= 25; Variance Reduction= 33%	(2, 2)	0.30082	0.07008
	(4, 3)	0.23605	0.05504
D= 2; Total FLC= 25; Variance Reduction= 50%	(1, 4)	0.01027	0.00445
	(5, 2)	0.00932	0.00144
D= 2; Total FLC= 25; Variance Reduction= 66%	(2, 1)	0.05350	0.29321
	(4, 5)	0.05430	0.30462
D= 2; Total FLC= 25; Variance Reduction= 75%	(2, 3)	0.00503	0.31039
	(3, 4)	0.00503	0.31050
D= 2; Total FLC= 25; Variance Reduction= 90%	(3, 2)	0.16216	0.25450
	(3, 4)	0.16216	0.25450
D= 1; Total FLC= 125; Variance Reduction= 25%	6	0.64876	6.08711
	93	0.14971	1.40472
D= 1; Total FLC= 125; Variance Reduction= 33%	12	0.17835	3.14344
	120	0.20262	3.57136

D= 1; Total FLC= 125; Variance Reduction= 50%	12	0.33445	3.14344
	120	0.16317	3.57136
D= 1; Total FLC= 125; Variance Reduction= 66%	25	0.03214	0.58621
	90	0.07347	1.33942
D= 1; Total FLC= 125; Variance Reduction= 75%	21	0.07341	0.40646
	80	0.11011	0.60969
D= 1; Total FLC= 125; Variance Reduction= 90%	23	0.07204	0.38693
	70	0.07204	0.38693
D= 3; Total FLC= 125; Variance Reduction= 25%	(1, 1, 2)	0.00000	0.00000
	(4, 4, 1)	0.00000	0.00000
D= 3; Total FLC= 125; Variance Reduction= 33%	(2, 3, 2)	0.06866	1.05015
	(5, 1, 2)	0.05556	0.68909
D= 3; Total FLC= 125; Variance Reduction= 50%	(1, 1, 3)	0.22836	0.77976
	(4, 5, 5)	0.22917	0.76307

D= 3; Total FLC= 125; Variance Reduction= 66%	(2, 1, 2)	0.21954	0.42767
	(3, 5, 4)	0.20727	0.42269
D= 3; Total FLC= 125; Variance Reduction= 75%	(1, 2, 4)	0.23951	0.36160
	(3, 4, 5)	0.23935	0.37415
D= 3; Total FLC= 125; Variance Reduction= 90%	(2, 2, 3)	0.17403	0.38575
	(4, 4, 2)	0.16888	0.37328

Table 52: Pilot Design Squared Residual Data

		e_i^2 Values		
	Candidate FLC	Smooth Polynomial Curve	Rough Polynomial Curve	Noisy Curve
D= 1; Total FLC= 25; Variance Reduction= 25%	5	0.00026	1.13472	2.16849
	23	0.00010	0.02355	1.81440
D= 1; Total FLC= 25; Variance Reduction= 33%	4	0.00018	1.58424	0.63287
	18	0.00001	0.00485	0.04039

D= 1; Total FLC= 25; Variance Reduction= 50%	8	0.00001	0.00852	0.28235
	21	0.00000	0.00042	0.11028
D= 1; Total FLC= 25; Variance Reduction= 66%	3	0.00000	0.00139	0.01736
	24	0.00000	0.01940	0.00041
D= 1; Total FLC= 25; Variance Reduction= 75%	6	0.00000	0.00081	0.01059
	18	0.00001	0.00349	0.00402
D= 1; Total FLC= 25; Variance Reduction= 90%	6	0.00000	0.01516	0.01059
	17	0.00000	0.04309	0.00063
D= 2; Total FLC= 25; Variance Reduction= 25%	(1, 2)	0.00098	0.87603	0.01332
	(4, 1)	0.00001	0.28553	1.80126
D= 2; Total FLC= 25; Variance Reduction= 33%	(2, 2)	0.00266	0.06740	5.24315
	(4, 3)	0.00055	0.05380	0.83010
D= 2; Total FLC= 25; Variance Reduction= 50%	(1, 4)	0.00002	3.63934	0.00556
	(5, 2)	0.00002	4.14586	0.11921

D= 2; Total FLC= 25; Variance Reduction= 66%	(2, 1)	0.00050	0.52297	0.92691
	(4, 5)	0.00271	0.09642	3.81951
D= 2; Total FLC= 25; Variance Reduction= 75%	(2, 3)	0.00003	0.18380	1.74144
	(3, 4)	0.00000	0.00288	2.08227
D= 2; Total FLC= 25; Variance Reduction= 90%	(3, 2)	0.00002	0.10416	0.56401
	(3, 4)	0.00001	0.00546	2.53870
D= 1; Total FLC= 125; Variance Reduction= 25%	6	0.00423	2.45459	2.74270
	93	0.00022	0.00402	0.37023
D= 1; Total FLC= 125; Variance Reduction= 33%	12	0.00047	0.94183	0.06551
	120	0.00082	2.85139	1.67559
D= 1; Total FLC= 125; Variance Reduction= 50%	12	0.00037	0.94047	0.06551
	120	0.00004	0.38427	1.76440
D= 1; Total FLC= 125; Variance Reduction= 66%	25	0.00000	0.01450	0.17824
	90	0.00004	0.01253	2.30840

D= 1; Total FLC= 125; Variance Reduction= 75%	21	0.00000	0.00133	0.02719
	80	0.00000	0.00101	0.01620
D= 1; Total FLC= 125; Variance Reduction= 90%	23	0.00000	0.02550	0.00217
	70	0.00000	0.00009	0.02622
D= 3; Total FLC= 125; Variance Reduction= 25%	(1, 1, 2)	0.00039	1.07702	0.18208
	(4, 4, 1)	0.01071	0.08062	4.43182
D= 3; Total FLC= 125; Variance Reduction= 33%	(2, 3, 2)	0.02611	0.66917	0.06609
	(5, 1, 2)	0.00020	0.35083	0.39473
D= 3; Total FLC= 125; Variance Reduction= 50%	(1, 1, 3)	0.02371	5.16667	0.27149
	(4, 5, 5)	0.03829	0.05015	1.53263
D= 3; Total FLC= 125; Variance Reduction= 66%	(2, 1, 2)	0.00886	0.45425	0.17203
	(3, 5, 4)	0.00020	0.06160	8.79880
D= 3; Total FLC= 125; Variance Reduction= 75%	(1, 2, 4)	0.00079	0.85430	1.13720
	(3, 4, 5)	0.00039	0.30880	4.61510

D= 3; Total FLC= 125; Variance Reduction= 90%	(2, 2, 3)	0.00312	0.10331	0.62485
	(4, 4, 2)	0.00111	0.05932	0.03426

Appendix E: Supplemental Data from Application Area 2: $\boldsymbol{n_{ir}}$ and FLC Selection

Table 53: Application Area 2 Variance Reduction Data

$N = 25$	$d = 1$	Pilot Design n_{ir}	FLC Selection
Variance Reduction	25%	3	1, 13, 25
	33%	4	1, 13, 19, 25
	50%	7	1, 4, 7, 13, 19, 22, 25
	66%	10	1, 2, 4, 7, 10, 13, 16, 19, 22, 25
	75%	14	1, 2, 4, 5, 7, 8, 10, 13, 16, 19, 21, 22, 24, 25
	90%	20	1-5, 7, 8, 10, 11, 13, 15, 16, 18-25
$N = 25$	$d = 2$	Pilot Design n_{ir}	FLC Selection
Variance Reduction	25%	7	(1,1); (1,3); (1, 5); (3; 3); (5; 1); (5; 3); (5, 5)
	33%	8	(1,1); (1,3); (1, 5); (3; 3); (3, 5); (5, 1); (5, 3); (5, 5)
	50%	12	(1,1); (1, 3); (1, 5); (2,2); (3, 1); (3, 3); (3, 5); (4, 2); (4, 4); (5, 1); (5, 3); (5, 5)
	66%	16	(1,1); (1, 2); (1, 3); (1,5); (2, 2); (2, 4); (2, 5); (3,1); (3, 3); (3, 5); (4,1); (4, 2); (4, 4); (5, 1); (5, 3); (5, 5)
	75%	18	(1,1); (1, 2); (1, 3); (1, 4); (1, 5); (2, 2); (2, 4); (2, 5); (3,1); (3, 3); (3, 5); (4, 1); (4, 2); (4, 4); (5, 1); (5, 3); (5, 4); (5, 5)
	90%	23	(1,1); (1, 2); (1, 3); (1, 4); (1, 5); (2, 1); (2, 2); (2, 3); (2, 4); (2, 5); (3,1); (3, 3); (3, 5); (4, 1); (4, 2); (4,3); (4, 4); (4,5); (5, 1); (5, 2); (5, 3); (5, 4); (5, 5)
$N = 125$	$d = 1$	Pilot Design n_{ir}	FLC Selection
Variance Reduction	25%	5	1, 32, 63, 94, 125
	33%	7	1, 16, 32, 63, 94, 110, 125
	50%	14	1, 16, 24, 32, 47, 55, 63, 71, 79, 94, 102, 110, 118, 125

	66%	28	1, 8, 12, 16, 20, 24, 28, 32, 39, 43, 47, 51, 55, 63, 67, 71, 75, 79, 83, 87, 94, 98, 102, 106, 110, 114, 118, 125
	75%	43	1, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 28, 32, 35, 39, 43, 47, 51, 55, 59, 63, 67, 71, 75, 79, 83, 87, 91, 94, 98, 102, 104, 106, 108, 110, 112, 114, 116, 118, 120, 122, 125
	90%	84	1-22, 24, 26-28, 30, 32, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65, 67, 69, 71, 73, 75, 77, 79, 81, 83, 85, 87, 89, 91, 93, 94, 96, 98, 100, 102, 104, 106, 108-125
$N = 125$	$d = 3$	Pilot Design n_{ir}	FLC Selection
Variance Reduction	25%	27	(1, 1,1); (1,1,5); (1,3,1); (1,3,3); (1,3,5); (1,5,1); (1,5,3); (1,5,5); (3,1,1); (3,1,3);(3,1,5); (3,3,1); (3,3,3); (3,3,5); (3,5,3); (5,1,1); (5,1,5); (5, 2,2); (5,2,4); (5,3,1); (5,3,3); (5,3,5); (5,4,2); (5,4,4); (5,5,1); (5,5,3); (5,5,5)
	33%	34	(1,1,1); (1,1,2); (1,1,4); (1,1,5); (1,3,1); (1,3,3); (1,3,5); (1,5,1); (1,5,3); (1,5,5); (2,4,4); (2,5,1); (3,1,1); (3,1,3); (3,1,5); (3,3,1); (3, 3,3); (3,3,5); (3,5,3); (3,5, 5); (4,5,1); (5, 1,1); (5,1,3);(5,1,5); (5,2,2); (5,2,4); (5, 3,1); (5,3,3); (5,3,5); (5,4,2); (5,4,4); (5,5,1); (5,5,3); (5,5,5)
	50%	53	(1,1,1); (1,1,2); (1,1,4); (1,1,5); (1,3,1); (1,3,3); (1,3,5); (1,4,2); (1,5,1); (1,5, 3); (1,5,5); (2,2,1); (2,2,2); (2,2,4); (2,2,5); (2,4,2); (2,4,4); (2,4,5); (2,5,1); (2,5,4); (3,1,1); (3,1,3); (3,1,5); (3,3,1); (3,3,3); (3,3,5); (3,4,1); (3,5,2); (3,5,3); (3,5,5); (4,1,2); (4,1,4); (4,2,1); (4,2,3); (4,2, 5); (4,3,4); (4,4,3); (4,4,5); (4,5,1); (5,1,1); (5,1,3);(5,1,5); (5,2,2); (5,2,4); (5,3,1); (5,3,3); (5,3,5); (5,4,2); (5,4,4); (5,5,1); (5,5,3); (5,5,5)
	66%	75	(1, 1, 1); (1, 1, 2); (1, 1, 4); (1, 1, 5); (1, 2, 1); (1, 2, 3); (1, 2, 5); (1, 3, 1); (1, 3, 3); (1, 3, 5); (1, 4, 1); (1, 4, 2); (1, 4, 4); (1, 5, 1); (1, 5, 2); (1, 5, 3); (1, 5, 4); (1, 5, 5); (2, 1, 1); (2, 1, 3); (2, 1, 5); (2, 2, 1); (2, 2, 2); (2, 2, 4); (2, 2, 5);

			(2, 3, 3); (2, 4, 2); (2, 4, 4); (2, 4, 5); (2, 5, 1); (2, 5, 4); (2, 5, 5); (3, 1, 1); (3, 1, 3); (3, 1, 5); (3, 2, 2); (3, 2, 4); (3, 3, 1); (3, 3, 3); (3, 3, 5); (3, 4, 1); (3, 4, 4); (3, 5, 2); (3, 5, 3); (3, 5, 5); (4, 1, 1); (4, 1, 2); (4, 1, 4); (4, 2, 1); (4, 2, 3); (4, 2, 5); (4, 3, 2); (4, 3, 4); (4, 4, 1); (4, 4, 3); (4, 4, 5); (4, 5, 1); (4, 5, 2); (4, 5, 4); (4, 5, 5); (5, 1, 1); (5, 1, 3); (5, 1, 4); (5, 1, 5); (5, 2, 1); (5, 2, 2); (5, 2, 4); (5, 3, 1); (5, 3, 3); (5, 3, 5); (5, 4, 2); (5, 4, 4); (5, 5, 1); (5, 5, 3); (5, 5, 5)
	75%	87	(1, 1, 1); (1, 1, 2); (1, 1, 3); (1, 1, 4); (1, 1, 5); (1, 2, 1); (1, 2, 3); (1, 2, 5); (1, 3, 1); (1, 3, 2); (1, 3, 3); (1, 3, 4); (1, 3, 5); (1, 4, 1); (1, 4, 2); (1, 4, 4); (1, 4, 5); (1, 5, 1); (1, 5, 2); (1, 5, 3); (1, 5, 4); (1, 5, 5); (2, 1, 1); (2, 1, 2); (2, 1, 3); (2, 1, 4); (2, 1, 5); (2, 2, 1); (2, 2, 2); (2, 2, 4); (2, 2, 5); (2, 3, 1); (2, 3, 3); (2, 4, 2); (2, 4, 4); (2, 4, 5); (2, 5, 1); (2, 5, 3); (2, 5, 4); (2, 5, 5); (3, 1, 1); (3, 1, 3); (3, 1, 5); (3, 2, 2); (3, 2, 4); (3, 3, 1); (3, 3, 3); (3, 3, 5); (3, 4, 1); (3, 4, 2); (3, 4, 4); (3, 5, 2); (3, 5, 3); (3, 5, 5); (4, 1, 1); (4, 1, 2); (4, 1, 4); (4, 1, 5); (4, 2, 1); (4, 2, 3); (4, 2, 5); (4, 3, 2); (4, 3, 4); (4, 4, 1); (4, 4, 3); (4, 4, 5); (4, 5, 1); (4, 5, 2); (4, 5, 4); (4, 5, 5); (5, 1, 1); (5, 1, 3); (5, 1, 4); (5, 1, 5); (5, 2, 1); (5, 2, 2); (5, 2, 4); (5, 3, 1); (5, 3, 3); (5, 3, 5); (5, 4, 1); (5, 4, 2); (5, 4, 4); (5, 4, 5); (5, 5, 1); (5, 5, 3); (5, 5, 5)
	90%	110	(1, 1, 1); (1, 1, 2); (1, 1, 3); (1, 1, 4); (1, 1, 5); (1, 2, 1); (1, 2, 2); (1, 2, 3); (1, 2, 4); (1, 2, 5); (1, 3, 1); (1, 3, 2); (1, 3, 3); (1, 3, 4); (1, 3, 5); (1, 4, 1); (1, 4, 2); (1, 4, 3); (1, 4, 4); (1, 4, 5); (1, 5, 1); (1, 5, 2); (1, 5, 3); (1, 5, 4); (1, 5, 5); (2, 1, 1); (2, 1, 2); (2, 1, 3); (2, 1, 4); (2, 1, 5); (2, 2, 1); (2, 2, 2); (2, 2, 4); (2, 2, 5); (2, 3, 1); (2, 3, 3); (2, 3, 5); (2, 4, 1); (2, 4, 2); (2, 4, 4); (2, 4, 5); (2, 5, 1); (2, 5, 2); (2, 5, 3); (2, 5, 4); (2, 5, 5); (3, 1, 1); (3, 1, 2); (3, 1, 3); (3, 1, 5); (3, 2, 2); (3, 2, 4); (3, 2, 5); (3, 3, 1); (3, 3, 3); (3, 3, 5); (3, 4, 1); (3, 4, 2); (3, 4, 4); (3, 5, 1); (3, 5, 2); (3, 5, 3); (3, 5, 4); (3, 5, 5); (4, 1, 1);

			(4, 1, 2); (4, 1, 3); (4, 1, 4); (4, 1, 5); (4, 2, 1); (4, 2, 3); (4, 2, 4); (4, 2, 5); (4, 3, 1); (4, 3, 2); (4, 3, 4); (4, 3, 5); (4, 4, 1); (4, 4, 3); (4, 4, 5); (4, 5, 1); (4, 5, 2); (4, 5, 3); (4, 5, 4); (4, 5, 5); (5, 1, 1); (5, 1, 2); (5, 1, 3); (5, 1, 4); (5, 1, 5); (5, 2, 1); (5, 2, 2); (5, 2, 3); (5, 2, 4); (5, 2, 5); (5, 3, 1); (5, 3, 2); (5, 3, 3); (5, 3, 4); (5, 3, 5); (5, 4, 1); (5, 4, 2); (5, 4, 3); (5, 4, 4); (5, 4, 5); (5, 5, 1); (5, 5, 2); (5, 5, 3); (5, 5, 4); (5, 5, 5)

Appendix F: MC Response Data Set and Augmented data set $\boldsymbol{Y}$

Table 54: Monte Carlo Data Sets

x	Y^{MC}- System 1	Y^{MC}- System 2
1.00	6.798998681	25.96423782
2.00	39.20088643	26.83080437
3.00	120.7742288	27.38741513
4.00	277.3377493	28.43806341
5.00	525.5891761	28.80622686
6.00	900.1173153	28.92755346
7.00	1416.673854	29.28707671
8.00	2118.296116	30.6192136
9.00	2981.370319	31.0832378
10.00	4066.837971	31.55190042
11.00	5467.005852	31.73343556
12.00	7123.517974	32.68922354
13.00	8989.831558	33.3273705
14.00	11223.89717	33.55929165
15.00	13641.91875	33.99557626
16.00	16507.60378	34.5891125
17.00	20076.91167	35.11661943
18.00	23844.52609	35.19720066
19.00	27508.46435	35.29033292
20.00	32193.68279	35.48173608

21.00	37574.64921	35.71051947
22.00	43056.76744	35.83052729
23.00	49309.72943	36.22838575
24.00	56233.81793	36.23234205
25.00	62767.05007	36.3186531
26.00	71004.70142	36.40782906
27.00	79277.58911	36.89625045
28.00	88248.21932	36.90168468
29.00	97605.90169	36.93945145
30.00	108343.5368	37.00202468
31.00	120606.1794	37.36379894
32.00	130622.8326	37.40786396
33.00	146249.6938	38.33809514
34.00	157646.417	38.773925
35.00	172027.4515	38.84404704
36.00	187155.3353	38.98124431
37.00	203848.8216	39.11639523
38.00	220638.7253	39.2291271
39.00	240023.5079	39.3741915
40.00	256562.1858	39.62250629
41.00	279219.6379	39.88628249
42.00	295371.1084	40.110814
43.00	322052.3614	40.40346665
44.00	342493.596	40.46390258
45.00	365872.0923	40.63606992
46.00	390821.0363	40.94009127
47.00	421033.9519	40.96887677
48.00	446147.9875	41.14913862
49.00	472860.8039	41.1717623
50.00	503594.6952	41.5549779
51.00	536075.2562	41.58335413
52.00	568951.7473	41.5888111
53.00	593618.1417	41.86775312
54.00	634050.9005	41.87521098
55.00	666695.4716	41.9251195

56.00	711200.5516	42.04746928
57.00	739593.3392	42.2525376
58.00	778167.6302	42.30922185
59.00	824730.3776	42.3143605
60.00	869762.6102	42.4238864
61.00	907166.1314	42.43304956
62.00	962608.6404	42.4757617
63.00	1005810.023	42.61212906
64.00	1055453.011	42.65427277
65.00	1097689.124	42.71026354
66.00	1153461.244	42.7491217
67.00	1203959.502	42.7521003
68.00	1250402.16	42.81264991
69.00	1320917.656	43.27262685
70.00	1376327.818	43.38824637
71.00	1444749.612	43.43385753
72.00	1504225.325	43.4585162
73.00	1569478.194	43.5670985
74.00	1640224.533	43.65902568
75.00	1673881.4	44.15820184
76.00	1759440.408	44.24688895
77.00	1824768.804	44.30995103
78.00	1914540.966	44.49249799
79.00	1976080.431	44.51563326
80.00	2056605.75	44.76381163
81.00	2130335.715	44.76906396
82.00	2224269.352	45.09871031
83.00	2289766.683	45.27221917
84.00	2378400.539	45.27221917
85.00	2462327.28	45.30898549
86.00	2541476.247	45.32262791
87.00	2605987.964	45.37378699
88.00	2765311.609	45.41460056
89.00	2825989.644	45.42139903
90.00	2922527.544	45.50618668

91.00	3028085.649	45.59993284
92.00	3109106.208	45.63530082
93.00	3212835.277	45.6857323
94.00	3346107.221	45.80464873
95.00	3466549.855	45.87721504
96.00	3522833.372	45.90136213
97.00	3647467.093	45.99519924
98.00	3742708.899	46.12251713
99.00	3880245.295	46.43732735
100.00	4003763.519	46.48456424
101.00	4119662.276	46.54065732
102.00	4242915.493	46.58609795
103.00	4407326.277	46.68001465
104.00	4520080.139	46.74057563
105.00	4651376.968	46.89428023
106.00	4776308.696	46.96404984
107.00	4912306.465	46.98086413
108.00	5060561.904	47.22738266
109.00	5190668.86	47.47142283
110.00	5300763.436	48.01841568
111.00	5482268.832	48.1423116
112.00	5665537.935	48.15087222
113.00	5805244.484	48.35825975
114.00	5954992.728	48.53866939
115.00	6148459.14	48.58040383
116.00	6219370.684	48.6167154
117.00	6397475.503	48.76176844
118.00	6567145.906	48.80493533
119.00	6679552.676	48.96042482
120.00	6969852.076	49.10722863
121.00	7059941.632	49.4799623
122.00	7271409.705	49.5511871
123.00	7458622.957	49.57875616
124.00	7613585.012	49.592535
125.00	7745872.508	49.82671852

126.00	8042189.485	50.37837253
127.00	8198976.434	50.41435442
128.00	8393665.715	50.4886374
129.00	8657512.299	50.70244823
130.00	8800848.915	50.70474471
131.00	8988174.761	51.23513928
132.00	9217052.355	51.23976633
133.00	9339650.97	51.25903625
134.00	9684586.608	51.291437
135.00	9842126.553	51.58980811
136.00	10049038.1	51.72476575
137.00	10310965.89	51.82501481
138.00	10467165.03	52.02606998
139.00	10765988.16	52.03075388
140.00	10994127.65	52.0393486
141.00	11246532.15	52.05028527
142.00	11500265.03	52.07998028
143.00	11617519.92	52.21541541
144.00	12089006.5	52.35990001
145.00	12249891.22	52.44415332
146.00	12420107.42	52.62109552
147.00	12787795.15	52.70987357
148.00	13017123.82	52.73289515
149.00	13279966.88	52.87053581
150.00	13519452.39	52.92558298
151.00	13871955.95	52.98232408
152.00	14099208.19	53.00151441
153.00	14427453.39	53.04717105
154.00	14485317.04	53.1434638
155.00	15073203.74	53.19012088
156.00	15281094.27	53.53415999
157.00	15537569.04	53.762284
158.00	15798595.82	53.85934982
159.00	16011202.29	54.29245119
160.00	16390851.71	54.35123866

161.00	16715548.23	54.37564722
162.00	17023585.36	54.4608214
163.00	17285074.71	54.57860097
164.00	17764456.9	54.8280981
165.00	18074003.11	54.90647381
166.00	18180119.36	54.94014785
167.00	18613399.29	55.09282927
168.00	19098113.69	55.11026883
169.00	19368427.41	55.19675041
170.00	19712850.13	55.24935331
171.00	20110615.24	55.34089395
172.00	20319137.24	55.36472271
173.00	20685912.37	55.49415518
174.00	21102992.06	55.55207862
175.00	21522826.85	55.92365268
176.00	21606361.31	56.0379989
177.00	22138684.19	56.17442311
178.00	22710102.59	56.28774615
179.00	22946791.5	56.56328893
180.00	23066222.93	56.69383553
181.00	23757838.06	56.76386662
182.00	24136169.5	56.76964191
183.00	24604449.8	56.83327244
184.00	24992233.25	56.89135504
185.00	25283281.09	56.91563855
186.00	25860058.3	56.96038569
187.00	26122857.56	56.99357088
188.00	26724560.61	57.45335456
189.00	27145031.53	57.50089839
190.00	27551035.7	57.55474048
191.00	28099493	57.67022357
192.00	28461413.43	57.69077815
193.00	28855907.03	57.95359938
194.00	29479647.59	58.01671831
195.00	29646733	58.02516524

196.00	30173953.46	58.08336154
197.00	30710199.32	58.4450221
198.00	31220954.29	58.62894467
199.00	31523039.93	58.9909463
200.00	32110147.85	59.06106834
201.00	32614588.17	59.09572009
202.00	32787102.07	59.17002581
203.00	33483035.05	59.32841431
204.00	33780394.21	59.52645678
205.00	34613439.46	59.54332791
206.00	35154488.91	59.6062422
207.00	35237131.41	59.61838396
208.00	36109420.77	59.74796421
209.00	36450987.26	59.89916771
210.00	36972996.84	59.91167326
211.00	37603789.94	60.01976671
212.00	38181870.4	60.18156581
213.00	38473145	60.30177827
214.00	39479950.94	60.30566636
215.00	39886857.19	60.42612894
216.00	40123172.36	60.56412202
217.00	40927040.07	60.67435278
218.00	41485954.18	61.16022759
219.00	41856353.68	61.28514668
220.00	42989979.06	61.31413683
221.00	43182665.8	61.36797891
222.00	44243616.84	61.4045406
223.00	44586029.87	62.08609319
224.00	44653404.57	62.14343683
225.00	45930403.97	62.53320079
226.00	46422631.58	62.87785381
227.00	46909818.08	63.25506673
228.00	47571552.49	63.52696017
229.00	48024505.81	63.54223969
230.00	48781255.67	64.70289135

231.00	48762747.78	65.15031727
232.00	49808021.92	65.58373697
233.00	50710921.49	66.58727342
234.00	51473388.14	66.58727342
235.00	52420685.72	67.1321517
236.00	52769229.83	67.62377906
237.00	53204848.49	67.86756911
238.00	54203237.71	68.21408659
239.00	54364575.81	68.28693712
240.00	54878193.13	68.31140253
241.00	55742779.47	68.62717909
242.00	56787118.21	68.74113877
243.00	57590041.03	69.28656275

Table 55: Augmented Data Sets to Include Interpolated Data

x	Y- System 1	Y- System 2
1.00	6.8	32.90095391
2.00	3755.6	33.04298327
3.00	7504.4	33.18501263
4.00	11253.2	33.32704199
5.00	15002	33.46907134
6.00	18750.8	33.6111007
7.00	22499.6	33.75313006
8.00	26248.4	33.89515942
9.00	29997.2	34.03718878
10.00	33746	34.17921814
11.00	37494.8	34.32124749
12.00	41243.6	34.46327685
13.00	44992.4	34.60530621
14.00	48741.2	34.74733557
15.00	52490	34.88936493
16.00	56238.8	35.03139429
17.00	59987.6	35.17342364

18.00	63736.4	35.315453
19.00	67485.2	35.45748236
20.00	71234	35.59951172
21.00	74982.8	35.74154108
22.00	78731.6	35.88357044
23.00	82480.4	36.02559979
24.00	86229.2	36.16762915
25.00	89978	36.30965851
26.00	93726.8	36.45168787
27.00	97475.6	36.59371723
28.00	101224.4	36.73574659
29.00	104973.2	36.87777594
30.00	108722	37.0198053
31.00	200859.0114	37.05959115
32.00	292996.0229	37.09937699
33.00	385133.0343	37.13916284
34.00	477270.0457	37.17894869
35.00	569407.0572	37.21873453
36.00	661544.0686	37.25852038
37.00	753681.08	37.29830622
38.00	774624.7345	37.33809207
39.00	795568.389	37.37787791
40.00	816512.0435	37.41766376
41.00	837455.698	37.45744961
42.00	858399.3525	37.49723545
43.00	879343.007	37.5370213
44.00	900286.6615	37.57680714
45.00	921230.316	37.61659299
46.00	942173.9705	37.65637883
47.00	936800.1154	37.69616468
48.00	931426.2604	37.73595053
49.00	926052.4054	37.77573637
50.00	920678.5503	37.81552222
51.00	915304.6953	37.85530806
52.00	909930.8403	37.89509391

53.00	904556.9852	37.93487975
54.00	899183.1302	37.9746656
55.00	893809.2752	38.01445145
56.00	888435.4201	38.05423729
57.00	883061.5651	38.09402314
58.00	877687.7101	38.13380898
59.00	872313.855	38.17359483
60.00	866940	38.21338068
61.00	935462	38.3199295
62.00	1003984	38.42647832
63.00	1072506	38.53302714
64.00	1141028	38.63957596
65.00	1209550	38.74612478
66.00	1278072	38.8526736
67.00	1346594	38.95922242
68.00	1415116	39.06577124
69.00	1483638	39.17232006
70.00	1552160	39.27886888
71.00	1620682	39.3854177
72.00	1689204	39.49196652
73.00	1757726	39.59851534
74.00	1826248	39.70506416
75.00	1894770	39.81161298
76.00	1963292	39.9181618
77.00	2031814	40.02471062
78.00	2100336	40.13125944
79.00	2168858	40.23780826
80.00	2237380	40.34435708
81.00	2305902	40.4509059
82.00	2374424	40.55745472
83.00	2442946	40.66400354
84.00	2511468	40.77055236
85.00	2579990	40.87710118
86.00	2648512	40.98365
87.00	2717034	41.09019882

88.00	2785556	41.19674764
89.00	2854078	41.30329646
90.00	2922600	41.40984528
91.00	3023242.4	42.92003849
92.00	3123884.8	44.43023171
93.00	3224527.2	45.94042492
94.00	3335309.8	46.15677098
95.00	3446092.4	46.37311703
96.00	3556875	46.58946308
97.00	3667657.6	46.80580913
98.00	3778440.2	47.02215518
99.00	3889222.8	47.23850124
100.00	3728610.099	48.08721517
101.00	3990981.65	48.93592909
102.00	4253353.2	49.78464302
103.00	4383329.1	50.36512517
104.00	4513305	50.94560733
105.00	4643280.9	51.52608948
106.00	4773256.8	52.10657163
107.00	4912115.467	52.85703739
108.00	5050974.133	53.60750316
109.00	5189832.8	54.35796892
110.00	5336543.6	54.36021234
111.00	5483254.4	54.36245576
112.00	5629965.2	54.36469918
113.00	5784744.8	54.84377172
114.00	5939524.4	55.32284427
115.00	6094304	55.80191681
116.00	6257368.4	56.13476535
117.00	6420432.8	56.46761388
118.00	6583497.2	56.80046242
119.00	6755062.4	57.97103894
120.00	6926627.6	59.14161546
121.00	7098192.8	60.31219199
122.00	7275541.2	63.14901965

123.00	7469392.6	63.17506758
124.00	7663244	63.20111551
125.00	7857095.4	63.22716344
126.00	8050946.8	63.25321136
127.00	8244798.2	63.27925929
128.00	8438649.6	63.30530722
129.00	8632501	63.33135515
130.00	8826352.4	63.35740308
131.00	9020203.8	63.38345101
132.00	9214055.2	63.40949893
133.00	9437943.822	63.41578961
134.00	9661832.444	63.42208028
135.00	9885721.067	63.42837095
136.00	10109609.69	63.43466162
137.00	10333498.31	63.44095229
138.00	10557386.93	63.44724296
139.00	10781275.56	63.45353364
140.00	11005164.18	63.45982431
141.00	11229052.8	63.46611498
142.00	11645785.68	63.57124674
143.00	12062518.56	63.67637849
144.00	12479251.43	63.78151025
145.00	12789926.69	63.88664201
146.00	13100601.95	63.99177377
147.00	13411277.21	64.09690552
148.00	13721952.47	64.20203728
149.00	14032627.73	64.30716904
150.00	14343302.99	64.4123008
151.00	14653978.24	64.51743255
152.00	14964653.5	64.62256431
153.00	15275328.76	64.72769607
154.00	15586004.02	64.83282783
155.00	15896679.28	64.93795958
156.00	16207354.54	65.04309134
157.00	16518029.79	65.1482231

158.00	16828705.05	65.25335485
159.00	17139380.31	65.35848661
160.00	17450055.57	65.46361837
161.00	17760730.83	65.56875013
162.00	18071406.09	65.67388188
163.00	18382081.35	65.77901364
164.00	18692756.6	65.8841454
165.00	19003431.86	65.98927716
166.00	19314107.12	66.09440891
167.00	19624782.38	66.19954067
168.00	19935457.64	66.30467243
169.00	20246132.9	66.40980419
170.00	20556808.16	66.51493594
171.00	20867483.41	66.6200677
172.00	21178158.67	66.72519946
173.00	21488833.93	66.83033122
174.00	21755315.08	66.93546297
175.00	22021796.24	67.04059473
176.00	22288277.39	67.14572649
177.00	22554758.54	67.25085825
178.00	22821239.69	67.35599
179.00	23087720.85	67.46112176
180.00	23354202	67.56625352
181.00	23763069.87	67.58604008
182.00	24171937.73	67.60582664
183.00	24580805.6	67.62561321
184.00	24989673.47	67.64539977
185.00	25398541.33	67.66518633
186.00	25807409.2	67.68497289
187.00	26216277.07	67.70475946
188.00	26625144.93	67.72454602
189.00	27034012.8	67.74433258
190.00	27495515.69	67.95451031
191.00	27957018.57	68.16468804
192.00	28418521.46	68.37486577

193.00	28880024.34	68.5850435
194.00	29341527.23	68.79522123
195.00	29803030.11	69.00539896
196.00	30264533	69.21557669
197.00	30726035.89	69.42575441
198.00	31187538.77	69.63593214
199.00	31649041.66	69.84610987
200.00	32110544.54	70.0562876
201.00	32572047.43	70.26646533
202.00	33033550.31	70.47664306
203.00	33495053.2	70.68682079
204.00	34095268.58	70.82481856
205.00	34695483.95	70.96281633
206.00	35295699.33	71.1008141
207.00	35895914.7	71.23881187
208.00	36496130.08	71.37680964
209.00	37096345.46	71.51480741
210.00	37696560.83	71.65280518
211.00	38296776.21	71.79080295
212.00	38896991.59	71.92880072
213.00	39497206.96	72.06679849
214.00	40097422.34	72.20479626
215.00	40697637.71	72.23727991
216.00	41297853.09	72.26976356
217.00	41898068.47	72.3022472
218.00	42498283.84	72.33473085
219.00	43098499.22	72.3672145
220.00	43698714.6	72.39969815
221.00	44298929.97	72.4321818
222.00	44899145.35	72.46466545
223.00	45499360.72	72.49714909
224.00	46099576.1	72.52963274
225.00	46699791.48	72.56211639
226.00	47300006.85	72.59460004
227.00	47900222.23	72.62708369

228.00	48500437.61	72.65956733
229.00	49100652.98	72.69205098
230.00	49700868.36	72.72453463
231.00	50301083.73	72.75701828
232.00	50901299.11	72.78950193
233.00	51501514.49	72.82198558
234.00	52085669.66	72.85446922
235.00	52669824.84	72.88695287
236.00	53253980.02	72.91943652
237.00	53833065.09	72.95192017
238.00	54412150.17	72.98440382
239.00	54991235.24	73.01688747
240.00	55570320.32	73.04937111
241.00	56149405.4	73.08185476
242.00	56728490.47	73.11433841
243.00	57443273.2	73.14682206

Appendix G: Additional Initial Sample Size and FLC Selection Data

Table 56: Initial Sample Size and FLC Selection Data Compilation

	N	Pilot Design n_{ir}		FLC Selection
		Smooth Expected Response	Rough Expected Response	
$d = 1$	10	4	6	(1); (5); (8); (10)
				(1); (3); (5); (7); (8); (10)
	25	7	10	(1); (4); (7); (13); (19); (22); (25)
				(1); (2); (4); (7); (10); (13); (16); (19); (22); (25)
	50	9	15	(1); (7); (13); (19); (25); (32); (38); (44); (50)
				(1); (4); (7); (10); (13); (16); (19); (25); (29); (32); (38); (41); (44); (47); (50)
	75	11	21	(1); (10); (19); (28); (33); (38); (43); (48); (57); (66); (75)
				(1); (5); (7); (10); (14); (16); (19); (23); (28); (33); (38); (43); (48); (53); (57); (60); (62); (66); (69); (71); (75)

	100	13	25	(1); (7); (13); (25); (37); (43); (50); (57); (63); (75); (82); (88); (100)
				(1); (4); (7); (10); (13); (16); (19); (25); (31); (37); (43); (46); (50); (54); (57); (63); (69); (75); (79); (82); (88); (91); (94); (97); (100)
	125	14	28	(1); (16); (24); (32); (47); (55); (63); (71); (79); (94); (102); (110); (118); (125)
				(1); (8); (12); (16); (20); (24); (28); (32); (39); (43); (47); (51); (55); (63); (67); (71); (75); (79); (83); (87); (94); (98); (102); (106); (110); (114); (118); (125)
$d = 2$	N	Pilot Design n_{ir}		FLC Selection
		Smooth Expected Response	Rough Expected Response	
	9	6	7	(1, 1); (1, 2); (1, 3); (2, 2); (3, 1); (3, 3)
				(1, 1); (1, 2); (1, 3); (2, 2); (3, 1); (3, 2); (3, 3)
	25	12	16	(1, 1); (1, 3); (1, 5); (2, 2); (3, 1); (3, 3); (3, 5); (4, 2); (4, 4); (5, 1); (5, 3); (5, 5)
				(1, 1); (1, 2); (1, 3); (1, 5); (2, 2); (2, 4); (2, 5); (3, 1); (3, 3); (3, 5); (4, 1); (4, 2); (4, 4); (5, 1); (5, 3); (5, 5)
	49	20	28	(1, 1); (1, 4); (1, 7); (2, 2); (2, 6); (3, 3); (3, 5); (3, 7); (4, 1); (4, 4); (4, 7); (5, 1); (5, 3); (5, 5); (6, 2); (6, 6); (7, 1); (7, 4); (7, 5); (7, 7)
				(1, 1); (1, 3); (1, 4); (1, 5); (1, 7); (2, 2); (2, 4); (2, 6); (3, 1); (3, 3); (3, 5); (3, 7); (4, 1); (4, 4); (4, 6); (4, 7); (5, 1); (5, 3); (5, 5); (5, 7); (6, 2); (6, 4); (6, 6); (7, 1); (7, 3); (7, 4); (7, 5); (7, 7)
	81	28	43	(1, 1); (1, 3); (1, 5); (1, 7); (1, 9); (2, 2); (3, 1); (3, 3); (3, 5); (3, 7); (3, 9); (5, 1); (5, 3); (5, 5); (5, 7); (5, 9); (7, 1); (7, 3); (7, 5); (7, 7); (7, 9); (8, 2); (8, 8); (9, 1); (9, 3); (9, 5); (9, 7); (9, 9)
				(1, 1); (1, 2); (1, 3); (1, 5); (1, 7); (1, 9); (2, 2); (2, 4); (2, 6); (2, 8); (3, 1); (3, 3); (3, 5); (3, 7); (3, 9); (4, 2); (4, 4); (4, 6); (4, 8); (5, 1); (5, 3); (5, 5); (5, 7); (5, 9); (6, 2); (6, 4); (6, 6); (6, 8); (7, 1); (7, 3); (7, 5); (7, 7); (7, 9); (8, 1); (8, 2); (8, 4); (8, 6); (8, 8); (9, 1); (9, 3); (9, 5); (9, 7); (9, 9)

	100	34	53	(1, 1); (1, 3); (1, 6); (1, 8); (1, 10); (2, 1); (2, 5); (2, 7); (3, 3); (3, 8); (3, 10); (4, 1); (4, 5); (4, 7); (4, 9); (5, 2); (5, 4); (5, 10); (6, 3); (6, 7); (6, 9); (7, 1); (7, 5); (7, 8); (8, 2); (8, 7); (8, 10); (9, 4); (9, 9); (10, 1); (10, 3); (10, 6); (10, 8); (10, 10)
				(1, 1); (1, 3); (1, 6); (1, 8); (1, 10); (2, 1); (2, 5); (2, 7); (3, 3); (3, 8); (3, 10); (4, 1); (4, 4); (4, 5); (4, 7); (4, 9); (5, 2); (5, 4); (5, 6); (5, 8); (5, 10); (6, 1); (6, 2); (6, 3); (6, 5); (6, 7); (6, 9); (7, 1); (7, 3); (7, 5); (7, 6); (7, 8); (7, 10); (8, 2); (8, 3); (8, 4); (8, 5); (8, 7); (8, 8); (8, 10); (9, 1); (9, 4); (9, 6); (9, 9); (9, 10); (10, 1); (10, 2); (10, 3); (10, 5); (10, 6); (10, 7); (10, 8); (10, 10)
	121	41	62	(1, 1); (1, 4); (1, 6); (1, 9); (1, 11); (3, 3); (3, 9); (3, 11); (4, 1); (4, 4); (4, 6); (4, 11); (5, 3); (5, 5); (5, 8); (5, 10); (6, 1); (6, 4); (6, 6); (6, 11); (7, 3); (7, 5); (7, 7); (7, 9); (8, 1); (8, 4); (8, 8); (8, 11); (9, 3); (9, 6); (9, 10); (10, 2); (10, 5); (10, 8); (10, 11); (11, 1); (11, 4); (11, 6); (11, 7); (11, 9); (11, 11)
				(1, 1); (1, 3); (1, 4); (1, 6); (1, 8); (1, 9); (1, 11); (2, 2); (2, 4); (2, 7); (2, 8); (2, 10); (3, 1); (3, 3); (3, 5); (3, 6); (3, 8); (3, 9); (3, 11); (4, 1); (4, 2); (4, 4); (4, 6); (4, 9); (4, 11); (5, 3); (5, 5); (5, 7); (5, 8); (5, 10); (6, 1); (6, 4); (6, 6); (6, 9); (6, 11); (7, 2); (7, 3); (7, 5); (7, 7); (7, 9); (7, 10); (8, 1); (8, 4); (8, 6); (8, 8); (8, 11); (9, 1); (9, 3); (9, 6); (9, 9); (9, 10); (10, 2); (10, 5); (10, 8); (10, 11); (11, 1); (11, 3); (11, 4); (11, 6); (11, 7); (11, 9); (11, 11)
$d = 3$	N	Pilot Design n_{ir}		FLC Selection
		Smooth Expected Response	Rough Expected Response	
	27	17	20	(1, 1, 1); (1, 1, 2); (1, 1, 3); (1, 2, 2); (1, 3, 1); (1, 3, 3); (2, 1, 2); (2, 2, 1); (2, 2, 2); (2, 2, 3); (2, 3, 2); (3, 1, 1); (3, 1, 3); (3, 2, 2); (3, 2, 3); (3, 3, 1); (3, 3, 3)
				(1, 1, 1); (1, 1, 2); (1, 1, 3); (1, 2, 2); (1, 3, 1); (1, 3, 3); (2, 1, 2); (2, 2, 1); (2, 2, 2); (2, 2, 3); (2, 3, 1); (2, 3, 2); (2, 3, 3); (3, 1, 1); (3, 1, 3); (3, 2, 1); (3, 2, 2); (3, 2, 3); (3, 3, 1); (3, 3, 3)

	64	32	41	(1, 1, 1); (1, 1, 3); (1, 1, 4); (1, 2, 2); (1, 3, 1); (1, 3, 3); (1, 4, 1); (1, 4, 2); (1, 4, 4); (2, 1, 1); (2, 1, 3); (2, 2, 1); (2, 2, 4); (2, 3, 2); (2, 4, 1); (2, 4, 4); (3, 1, 2); (3, 1, 4); (3, 2, 3); (3, 3, 1); (3, 3, 4); (3, 4, 2); (4, 1, 1); (4, 1, 3); (4, 1, 4); (4, 2, 1); (4, 2, 2); (4, 2, 4); (4, 3, 3); (4, 4, 1); (4, 4, 3); (4, 4, 4)
				(1, 1, 1); (1, 1, 2); (1, 1, 3); (1, 1, 4); (1, 2, 2); (1, 2, 4); (1, 3, 1); (1, 3, 3); (1, 3, 4); (1, 4, 1); (1, 4, 2); (1, 4, 4); (2, 1, 1); (2, 1, 3); (2, 1, 4); (2, 2, 1); (2, 2, 4); (2, 3, 2); (2, 4, 1); (2, 4, 3); (2, 4, 4); (3, 1, 1); (3, 1, 2); (3, 1, 4); (3, 2, 3); (3, 3, 1); (3, 3, 4); (3, 4, 1); (3, 4, 2); (3, 4, 4); (4, 1, 1); (4, 1, 3); (4, 1, 4); (4, 2, 1); (4, 2, 2); (4, 2, 4); (4, 3, 2); (4, 3, 3); (4, 4, 1); (4, 4, 3); (4, 4, 4)
	125	53	75	(1, 1, 1); (1, 1, 2); (1, 1, 4); (1, 1, 5); (1, 2, 3); (1, 3, 1); (1, 3, 3); (1, 3, 5); (1, 4, 2); (1, 5, 1); (1, 5, 3); (1, 5, 5); (2, 2, 1); (2, 2, 2); (2, 2, 4); (2, 2, 5); (2, 4, 2); (2, 4, 4); (2, 4, 5); (2, 5, 1); (2, 5, 4); (3, 1, 1); (3, 1, 3); (3, 1, 5); (3, 3, 1); (3, 3, 3); (3, 3, 5); (3, 4, 1); (3, 5, 2); (3, 5, 3); (3, 5, 5); (4, 1, 2); (4, 1, 4); (4, 2, 1); (4, 2, 3); (4, 2, 5); (4, 3, 4); (4, 4, 3); (4, 4, 5); (4, 5, 1); (5, 1, 1); (5, 1, 3); (5, 1, 5); (5, 2, 2); (5, 2, 4); (5, 3, 1); (5, 3, 3); (5, 3, 5); (5, 4, 2); (5, 4, 4); (5, 5, 1); (5, 5, 3); (5, 5, 5)
				(1, 1, 1); (1, 1, 2); (1, 1, 4); (1, 1, 5); (1, 2, 1); (1, 2, 3); (1, 2, 5); (1, 3, 1); (1, 3, 3); (1, 3, 5); (1, 4, 1); (1, 4, 2); (1, 4, 4); (1, 5, 1); (1, 5, 2); (1, 5, 3); (1, 5, 4); (1, 5, 5); (2, 1, 1); (2, 1, 3); (2, 1, 5); (2, 2, 1); (2, 2, 2); (2, 2, 4); (2, 2, 5); (2, 3, 3); (2, 4, 2); (2, 4, 4); (2, 4, 5); (2, 5, 1); (2, 5, 4); (2, 5, 5); (3, 1, 1); (3, 1, 3); (3, 1, 5); (3, 2, 2); (3, 2, 4); (3, 3, 1); (3, 3, 3); (3, 3, 5); (3, 4, 1); (3, 4, 4); (3, 5, 2); (3, 5, 3); (3, 5, 5); (4, 1, 1); (4, 1, 2); (4, 1, 4); (4, 2, 1); (4, 2, 3); (4, 2, 5); (4, 3, 2); (4, 3, 4); (4, 4, 1); (4, 4, 3); (4, 4, 5); (4, 5, 1); (4, 5, 2); (4, 5, 4); (4, 5, 5); (5, 1, 1); (5, 1, 3); (5, 1, 4); (5, 1, 5); (5, 2, 1); (5, 2, 2); (5, 2, 4); (5, 3, 1); (5, 3, 3); (5, 3, 5); (5, 4, 2); (5, 4, 4); (5, 5, 1); (5, 5, 3); (5, 5, 5)

	216	91	197	(1, 1, 1); (1, 1, 3); (1, 1, 5); (1, 1, 6); (1, 2, 3); (1, 2, 5); (1, 3, 1); (1, 3, 2); (1, 3, 4); (1, 3, 6); (1, 4, 1); (1, 4, 3); (1, 4, 5); (1, 4, 6); (1, 5, 2); (1, 5, 4); (1, 6, 1); (1, 6, 3); (1, 6, 5); (1, 6, 6); (2, 1, 2); (2, 1, 4); (2, 1, 5); (2, 2, 1); (2, 2, 2); (2, 2, 6); (2, 3, 1); (2, 3, 3); (2, 3, 5); (2, 4, 2); (2, 4, 5); (2, 5, 1); (2, 5, 4); (2, 5, 6); (2, 6, 2); (2, 6, 5); (3, 1, 1); (3, 1, 3); (3, 1, 4); (3, 1, 6); (3, 2, 3); (3, 2, 5); (3, 3, 2); (3, 3, 4); (3, 3, 6); (3, 4, 1); (3, 4, 3); (3, 4, 6); (3, 5, 2); (3, 5, 5); (3, 6, 1); (3, 6, 3); (3, 6, 4); (3, 6, 6); (4, 1, 2); (4, 1, 4); (4, 1, 6); (4, 2, 1); (4, 2, 3); (4, 2, 5); (4, 3, 1); (4, 3, 4); (4, 3, 6); (4, 4, 2); (4, 4, 4); (4, 4, 6); (4, 5, 1); (4, 5, 3); (4, 5, 5); (4, 5, 6); (4, 6, 1); (4, 6, 2); (4, 6, 4); (4, 6, 6); (5, 1, 1); (5, 1, 2); (5, 1, 3); (5, 1, 4); (5, 1, 5); (5, 1, 6); (5, 2, 1); (5, 2, 2); (5, 2, 4); (5, 2, 5); (6, 1, 1); (6, 1, 6); (6, 3, 3); (6, 4, 1); (6, 6, 1); (6, 6, 4); (6, 6, 6)

				(1, 1, 1); (1, 1, 2); (1, 1, 3); (1, 1, 4); (1, 1, 5); (1, 1, 6); (1, 2, 1); (1, 2, 2); (1, 2, 3); (1, 2, 4); (1, 2, 5); (1, 2, 6); (1, 3, 1); (1, 3, 2); (1, 3, 3); (1, 3, 4); (1, 3, 5); (1, 3, 6); (1, 4, 1); (1, 4, 2); (1, 4, 3); (1, 4, 4); (1, 4, 5); (1, 4, 6); (1, 5, 1); (1, 5, 2); (1, 5, 3); (1, 5, 4); (1, 5, 5); (1, 5, 6); (1, 6, 1); (1, 6, 2); (1, 6, 3); (1, 6, 4); (1, 6, 5); (1, 6, 6); (2, 1, 1); (2, 1, 2); (2, 1, 3); (2, 1, 4); (2, 1, 5); (2, 1, 6); (2, 2, 1); (2, 2, 2); (2, 2, 4); (2, 2, 6); (2, 3, 1); (2, 3, 2); (2, 3, 3); (2, 3, 5); (2, 3, 6); (2, 4, 1); (2, 4, 2); (2, 4, 4); (2, 4, 5); (2, 4, 6); (2, 5, 1); (2, 5, 2); (2, 5, 3); (2, 5, 4); (2, 5, 5); (2, 5, 6); (2, 6, 1); (2, 6, 2); (2, 6, 3); (2, 6, 4); (2, 6, 5); (2, 6, 6); (3, 1, 1); (3, 1, 2); (3, 1, 3); (3, 1, 4); (3, 1, 5); (3, 1, 6); (3, 2, 1); (3, 2, 3); (3, 2, 4); (3, 2, 5); (3, 2, 6); (3, 3, 1); (3, 3, 2); (3, 3, 4); (3, 3, 6); (3, 4, 1); (3, 4, 3); (3, 4, 5); (3, 4, 6); (3, 5, 1); (3, 5, 2); (3, 5, 4); (3, 5, 5); (3, 5, 6); (3, 6, 1); (3, 6, 2); (3, 6, 3); (3, 6, 4); (3, 6, 5); (3, 6, 6); (4, 1, 1); (4, 1, 2); (4, 1, 3); (4, 1, 4); (4, 1, 5); (4, 1, 6); (4, 2, 1); (4, 2, 2); (4, 2, 3); (4, 2, 5); (4, 2, 6); (4, 3, 1); (4, 3, 3); (4, 3, 4); (4, 3, 5); (4, 3, 6); (4, 4, 1); (4, 4, 2); (4, 4, 4); (4, 4, 6); (4, 5, 1); (4, 5, 2); (4, 5, 3); (4, 5, 5); (4, 5, 6); (4, 6, 1); (4, 6, 2); (4, 6, 3); (4, 6, 4); (4, 6, 5); (4, 6, 6); (5, 1, 1); (5, 1, 2); (5, 1, 3); (5, 1, 4); (5, 1, 5); (5, 1, 6); (5, 2, 1); (5, 2, 2); (5, 2, 3); (5, 2, 4); (5, 2, 5); (5, 2, 6); (5, 3, 1); (5, 3, 2); (5, 3, 4); (5, 3, 5); (5, 3, 6); (5, 4, 1); (5, 4, 3); (5, 4, 5); (5, 4, 6); (5, 5, 1); (5, 5, 2); (5, 5, 4); (5, 5, 5); (5, 5, 6); (5, 6, 1); (5, 6, 2); (5, 6, 3); (5, 6, 4); (5, 6, 5); (5, 6, 6); (6, 1, 1); (6, 1, 2); (6, 1, 3); (6, 1, 4); (6, 1, 5); (6, 1, 6); (6, 2, 1); (6, 2, 2); (6, 2, 3); (6, 2, 4); (6, 2, 5); (6, 2, 6); (6, 3, 1); (6, 3, 2); (6, 3, 3); (6, 3, 4); (6, 3, 5); (6, 3, 6); (6, 4, 1); (6, 4, 2); (6, 4, 3); (6, 4, 4); (6, 4, 5); (6, 4, 6); (6, 5, 1); (6, 5, 2); (6, 5, 3); (6, 5, 4); (6, 5, 5); (6, 5, 6); (6, 6, 1); (6, 6, 2); (6, 6, 3); (6, 6, 4); (6, 6, 5); (6, 6, 6)
$d = 4$	N	Pilot Design n_{ir}		FLC Selection
		Smooth Expected Response	Rough Expected Response	

<table>
<tr><td rowspan="2"></td><td rowspan="2">81</td><td rowspan="2">46</td><td rowspan="2">57</td><td>(1, 1, 1, 1); (1, 1, 1, 2); (1, 1, 1, 3); (1, 1, 2, 2); (1, 1, 3, 1); (1, 1, 3, 3); (1, 2, 1, 2); (1, 2, 2, 1); (1, 2, 2, 3); (1, 2, 3, 1); (1, 2, 3, 2); (1, 3, 1, 1); (1, 3, 1, 3); (1, 3, 2, 2); (1, 3, 3, 1); (1, 3, 3, 3); (2, 1, 1, 2); (2, 1, 2, 1); (2, 1, 2, 3); (2, 1, 3, 2); (2, 1, 3, 3); (2, 2, 1, 1); (2, 2, 1, 3); (2, 2, 2, 2); (2, 2, 3, 1); (2, 2, 3, 3); (2, 3, 1, 2); (2, 3, 2, 1); (2, 3, 2, 3); (2, 3, 3, 2); (3, 1, 1, 1); (3, 1, 1, 3); (3, 1, 2, 2); (3, 1, 3, 1); (3, 1, 3, 3); (3, 2, 1, 1); (3, 2, 1, 2); (3, 2, 2, 1); (3, 2, 2, 3); (3, 2, 3, 2); (3, 3, 1, 1); (3, 3, 1, 3); (3, 3, 2, 2); (3, 3, 2, 3); (3, 3, 3, 1); (3, 3, 3, 3)</td></tr>
<tr><td>(1, 1, 1, 1); (1, 1, 1, 2); (1, 1, 1, 3); (1, 1, 2, 1); (1, 1, 2, 2); (1, 1, 3, 1); (1, 1, 3, 3); (1, 2, 1, 2); (1, 2, 2, 1); (1, 2, 2, 3); (1, 2, 3, 1); (1, 2, 3, 2); (1, 3, 1, 1); (1, 3, 1, 2); (1, 3, 1, 3); (1, 3, 2, 2); (1, 3, 2, 3); (1, 3, 3, 1); (1, 3, 3, 2); (1, 3, 3, 3); (2, 1, 1, 2); (2, 1, 1, 3); (2, 1, 2, 1); (2, 1, 2, 3); (2, 1, 3, 2); (2, 1, 3, 3); (2, 2, 1, 1); (2, 2, 1, 3); (2, 2, 2, 2); (2, 2, 3, 1); (2, 2, 3, 3); (2, 3, 1, 1); (2, 3, 1, 2); (2, 3, 2, 1); (2, 3, 2, 3); (2, 3, 3, 2); (3, 1, 1, 1); (3, 1, 1, 3); (3, 1, 2, 1); (3, 1, 2, 2); (3, 1, 3, 1); (3, 1, 3, 2); (3, 1, 3, 3); (3, 2, 1, 1); (3, 2, 1, 2); (3, 2, 1, 3); (3, 2, 2, 1); (3, 2, 2, 3); (3, 2, 3, 1); (3, 2, 3, 2); (3, 3, 1, 1); (3, 3, 1, 3); (3, 3, 2, 2); (3, 3, 2, 3); (3, 3, 3, 1); (3, 3, 3, 2); (3, 3, 3, 3)</td></tr>
</table>

	256	117	158	(1, 1, 1, 1); (1, 1, 1, 2); (1, 1, 1, 3); (1, 1, 1, 4); (1, 1, 2, 1); (1, 1, 2, 3); (1, 1, 3, 2); (1, 1, 3, 3); (1, 1, 3, 4); (1, 1, 4, 1); (1, 1, 4, 3); (1, 1, 4, 4); (1, 2, 1, 1); (1, 2, 1, 4); (1, 2, 2, 2); (1, 2, 2, 4); (1, 2, 3, 1); (1, 2, 3, 3); (1, 2, 4, 1); (1, 2, 4, 2); (1, 2, 4, 4); (1, 3, 1, 2); (1, 3, 1, 3); (1, 3, 2, 1); (1, 3, 3, 1); (1, 3, 3, 4); (1, 3, 4, 2); (1, 4, 1, 1); (1, 4, 1, 3); (1, 4, 1, 4); (1, 4, 2, 2); (1, 4, 2, 4); (1, 4, 3, 1); (1, 4, 3, 3); (1, 4, 4, 1); (1, 4, 4, 3); (1, 4, 4, 4); (2, 1, 1, 1); (2, 1, 1, 4); (2, 1, 2, 2); (2, 1, 3, 1); (2, 1, 3, 4); (2, 1, 4, 2); (2, 2, 1, 1); (2, 2, 1, 3); (2, 2, 2, 4); (2, 2, 4, 3); (2, 2, 4, 4); (2, 3, 1, 4); (2, 3, 2, 1); (2, 3, 2, 3); (2, 3, 4, 1); (2, 3, 4, 3); (2, 4, 1, 1); (2, 4, 1, 3); (2, 4, 3, 1); (2, 4, 3, 2); (2, 4, 3, 4); (2, 4, 4, 4); (3, 1, 1, 2); (3, 1, 2, 1); (3, 1, 2, 3); (3, 1, 3, 4); (3, 1, 4, 1); (3, 1, 4, 3); (3, 1, 4, 4); (3, 2, 1, 4); (3, 2, 2, 1); (3, 2, 3, 2); (3, 2, 4, 1); (3, 3, 1, 1); (3, 3, 1, 3); (3, 3, 2, 2); (3, 3, 2, 4); (3, 3, 3, 3); (3, 3, 4, 2); (3, 3, 4, 4); (3, 4, 1, 2); (3, 4, 1, 4); (3, 4, 2, 1); (3, 4, 2, 3); (3, 4, 2, 4); (3, 4, 4, 1); (3, 4, 4, 2); (3, 4, 4, 3); (4, 1, 1, 1); (4, 1, 1, 3); (4, 1, 1, 4); (4, 1, 2, 1); (4, 1, 2, 4); (4, 1, 3, 2); (4, 1, 3, 3); (4, 1, 4, 1); (4, 1, 4, 2); (4, 1, 4, 4); (4, 2, 1, 2); (4, 2, 2, 3); (4, 2, 3, 1); (4, 2, 3, 4); (4, 2, 4, 2); (4, 2, 4, 4); (4, 3, 1, 1); (4, 3, 1, 4); (4, 3, 2, 3); (4, 3, 3, 1); (4, 3, 4, 1); (4, 3, 4, 3); (4, 4, 1, 1); (4, 4, 1, 3); (4, 4, 1, 4); (4, 4, 2, 2); (4, 4, 3, 1); (4, 4, 3, 3); (4, 4, 3, 4); (4, 4, 4, 1); (4, 4, 4, 2); (4, 4, 4, 4)

				(1, 1, 1, 1); (1, 1, 1, 2); (1, 1, 1, 3); (1, 1, 1, 4); (1, 1, 2, 1); (1, 1, 2, 3); (1, 1, 3, 2); (1, 1, 3, 3); (1, 1, 3, 4); (1, 1, 4, 1); (1, 1, 4, 3); (1, 1, 4, 4); (1, 2, 1, 1); (1, 2, 1, 4); (1, 2, 2, 2); (1, 2, 2, 4); (1, 2, 3, 1); (1, 2, 3, 3); (1, 2, 4, 1); (1, 2, 4, 2); (1, 2, 4, 4); (1, 3, 1, 1); (1, 3, 1, 2); (1, 3, 1, 3); (1, 3, 1, 4); (1, 3, 2, 1); (1, 3, 2, 3); (1, 3, 3, 1); (1, 3, 3, 2); (1, 3, 3, 4); (1, 3, 4, 2); (1, 3, 4, 4); (1, 4, 1, 1); (1, 4, 1, 2); (1, 4, 1, 3); (1, 4, 1, 4); (1, 4, 2, 1); (1, 4, 2, 2); (1, 4, 2, 4); (1, 4, 3, 1); (1, 4, 3, 3); (1, 4, 4, 1); (1, 4, 4, 2); (1, 4, 4, 3); (1, 4, 4, 4); (2, 1, 1, 1); (2, 1, 1, 3); (2, 1, 1, 4); (2, 1, 2, 2); (2, 1, 2, 4); (2, 1, 3, 1); (2, 1, 3, 3); (2, 1, 3, 4); (2, 1, 4, 1); (2, 1, 4, 2); (2, 1, 4, 4); (2, 2, 1, 1); (2, 2, 1, 3); (2, 2, 2, 4); (2, 2, 3, 1); (2, 2, 3, 2); (2, 2, 4, 3); (2, 2, 4, 4); (2, 3, 1, 2); (2, 3, 1, 4); (2, 3, 2, 1); (2, 3, 2, 3); (2, 3, 3, 4); (2, 3, 4, 1); (2, 3, 4, 3); (2, 4, 1, 1); (2, 4, 1, 3); (2, 4, 1, 4); (2, 4, 2, 2); (2, 4, 3, 1); (2, 4, 3, 2); (2, 4, 3, 4); (2, 4, 4, 2); (2, 4, 4, 4); (3, 1, 1, 1); (3, 1, 1, 2); (3, 1, 1, 4); (3, 1, 2, 1); (3, 1, 2, 3); (3, 1, 3, 2); (3, 1, 3, 4); (3, 1, 4, 1); (3, 1, 4, 3); (3, 1, 4, 4); (3, 2, 1, 2); (3, 2, 1, 4); (3, 2, 2, 1); (3, 2, 2, 3); (3, 2, 3, 2); (3, 2, 3, 4); (3, 2, 4, 1); (3, 2, 4, 3); (3, 3, 1, 1); (3, 3, 1, 3); (3, 3, 2, 2); (3, 3, 2, 4); (3, 3, 3, 1); (3, 3, 3, 3); (3, 3, 4, 2); (3, 3, 4, 4); (3, 4, 1, 2); (3, 4, 1, 4); (3, 4, 2, 1); (3, 4, 2, 3); (3, 4, 2, 4); (3, 4, 3, 2); (3, 4, 4, 1); (3, 4, 4, 2); (3, 4, 4, 3); (3, 4, 4, 4); (4, 1, 1, 1); (4, 1, 1, 3); (4, 1, 1, 4); (4, 1, 2, 1); (4, 1, 2, 2); (4, 1, 2, 4); (4, 1, 3, 1); (4, 1, 3, 2); (4, 1, 3, 3); (4, 1, 4, 1); (4, 1, 4, 2); (4, 1, 4, 3); (4, 1, 4, 4); (4, 2, 1, 1); (4, 2, 1, 2); (4, 2, 1, 3); (4, 2, 2, 3); (4, 2, 2, 4); (4, 2, 3, 1); (4, 2, 3, 4); (4, 2, 4, 2); (4, 2, 4, 4); (4, 3, 1, 1); (4, 3, 1, 2); (4, 3, 1, 4); (4, 3, 2, 1); (4, 3, 2, 3); (4, 3, 3, 1); (4, 3, 3, 2); (4, 3, 3, 4); (4, 3, 4, 1); (4, 3, 4, 3); (4, 4, 1, 1); (4, 4, 1, 3); (4, 4, 1, 4); (4, 4, 2, 2); (4, 4, 2, 4); (4, 4, 3, 1); (4, 4, 3, 3); (4, 4, 3, 4); (4, 4, 4, 1); (4, 4, 4, 2); (4, 4, 4, 4)