DACE Approach for Image Representation

A project report submitted in partial fulfilment of the
requirements for the Award of
the Degree of

Master of Technology

in

Artificial Intelligence

By

D . FLORANCE

Department of Computer & Information Sciences
School of Mathematics & Computer / Information Sciences
University of Hyderabad, Hyderabad 500046, India

June 2009

CERTIFICATE

This is to certify that the project work entitled **"DACE Approach for Image Representation"** being submitted to the University of Hyderabad by ***D. Florance***, *Registration number 07MCMI10*, in the partial fulfillment of the requirement for the award of degree of **Master of Technology in Artificial Intelligence** is a bonafide work carried out at the University of Hyderabad under my supervision.

Prof C.R.Rao
Project Supervisor,
Department of CIS,
University of Hyderabad

Dr. Rajeev wankar
Project Supervisor,
Department of CIS,
University of Hyderabad

Prof. Arun Agarwal
Head of Department,
Department of CIS,
University of Hyderabad

Prof. T. Amaranath
Dean,
School of MCIS,
University of Hyderabad

ACKNOWLEDGMENTS

First and foremost, I offer my sincerest gratitude to my supervisors,**Prof C.R.Rao** and **Dr.Rajeev Wankar** who has helped or supported me throughout this project with their patience and valuable suggestions. Their vast experience and profound knowledge have been a constant source for me throughout this project work. Their willingness to motivate me contributed tremendously to my project.

I am also grateful to the head of department **Prof. Arun Agarwal**, for providing excellent facilities and such a nice atmosphere for doing this project. I would like to extend my sincere thanks to Dean of school of Mathematics, Computers and Information Sciences for his valuable cooperation.

I convey my heartfelt thanks to **AI Lab staff** for allowing me to use the required equipments whenever needed.

I am extremely grateful to **P.Shalini** for her constructive suggestions and co-operation throughout the project

Finally, an honourable mention goes to my parents, sisters and friends for their understanding and support on me in completing this project. Without help of the particular person mentioned above, I would have faced many difficulties while doing this project.

D.FLORANCE

PAPER PRESENTED

D.Florance and P.Shalini, **"Image Representation: DACE Approach"**, Presented paper at **"INTERNATIONAL CONFERENCE ON OPERATION RESEACH FOR GROWING NATION"** held on 15 - 17 December 2008, Tirupati.

Abstract

The objective of this dissertation is to represent an image using DACE approach. Design and Analysis of Computer Experiments (DACE) is used to build surrogate models to construct approximate models. It uses kriging approximation model based on data from a computer experiment that are deterministic in nature. The DACE model is built for given deterministic input data that gives an approximate model as output. This concept can be used to build an approximation model for representing images. The model is constructed by calculating the parameters using fitted DACE model representing an image in mathematical equations.

Image is considered as deterministic computer experiments as for multiple evaluations same pixel values are obtained for fixed inputs. Thus DACE can be used for representing an image by constructing an approximate model for the image by taking sample points with respective pixel values. After the model is built it is capable to predict the unknown pixel values using the DACE model. The predicted pixel values can be represented as approximate for the original input image. The representation of the image through the parameters of the fitted DACE model and efficiency in representing an image and the applications involved in such type of representation is done in this dissertation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

An Image is represented in a spatial coordinate as a two dimensional function having static pixel elements for respective spatial coordinate domain. By observing the spatial and static representation of image literature survey is done for representing an image in spatial and static domain and that made to concentrate on the concept DACE ie Design and Analysis of Computer Experiments. It builds an approximate model for data from computer experiments which is a collection of pairs of input and response from runs of a computer model. By doing the survey a software package DACE, which is a Matlab toolbox for working with kriging approximations to computer models was studied and this concept was applied for representation of an image writing the code in Matlab.
The theoretical framework of DACE was developed by Sacks and co-workers who proposed a new model building strategy, that is especially suited for deterministic computer responses. The basic assumption is that computer responses can be modelled as a realization of a stochastic process. This finally leads to a response prediction that exactly describes all calculated computer responses. Main advantage of the approximations of Sacks et al., compared with other models, is the flexibility to automatically adapt to the calculated response data. Most of recent papers study two problems of DACE. One is to find and model the design optimization tool, and the other is to develop fast and practical reconstruction algorithms ie approximate models to recover the desired computer responses with low computational time and cost. In this project, DACE is applied for image representation and a new image representation scheme is proposed.

1.1 Motivation

Representation of an image as a computer code is crucial in computer vision and it is mainly about retrieving information from such type of codes which are deterministic in nature. Many applications depend on a good representation of image in order to perform well. The

representation of the image is involved in almost every field in computer vision and there might be a need for building an approximate model. Many ways have been suggested for modelling and representing colors, yet none of these ways has proved to be better than all others. Many computer vision applications depend on a good color representation in order to achieve good results. With time the fidelity of computer models to nature has also steadily increased. Even with the fastest computers, computing expense is a problem in representing an image with high fidelity simulation. A need has now arisen to replace expensive computer simulations with alternative cheap surrogates. These surrogates are approximate models which replace the representation of an image with the original color image. In this, the work is focused on representing an image by building DACE model considering the kriging approximation and then using the DACE model parameters the image is represented using the concept of kriging predictor model.

1.2 Aim and scope of the project

This project deals with surrogate modelling and our aim is to develop an approximate model for representing an image using the DACE surrogate model. The scope of the project is limited to surrogates for computer models only. Computer codes are black boxes i.e. usually no analytical expression is available for their output. Hence no a priori knowledge of the output variation with change in input is available. Computer codes are deterministic, i.e. there is no systematic, random, or human error involved in running a computer code. The DACE surrogate model mentioned above creates an approximation from a given set of samples for such deterministic black box functions. As shown in Figure (1.1) the original computer code has the information of the true colors of the illuminated object through imaging system, but the surrogate model gets information about the output of the code only by sampling it. It must be noted that surrogate modelling involves creating a model for the code output without any prior knowledge of the output variation. Hence sampling the model intelligently as well as choosing the surrogate model framework is the key to obtaining a good approximation. We shall elaborate on these key issues further in the report.

Images can be represented as gray level images as well as color images. For gray level images the analysis is based on the spatial distribution of the pixel intensities. The color images are represented using intensities measured through several color filters (normally three, RGB) for each pixel. Image representation has increased drastically by the developing of more powerful computer models and also advances in computer capacity. Despite the advances in computer capacity, the enormous computational cost of complex engineering simulations makes it impractical to rely exclusively on simulation for the purpose of

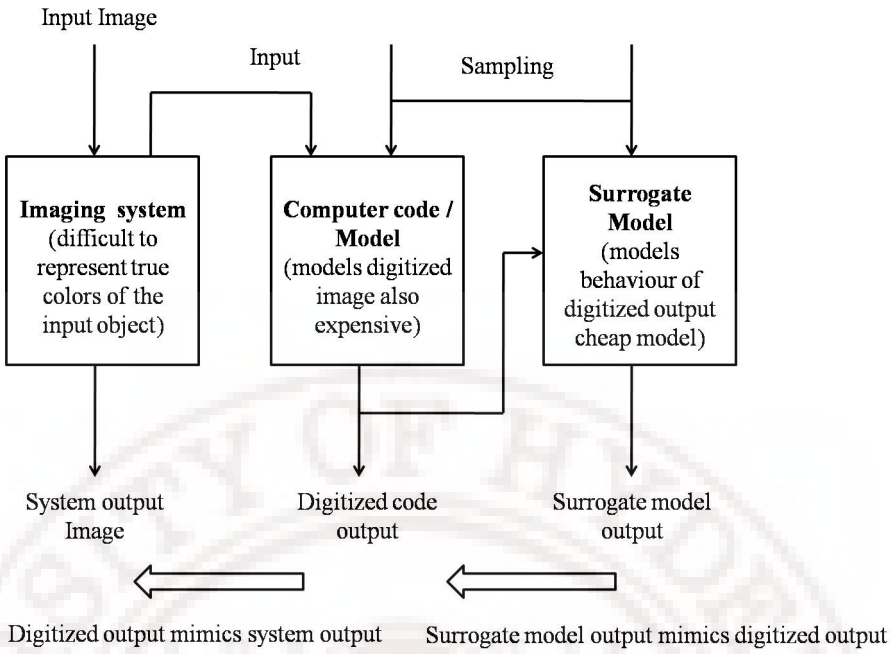


Figure 1.1: Surrogate Modelling Philosophy

designing models for representation of an image. To cut down the cost, surrogate models, also known as meta-models, are constructed from and then used in lieu of the actual representation models. A need has now arisen to replace expensive computer models with alternative cheap surrogates. These surrogates are approximate models which replace the behaviour of the original high fidelity code for representation.

1.3 Layout of the project

The layout of this report is as follows. In this chapter a brief introduction about the project and the motivation factor this work is discussed followed by aim and scope of the project. In the next chapter ie chapter 2 the entire concepts of DACE are covered including the mathematical derivations for building the model. In chapter 3 different image representation techniques, formats and interpolations are discussed. In chapter 4 it discusses the algorithm for representing image in DACE model describing each and every step involved for building the model. It also gives flow chart and later it explains the model representation for image. The chapter 5 discusses about demonstrating DACE model for image representation specifying few constraints and describes about the changes occurred when parameters are varied. The next chapter talks about the comparative study done on DACE representation describing the memory requirements and constraints. At the end conclusion which gives the final tool for DACE Image Representation followed by bibliography.

Chapter 2

DACE

In this chapter we discuss about the concept involved in building DACE analysing the mathematical concepts and deriving them by kriging interpolating technique and also discuss about the initial conditions taken before modelling. The objective of the surrogate model, as shown in Figure (1.1) of Chapter 1 is to predict the values of deterministic function y(x) , x $\in$ R^n, y $\in$ R^q over a variable-space D , D $\subset$ R^n when its values are known only at a limited, finite number of sites contained in S = {s_1,s_2,...,s_m |s_i $\in$D} . To make a good surrogate model we are free to choose S based on what type of input we are taking. Finding these design sites and a suitable surrogate model is the objective of DACE. According to [1] the problem of creation of a surrogate model for a computer experiment can be subdivided into 2 parts:

1. The Design problem: At which input sites in S = {s_1,s_2,...,s_m |s_i $\in$D}should the output data $y(s_1), y(s_2), ..., y(s_m)$ be collected?

2. The Analysis problem: How should the data be used to make a surrogate model that will help us predict y(x) data at untired sites with reasonable accuracy?

Note that this methodology is applicable only when y(x) is a deterministic function.

The surrogate model employs an interpolation scheme known as kriging developed in the fields of spatial statistics and geostatistics. This class of interpolating model has the flexibility to model response data for computer experiments. However, this flexibility is obtained at an increase in computational expense and a decrease in ease of use. The intent of this study is to provide an initial exploration of the accuracy and modelling capabilities of this approximation method.

2.1 Introduction

Despite the advances in computer capacity, the enormous computational cost of complex engineering simulations makes it impractical to rely exclusively on simulation for the purpose of designing computer experiments. To cut down the cost, surrogate models, also known as meta-models, are constructed from and then used in lieu of the actual computer experimental models. Various techniques for the construction of surrogate models, often also referred to as meta-models or approximation models, have been used in designing these models. Among these techniques, polynomial regression, Artificial Neural Network (ANN), Radial Basis Function (RBF), and Kriging (also referred to as Gaussian Process (GP) or Design and Analysis of Computer Experiments (DACE) models) are among some of the most prominent and commonly used techniques and in this chapter DACE model is discussed in detail.

The use of approximation models, or surrogates, has grown in popularity, a variety of modelling methods have been employed. Perhaps the most popular techniques involve polynomial models, typically linear or quadratic functions, created by performing a least squares curve fit to a set of data, where the data consist of one or more dependent response values along with one or more independent variables. Collectively, these polynomial-based modelling methods have come to be known as response surface models where the term taken from the statistical literature. These methods are popular for a number of reasons one of which is that they provide a compact and explicit functional relationship between the response and the independent variables. In addition, the method of least squares used in creating the models is relatively computationally inexpensive and straightforward.

Originally these polynomial modelling methods were developed to produce smooth approximation models of response data contaminated with random error found in typical physical (stochastic) experiments. Due to the ease of use of the polynomial modelling methods, these techniques migrated to the field of deterministic computer experiments where there is no random error (i.e., response data are identical each time the experiment is repeated). By considering the interpolation techniques in the concept of computer experiments an approximate model is designed. The interpolation technique used is based on concept kriging originally developed in the fields of spatial statistics and geostatistics. (The term kriging was first used in the work of Matherton [2] who attributed the original development of the statistical techniques to mining engineer D. G. Krige.) Recent studies by researchers at Boeing including Frank [3] and Booker [4] have employed DACE modelling methods in engineering optimization problems. The DACE modelling methods differ from the polynomial modelling methods in that the DACE models can capture oscillatory (multimodal) response trends whereas quadratic polynomials are by definition uni-modal. However, the

flexibility offered by DACE modelling methods is offset by the lack of an explicit model function as well as an increase in computational expense over that incurred in polynomial modelling.

2.2 Background on DACE

Prior to a description of the mathematical underpinnings of the approximation modelling methods, it is useful to compare the philosophy of polynomial modelling methods to that of DACE interpolating methods. To simplify this comparison, the phrase RS model will henceforth refer to polynomial models created via least squares surface fitting and the term DACE model will refer to the class of interpolating models based on Bayesian statistics and kriging. Although both RS models and DACE models are approximations to the true, unknown response surface and as such are technically response surface models, the statistical literature tends to reserve the term response surface model for polynomial models. The phrase polynomial RS model will be used to reinforce this distinction. The construction of polynomial RS models or DACE interpolating models relies on the sampling of the design space at n_s unique locations in the design space to obtain response values for the objective function or the constraints. Here, the design space is defined by the upper and lower bounds on the vector x of nv independent variables, where

$$x = [x_1, x_2, ..., x_{n_v}] \tag{2.1}$$

The upper and lower bounds create a design space in the shape of an nv-dimensional cube which has ${2_v}^n$ vertices. Note that experimental error is not present when obtaining results from deterministic computer models. Thus, no information is gained from the repeated sampling of the same location in the design space. From the sampled data approximation models are constructed to describe the variation in the response(s) with respect to the n_v independent variables. Mathematically, the true underlying functional relationship is expressed as

$$y = f(x) \tag{2.2}$$

where y is the observed response and f(x) is the unknown function. In many engineering optimization problems the cost of computing the objective function or constraints is computationally expensive. For this reason, approximation models are employed in the optimization problem as surrogates for these expensive function evaluations. These approximation models are expressed as

$$\hat{y} = \hat{f}(x) \tag{2.3}$$

Polynomial RS models can be thought of as global" models in which all of the n_s observed values of the response are equally weighted in the fitting of the polynomial surface. At

an unsampled location in design space, x; response observations that are near to x (in the sense of Euclidean distance) have an equal influence on the predicted response, $\hat{f}(x)$ as do the response observations that are far from x: It may be argued that such a global model may not be the best approximator if the true unknown response has multiple local optima. In such a situation an approximation scheme having local" modelling properties may be more attractive, i.e., where $\hat{f}(x)$ is more strongly influenced by nearby measured response values and is less strongly influenced by those further away. Such local modelling behaviour is characteristic of interpolation models, of which DACE models are one particular implementation.

2.3 DACE Model

The objective here is to provide an introduction to the statistics and mathematics of DACE modelling. A detailed treatment of the statistical and mathematical methods involved in DACE modelling is explained briefly in the next section. Before addressing the principles underlying DACE modelling methods, it is useful at this point to introduce the term prior distribution which is often used in Bayesian statistics. A prior distribution refers to the probability density function which one assigns to a variable of unknown value before any experimental data on that variable are collected. The prior distribution is the mechanism in Bayesian statistics through which one applies past experiences, knowledge, and intuition when performing an experiment. However, one's choice of the prior distribution biases the interpretation of the experimental data. This intentional bias is the source of much controversy in the statistical community. In spite of the differences between classical statistics and Bayesian statistics, Berger emphasizes that both classical and Bayesian statistics have merit, and he provides examples where the same interpretations of experimental data are obtained using both methods. In the DACE literature the true, unknown function to be modelled is typically expressed as

$$y(x) = f(x) + Z(x) \tag{2.4}$$

where f(x) is a known function of x and Z(x) is a Gaussian random function with zero mean and with variance of σ^2 (i.e., the behaviour of Z(x) follows a normal or Gaussian distribution). The f(x) term in Equation 2.4 in some sense is a global" model for the entire design space based on the ns response observations, while the Z(x) term creates a localized" deviation from the global model. In much of the current DACE literature the term f(x) in Equation 2.4 is considered a constant and is indicated using the term β. Equation 2.4 then becomes

$$y(x) = \beta + Z(x) \tag{2.5}$$

The term β takes on different meanings depending on one's statistical point of view. From the perspective of the kriging approach used in DACE, β is an unknown constant to be estimated based on the n_s observed response values. The covariance matrix of Z(x) is expressed as

$$Cov[Z(x^{(i)}), Z(x^{(j)})] = \sigma^2 R[R(x^{(i)}, x^{(j)})] \tag{2.6}$$

where R is the correlation matrix, and R is the correlation function which is selected by the user. In Equation 2.6 i = 1....n_s and j = 1....n_s

Note that the correlation matrix R is symmetric with values of unity along the diagonal. A choice for R often found in the statistical literature, is an exponential correlation function

$$R(x^{(i)}, x^{(j)}) = exp[-\sum_{k=1}^{n} \theta_k \mid x_k^{(i)} x_k^{(j)} \mid^2] \tag{2.7}$$

where x_k is the vector of unknown correlation parameters. For this research only a single correlation parameter is used instead of a vector of correlation parameters. The scalar correlation parameter is denoted as θ. Thus, Equation 2.7 may be rewritten as

$$R(x^{(i)}, x^{(j)}) = exp[-\theta \sum_{k=1}^{n} \mid x_k^{(i)} x_k^{(j)} \mid^2] \tag{2.8}$$

The process by which a value for θ is estimated is given below.

Another term of interest is the correlation vector, r(x), between the response at a location, x and the $x^{(1)}$,.....$x^{(ns)}$ response values. The correlation vector is expressed as

$$r(x) = R(x, x^{(i)}) = [R(x, x^{(i)}), R(x, x^{(2)}), ..., R(x, x^{(n)})] \tag{2.9}$$

While Equation 2.4 represents the true, unknown function to be approximated, the computed (i.e., estimated) DACE model is given the symbol $\hat{y}(x)$ In statistical notation, this estimated DACE model is defined as

$$\hat{y}(x) = E(y(x) \mid y(x^{(1)}).....y(x^{(ns)})) \tag{2.10}$$

where the expression E(.) is the statistical symbol for the expected value of (.) and the expression E(A|B) is the expected value of A given the information B. The terms $y(x^{(1)})$.....$y(x^{(n_s)})$ are the ns observed values of the response, y(x) is the true response one is attempting to estimate, and $\hat{y}(x)$ is the actual estimate of the response (which one hopes is close to y(x)). This distinction between $\hat{y}(x)$ and y(x) is necessary so that the concept of mean squared error (MSE) may be introduced where

$$MSE = E(\hat{y}(x) - y(x))^2 \tag{2.11}$$

This is simply a measure of the amount of error between the DACE model, $\hat{y}(x)$, and the true model, y(x); at all locations, x; in the design space. Since the DACE model performs

interpolation there is no error between the DACE model and the true model at the ns sites where the values of the response are known. If MSE is minimized, $\hat{y}(x)$ becomes

$$\hat{y}(x) = \hat{\beta} + r^T(x)R^{-1}(y - \hat{\beta}f) \tag{2.12}$$

where $\hat{\beta}$ is unknown, and both r(x) and R depend on the unknown parameter θ. Note that the vector f has length ns with all entries equal to unity

$$f = [1.....1] \tag{2.13}$$

which is a result of the assumption that all of the variability in y(x) is accounted for in the Z(x) term.

The unknown parameter θ is found using maximum likelihood estimation as described by Booker et al [4]. In this approach, the values for $\hat{\beta}$ and the estimated variance, $\hat{\sigma}^2$ are obtained using generalized least squares as

$$\hat{\beta} = (f^T R^{-1} f)^{-1} f^T R^{-1} y \tag{2.14}$$

$$\hat{\sigma}^2 = \frac{(y - \hat{\beta}f)^T R^{-1}(y - \hat{\beta}f)}{n_s} \tag{2.15}$$

Note that Equations 2.14 and 2.15 implicitly depend on the correlation parameter θ. The maximum likelihood estimation of θ is reduced to a one-dimensional optimization problem with simple bounds of the form

$$max_{\theta \in R^1}(-1/2)[(n_s ln\sigma^2) + ln \mid R \mid] subjectto 0 \leq \theta \leq \infty \tag{2.16}$$

Thus, by solving this one-dimensional optimization problem the DACE approximation model $\hat{y}(x)$ is completely defined. Note that if Equation 2.7 were used (i.e., retaining a vector of correlation parameters), then the one-dimensional minimization problem becomes an n_v dimensional minimization problem.

2.4 Modelling and Prediction

Given a set of m design sites $S = [s_1 \ . \ . \ .s_m]^T$ with $s_i \in R^n$ and responses $Y = [\ y_1 ... y_m]^{\ T}$ with $y_i \in R_q$. The data is assumed to satisfy the normalization condition

$$\begin{aligned} \mu[S_{:,j}] = 0, \quad V[S_{:,j}, S_{:,j}] = 1, \quad j = 1, ..., n, \\ \mu[Y_{:,j}] = 0, \quad V[Y_{:,j}, Y_{:,j}] = 1, \quad j = 1, ..., q, \end{aligned} \tag{2.17}$$

where $X:, j$ is the vector given by the jth column in matrix X, and $\mu[.]$ and V[. , .] denote respectively the mean and the covariance.

In this project we adopt a model $\hat{y}$ that expresses the deterministic response $y(x) \in R^q$,

for an n dimensional input $x \in D \subset R^n$, as a realization of a regression model F and a random function (stochastic process),

$$\hat{y}_l(x) = F(\beta_{:,l}, x) + z_l(x), l = 1, ..., q \tag{2.18}$$

We use a regression model which is a linear combination of p chosen functions $f_j : R^n \rightarrow R$,

$$\begin{aligned} F(\beta_{:,l}, x) &= \beta_{1,l} f_l(x) + ...\beta_{p,l} f_p(x) \\ &= [f_1(x)...f_p(x)]\beta_{:,l} \\ &= f(x)^T \beta_{:,l} \end{aligned} \tag{2.19}$$

The coefficients $\beta_{k,l}$ are regression parameters.
The random process z is assumed to have mean zero and covariance

$$E[z_l(w)z_l(x)] = \sigma_l^2 R(\theta, w, x), l = 1, ...q \tag{2.20}$$

between z(w) and z(x), where σ_l^2 is the process variance for the lth component of the response and R(θ, w, x) is the correlation model with parameters θ. An interpretation of the model (2.18) is that deviations from the regression model, though the response is deterministic, may resemble a sample path of a (suitably chosen) stochastic process z. In the following we will focus on the kriging predictor for y.
First, however, we must bear in mind that the true value can be written as

$$y_l(x) = F(\beta_{:,l}, x) + \alpha(\beta_{:,l}, x) \tag{2.21}$$

Where α is the approximation error. The assumption is that by proper choice of β this error behaves like white noise in the region of interest, i.e., for x$\in$D.

2.4.1 Kriging Predictor

For the set S of design sites we have the expanded m x p design matrix F with $F_{ij} = f_j(s_i)$,

$$F = [f(s_i)...f(s_m)]^T \tag{2.22}$$

with f(x) defined in (2.19). Further, define R as the matrix R of stochastic-process correlations between z's at design sites,

$$R_{ij} = R(\theta, s_i, s_j), i, j = 1, ...m \tag{2.23}$$

At an untried point x let

$$r(x) = [R(\theta, s_i, x)....R(\theta, s_i, x)]^T \tag{2.24}$$

be the vector of correlations between z's at design sites and x.
Now, for the sake of convenience, assume that q =1, implying that $\beta = \beta_{:,1}$ and $Y = Y_{:,1}$, and consider the linear predictor

$$\hat{y}(x) = c^T Y \tag{2.25}$$

with c = c(x)$\in R^m$. The error is

$$\begin{aligned} \hat{y}(x)y(x) &= c^T Y y(x) \\ &= c^T(F\beta + Z)(f(x)^T\beta + z) \\ &= c^T Z z + (F^T c f(x))^T \beta \end{aligned} \tag{2.26}$$

where $Z = [z_1....z_m]^T$ are the errors at the design sites. To keep the predictor unbiased we demand that $F^T c - f(x) = 0$, or

$$F^T c(x) = f(x) \tag{2.27}$$

Under this condition the mean squared error (MSE) of the predictor (3.25) is

$$\begin{aligned} \psi(x) &= E[(\hat{y}(x) - y(x))^2] \\ &= E[(c^T Z - z)^2] \\ &= E[z^2 + c^T Z Z^T c - 2c^T Z z] \\ &= \sigma^2(1 + c^T R c - 2c^T r) \end{aligned} \tag{2.28}$$

The Lagrangian function for the problem of minimizing ψ with respect to c and subject to the constraint (2.27) is

$$L(c, \lambda) = \sigma^2(1 + c^T R c - 2c^T r) - \lambda^T(F^T c - f) \tag{2.29}$$

The gradient of (2.29) with respect to c is

$$L'_c(c, \lambda) = 2\sigma^2(Rc - r) - F\lambda,$$

And from the first order necessary conditions for optimality we get the following system of equations

$$\begin{bmatrix} R & F \\ F^T & 0 \end{bmatrix} \begin{bmatrix} c \\ \tilde{\lambda} \end{bmatrix} = \begin{bmatrix} r \\ f \end{bmatrix}, \tag{2.30}$$

where we have defined

$$\tilde{\lambda} = -\frac{\lambda}{2\sigma^2}$$

The solution to (2.10) is

$$\begin{aligned} \tilde{\lambda} &= (F^T R^{-1} F)^{-1}(F^T R^{-1} r - f), \\ c &= R^{-1}(r - F\tilde{\lambda}) \end{aligned} \tag{2.31}$$

The matrix R and therefore R^{-1} is symmetric, and by means of (2.25) we find

$$\begin{aligned} \hat{y}(x) &= (r - F\tilde{\lambda})^T R^{-1} Y \\ &= r^T R^{-1} Y - (F^T R^{-1} r - f)^T (F^T R^{-1} F)^{-1} F^T R^{-1} Y \end{aligned} \tag{2.32}$$

In Section 3 we show that for the regression problem

$$F\beta \approx Y$$

the generalized least squares solution (with respect to R) is

$$\beta^* = (F^T R^{-1} F)^{-1} F^T R^{-1} Y$$

and inserting this in (3.32) we find the predictor

$$\begin{aligned} \hat{y}(x) &= r^T R^{-1} Y - (F^T R^{-1} r - f)^T \beta^* \\ &= f^T \beta^* + r^T R^{-1}(Y - F\beta^*) \\ &= f(x)^T \beta^* + r(x)^T \gamma^* \end{aligned} \tag{2.33}$$

For multiple responses (q ¿ 1) the relation (2.33) hold for each column in Y , so that (2.33) holds with $\beta^* \in R^{p\times q}$ given by (2.32) and $\gamma^* \in R^{m\times q}$ computed via the residuals, $R\gamma^* = Y - F\beta^*$.

Note that for a fixed set of design data the matrices β^* and γ^* are fixed. For every new x we just have to compute the vectors $f(x) \in R^p$ and $r(x) \in R^m$ and add two simple products.

Getting an estimate of the error involves a larger computational work. Again we first let q =1, and from (2.28) and (2.31) we get the following expression for the MSE of the predictor,

$$\begin{aligned} \psi(x) &= \sigma^2(1 + c^T(R_c - 2r)) \\ &= \sigma^2(1 + (F\tilde{\lambda} - r)^T R^{-1}(F\tilde{\lambda} + r)) \\ &= \sigma^2(1 + \tilde{\lambda}^T F^T R^{-1} F\tilde{\lambda} - r^T R^{-1} r) \\ &= \sigma^2(1 + u^T (F^T R^{-1} F)^{-1} u - r^T R^{-1} r) \end{aligned} \tag{2.34}$$

where $u = F^T R^{-1} r - f$ and σ^2 is found by means of (2.7) below. This expression generalizes immediately to the multiple response case: for the lth response function we replace σ by σ_l, the process variance for lth response function. Computational aspects for maximum likelihood estimation are explained in the appendix 1 and appendix 2.

2.4.2 Regression Models

For DACE model polynomial regression plays an important role in building the function for given data. It acts as a prediction function that works on the basis of statistical methods. It is very fast, even when fed with large amounts of data. When such type of models are enabled they discard insignificant variables so that they have no influence on the prediction model. This means that there is no need to think about the independent variables that are to be selected for predicting the values. The Polynomial Regression function picks out the most important variables automatically. Multiple polynomial regression predicts the value of a dependent variable on the basis of n independent variables, each of which being expressed by a polynom of the mth degree.
The polynomial regression expression is written in below equation

$$Y = A + \Sigma f(X_i) + \varepsilon \tag{2.35}$$

where

$$f(X_i) = B_{i1}X_i^1 + B_{i2}X_i^2 + ... + B_{im}X_i^m$$

Y is the dependent variable
X_i is independent variables
A, B_i are unknown coefficients
ε is error term

By selecting a polynom of a certain degree for a given independent variable, the Polynomial Regression function determines the best fitting curve for the variables. The term $\Sigma f(X_i)$ gives expressions as shown below where one of the terms inside the braces is selected. The number of available terms depends on the polynomial degree.

$$f(X_i) = \left\{ \begin{array}{c} B_{i1} * X_i^1 \\ B_{i1} * X_i^1 + B_{i2} * X_i^2 \\ \vdots \\ B_{i1} * X_i^1 + B_{i2} * X_i^2 + \ldots + B_{im} * X_i^m \end{array} \right\}$$

The toolbox provides regression models with polynomials of orders 0, 1 and 2. This work is extended and applied for higher degree polynomials and the final tool box for

representing an image has the polynomial regression function extended to degree from zero to five.

2.4.3 Correlation Models

In this section correlation model is discussed which plays an important role in modelling DACE. The Matlab toolbox provides seven different choices which are explained in a tabular format in Table 2.1 and in the current project only few correlation functions are used for building the model. The properties of a Gaussian random process x(t) of a very special type, namely, one that has zero mean and the exponential correlation function

$$\Psi(\tau) = (x(t)x(t+\tau)) = \sigma^2 exp(-\alpha \mid \tau \mid) \tag{2.36}$$

for time lag τ . The constants σ^2 and α are, respectively, the variance and the inverse correlation time of the process. (The quantity σ itself is the standard deviation)

In toolbox we restrict our attention to correlations of the form

$$R(\theta, w, x) = \prod_{j=1}^{n} R_j(\theta, w_j - x_j), \tag{2.37}$$

i.e., to products of stationary, one-dimensional correlations. More specific, the tool-box contains the following 7 choices which are explained in a tabular format in Table 2.1. In the table d_j is calculated using this equation $d_j = w_j - xj$.

Name	$R_j(\theta, d_j)$	
EXP	exp(-$\theta_j \mid d_j \mid$)	
EXPG	exp(-$\theta_j \mid d_j \mid^{\theta_{n+1})}$),	$0 \leq \theta_{n+1} \leq 2$
GAUSS	exp(-$\theta_j d_j^2$)	
LIN	max{0, 1-$\theta_j \mid d_j \mid$ }	
SPHERICAL	1-1.5$\xi_j + 0.5\xi_j^3$,	$\xi_j = min\{1, \theta_j \mid d_j \mid\}$
CUBIC	1-3$\xi_j^2 + 2\xi_j^3$,	$\xi_j = min\{1, \theta_j \mid d_j \mid\}$
SPLINE	$\zeta(\xi_j)$, (3.39)	$\xi_j = \theta_j \mid d_j \mid$

Table 2.1: Correlation functions

Some of the choices are illustrated in Figure 2.1 below. Note that in all cases the correlation decreases with $\mid d_j \mid$ and a larger value for θ_j leads to a faster decrease. The normalization (2.17) of the data implies that $\mid s_{ij} \mid \leq 1$ and therefore we are interested in cases where $\mid d_j \mid \leq 2$ as illustrated in the figure.

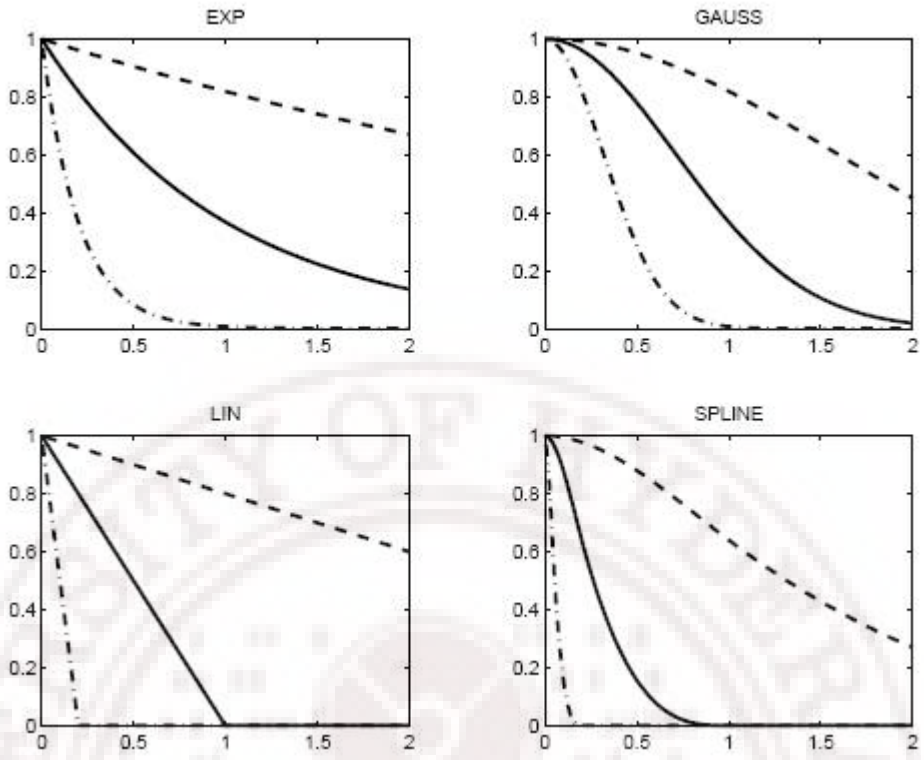


Figure 2.1: Correlation functions for $0 \leq d_j \leq 2$.Dashed, full and dash dotted line: $\theta_j =$ 0.2,1,5

The correlation functions in Table 2.1 can be separated into two groups, one containing functions that have a parabolic behavior near the origin (GAUSS,CUBIC and SPLINE), and the other containing functions with a linear behaviour near the origin (EXP,LIN, and SPHERICAL). The general exponential EXPG can have both shapes, depending on the last parameter: $\theta_{n+1} = 2$ and $\theta_{n+1} = 1$ gives the Gaussian and the exponential function, respectively.

The choice of correlation function should be motivated by the underlying phenomenon, e.g., a function we want to optimize or a physical process we want to model. If the underlying phenomenon is continuously differentiable, the correlation function will likely show a parabolic behaviour near the origin, which means that the Gaussian, the cubic or the spline function should be chosen. Conversely, physical phenomena usually show a linear behaviour near the origin, and EXP,EXPG,LIN or SPHERICAL would usually perform better. Also note, that for large distances the correlation is 0 according to the linear, cubic, spherical and spline functions, while it is asymptotically 0 when applying the other functions.

Often the phenomenon is anisotropic. This means that different correlations are iden-

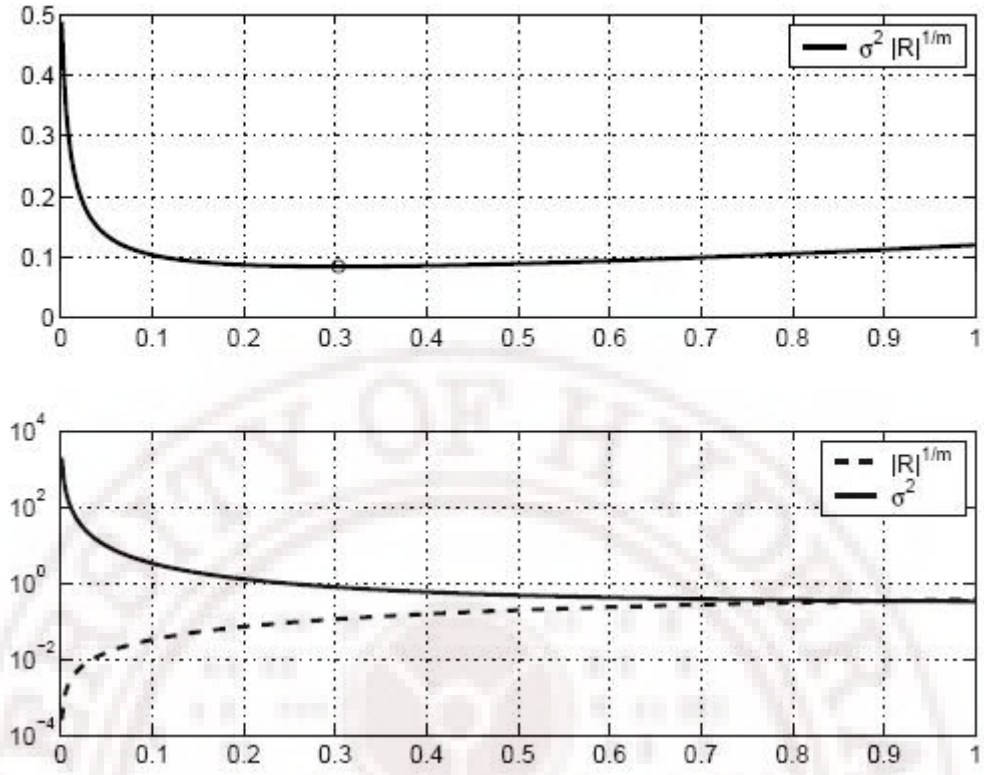


Figure 2.2: Typical behaviour of ψ , $\mid R \mid^{\frac{1}{m}}$ and σ^2 for o ¡ $\theta_j \leq 1$

tified in different directions, i.e., the shape of the functions in Figure 2.1 differ between different directions. This is accounted for in the functions in Table 2.1, since we allow different parameters θ_j in the n dimensions.

Assuming a Gaussian process, the optimal coefficients θ^* of the correlation function solves

$$min_\theta\{\psi(\theta) = \mid R \mid^{\frac{1}{m}} \sigma^2\} \tag{2.38}$$

where $\mid R \mid$ is the determinant of R. This definition of θ^* corresponds to maximum likelihood estimation. In Figure 3.2 we illustrate the typical behaviour of the functions involved in (2.40).

Note that $\mid R \mid^{\frac{1}{m}}$ is monotonously increasing in the interval. This is in accordance with expectation, since R_θ is close to the unit matrix for large θ, while it is indefinite for small θ. For $\theta = 0$, R is the matrix of all ones, which has rank one. In case of an indefinite R we define $\sigma^2 = \psi(\theta) = \infty$.

Thus building DACE model is discussed expressing in mathematical form and in the next chapter it describes about different image formats and interpolations are discussed.

Chapter 3

Image Representations

This chapter covers the existing image representations by studying about the image formats and also concentrating on image interpolation methods that are currently in use for different applications. Image is mathematically represented as two dimensional function with spatial coordinates with static amplitude called as pixels at respective pair of coordinates. The interest is to focus on procedures for extracting from image information in a form suitable for computer processing. Often, this information bears little resemblance to visual features that we use in interpreting the content of an image. Examples of the type of information used in computer experiments are statistical moments, Fourier transform coefficients, and multidimensional distance measures. The type of information for DACE representation is spatial and static in nature. This chapter has two sections which talks about the image formats and interpolation techniques that are used currently for image processing applications.

3.1 Image Formats

Image file formats are standardized means of organizing and storing images. Image files are composed of either pixel or vector (geometric) data that are rasterized to pixels when displayed in a vector graphic display. The pixels that compose an image are ordered as grid (columns and rows); each pixel consists of numbers representing magnitudes of brightness and colour.

3.1.1 Image file sizes

Image file size expressed as the number of bytes increases with the number of pixels composing an image, and the colour depth of the pixels. The greater the number of rows and columns, the greater the image resolution, and larger the file. Also, each pixel of an image

increases in size when its colour depth increases hyphen an 8-bit pixel (1byte)stores 256 colours, a 24 bit pixel (3 bytes) stores 16 million colors known as truecolor.
Image compression uses algorithms to decrease the size of a file. High resolution cameras produce large image files, ranging from hundreds of kilobytes to megabytes, per the camera's resolution and the image - storage format capacity. High resolution digital cameras record 12 megapixel (1MP = 1,000,000 pixels / 1 million) images, or more, in truecolor. For example, an image recorded by a 12 MP camera; since each pixel uses 3 bytes to record truecolor, the uncompressed image would occupy 36,000,000 bytes of memory hyphen a great amount of digital storage for one image, given that cameras must record and store many images to be practical. Faced with large file sizes, both within the camera and a storage disc, image file formats were developed to store such large images. An overview of the major file formats are discussed in next sections.

3.1.2 BMP format

The **BMP** is one of the simplest formats. It was jointly developed by Microsoft and IBM, which explains why it is particularly widespread on Windows and OS/2 platforms. A BMP file is a bitmap file, i.e. a graphic image file, with pixels stored in the form of point table and managing the colours either as true colours or using an indexed palette. The BMP format has been studied in such a way as to obtain a bitmap that is independent of the peripheral display device (DIB, Device independent bitmap). The structure of a bitmap file is the following:

- File header
- Bitmap information header (also called information Header)
- Palette
- Image coding

This gives brief idea on BMP image format and in next section TIF format is discussed.

3.1.3 TIF format

The TIF format (Tagged Image File Format) is a bitmap (raster) graphic file format. It was developed in 1987 by Aldus (now belonging to Adobe).

Characteristics of TIF format

The TIFF format is an old graphic format, which makes it possible to store very large (more than 4 GB compressed) bitmap images (raster) losing quality and regardless of the platforms or the peripherals used (Device-Independant Bitmap, which is written as DIB).

The TIFF format makes it possible to store images in black and white, true colours (up to 32 bits per pixel) as well as indexed images, using a palette. Moreover, the TIF format allows several colour spaces to be used:

- RGB
- CMYK
- CIE L*a*b
- YUV/YCrCb

structure of TIF file
The principle of the TIF format consists in defining tags (hence the name Tagged Image File Format) describing the characteristics of the image.
The tags make it possible to store information regarding the image dimensions, the number of colours used, the type of compression or the gamma correction.
Thus, image description using tags makes software programming simple by making it possible to save in TIFF format. On the other hand, the amount of options is so vast, that many image readers supporting TIFF format do not integrate them all, so that it happens that an image saved using TIFF format may not be readable with another.

3.1.4 PNG format

The PNG format (Portable Network Graphics or Ping format) is a bitmap (raster) graphic file format. It was developed in 1995 in order to provide a free alternative to the GIF format, which is a proprietary format whose rights are held by Unisys (proprietor of the LZW compression algorithm), to whom all software publishers using this type of format are under obligation to pay royalties. Thus, PNG is also a recursive acronym for PNG Not GIF.
Characteristics of PNG format
The PNG format makes it possible to store images in black and white (a colour depth of 16 bits per pixel), true color (a colour depth of 48 bits per pixel), as well as indexed images, using a palette of 256 colours. Moreover, it supports alpha channel transparency, i.e. the possibility of defining 256 levels of transparency, while the GIF format only allows one colour of the pallet to be defined as transparent. It also has an interlacing function which makes it possible to show the image gradually. The compression offered by this format is a (lossless compression) 5 to 25Finally, PNG stores image gamma information, which makes a gamma correction possible and allows it to become display-device-independent. Error correction mechanisms are also stored in the file in order to guarantee its integrity.

3.1.5 JPEG format

The acronym JPEG (Joint Photographic Expert Group comes from the meeting held in 1982 of a group of photographic experts, whose main concern was to work on the ways to transmit information (still or animated images). In 1986, the ITU-T developed compression methods intended for fax sending. These two groups joined to create a joint photographic experts group (JPEG).
Unlike LZW compression, JPEG compression is a lossy, , which affords it one of the best compression ratios, despite the slight loss of quality (20: 1 with 25:1 without any significant loss of quality).
This compression method is much more effective for photographic images (comprising many different coloured pixels) than on geometrical images (unlike LZW compression) because the nuance differences due to compression are very visible in the latter.
The stages of JPEG compression are the following:

- Chrominance re-sampling, because the eye cannot distinguish chrominance differences within a square of 2x2 points
- Image division into blocks of 8x8 points, then the application of the DCT function (Discrete Cosine Transform, which breaks up the image into a sum of frequencies
- Quantification of each block, i.e., a loss coefficient is applied (which makes it possible to determine the size/quality ratio) which will cancel out or decrease high frequency values, in order to attenuate the details by intelligently going over the block with RLE coding (in a zigzag pattern in order to remove as many zero values as possible)
- Image encoding then compression using the Huffman method

File formats that save a flow coded in JPEG are actually called JFIF (JPEG File Interchange Format), but the term is usually deformed into JPEG file". It should be noted that there is a lossless form of JPEG coding. Although little used by the data-processing community in general, it is used especially for the transmission of medical images in order to avoid confusing artefacts (purely dependent on the image and its digitalization) with real pathological signs. Compression is thus much less effective (only factor 2).
These are few image formats that are discussed and in next section it describes about interpolation techniques.

3.2 Image Interpolation

Image interpolation is an important image processing operation applied in diverse areas ranging from computer graphics, rendering, editing, medical image reconstruction, to online image viewing. Image interpolation techniques are referred in literature by many terminologies, such as image resizing, image re-sampling, digital zooming, image magnification or enhancement, etc. Basically, an image interpolation algorithm is used to convert an image from one resolution (dimension) to another resolution without losing the visual content in the picture. Image interpolation algorithms can be grouped in two categories, non-adaptive and adaptive. The computational logic of an adaptive image interpolation technique is mostly dependent upon the intrinsic image features and contents of the input image whereas computational logic of a non-adaptive image interpolation technique is fixed irrespective of the input image features.

3.2.1 Introduction

We are in the midst of a visually enchanting world. Image processing technologies play central role to process visual information in order to make it suitable for multimedia applications and visual perception. In this era of Internet and multimedia communication, sizing and resizing of images, to make it particularly suitable for viewing, transmission, downloading, sharing, editing, and further processing, became very vital. Image interpolation became a versatile and widely used tool in image processing today because of its numerous applications in diverse areas ranging from computer graphics, rendering, editing, medical image reconstruction, to online image viewing to name a few. Image interpolation techniques are referred in literature by many terminologies, such as image resizing, image re-sampling, digital zooming, image magnification, image enhancement, etc. Basically, image interpolation algorithms convert or resize a digital image from one resolution (dimension) to another resolution without losing the visual content in the picture. Image interpolation is part of many commercial image processing tools or freeware graphic viewers such as Adobe Photoshop CS2 software [5], IrfanView [6], etc.

A static image is a two-dimensional spatially varying signal. This needs to be digitized in order to process it by a digital computer. In digital imagery, this two dimensional spatially varied signal is sampled based on Nyquist criteria and then intensity of each sampled point is quantized to discrete valued integer numbers. Each such sampled and digitized value is called a pixel value of the digital image. The resolution or dimension of the digital image is the number of sampled points of the two-dimensional signal. In case of color imagery, each pixel contains three intensity values representing the contribution of three primary colors Red, Green, and Blue to form the color of a particular pixel point in the image. As

a result, a color image has primarily three such two-dimensional planes representing red, green, and blue components of the image pixels.

When the image is interpolated from a higher resolution to a lower resolution, it is traditionally called image down-scaling or down-sampling. On the other hand, when the image is interpolated from a lower resolution to a higher resolution, it is referred as image up-scaling or up-sampling. Most of the image interpolation techniques in the literature have been developed by interpolating the pixels based on characteristics of local features such as edge information, nearest neighbour criteria, etc. Image interpolation techniques can be broadly categorized into two categories non-adaptive, and adaptive techniques. The principles of adaptive interpolation algorithms basically rely on the intrinsic image features or contents of the image and accordingly the computational logic is mostly dependent upon the intrinsic image features and contents of the input image. The non-adaptive algorithms do not rely upon the image features or its contents and the same computational logic is repeated in every pixel or group of local pixels irrespective of the image contents. The computation of the proposed technique is non-adaptive in nature.

The non-adaptive image interpolation algorithms widely used in the literature are nearest-neighbour replacement, bilinear interpolation, bicubic interpolation, and kriging. Image interpolation is commonly performed by linear filters. The well-known bilinear and bicubic interpolation algorithms use linear interpolation filters, which have a drawback that they tend to produce undesirable blurring in the interpolated images. To enhance the sharpness and edge performance of image interpolation, some interpolation methods have been introduced, such as Kriging method. This method uses a simple method to classify the local image structure in the filters aperture and computes the optimal linear filter for each class. Kriging method has a better performance on edges than common linear methods. This method uses a more complicated classification method based on a stochastic model and obtains a high resolution image by mixing the outputs of optimized linear filters with coefficients that depend on the classification. As an approach of non-linear filtering methods, kriging can be used for image interpolation. Some applications use DACE as function approximators.

3.2.2 Non-Adaptive Algorithms for Image Interpolation

In non-adaptive image interpolation algorithms, certain computations are performed indiscriminately to the whole image for interpolation regardless of its contents. In this section, we will discuss the popularly used non-adaptive image interpolation techniques which are typically used in commercial image processing tools or freeware graphic viewer such as Adobe Photoshop CS2 software [5] and IrfanView [6].

Nearest Neighbour Replacement

The simplest interpolation method is just to replace the interpolated point with the nearest neighbouring pixel. The only advantage of this approach is the simplicity and low computation. However, the resultant pixelization or blocky effect makes the image quality unacceptable for most high quality imaging applications.

Bilinear Interpolation

The bilinear interpolation can be considered as a weighted average of four neighbouring pixel values. As shown in Figure 3.1, the intensity value I(x, y) at the interpolated point

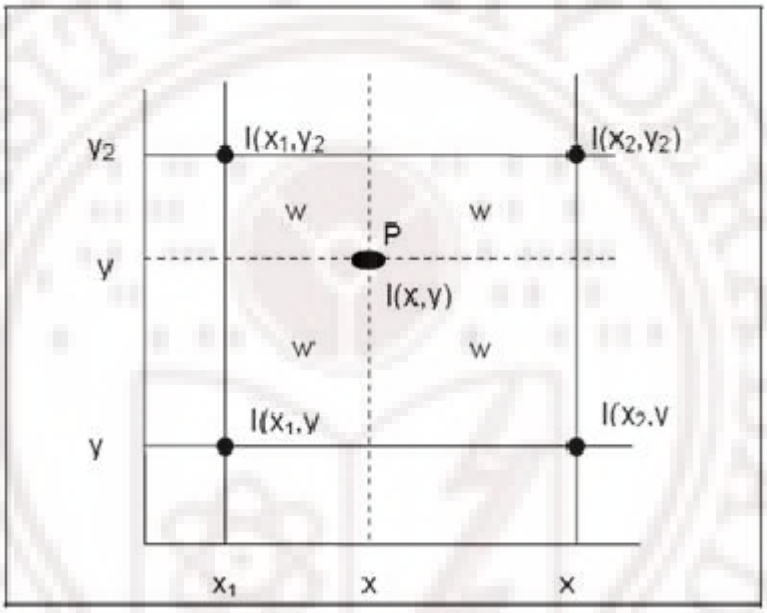


Figure 3.1: Bilinear interpolation using 4 neighbouring points

P at (x, y) in the image can be estimated as

$$\begin{aligned} I(x,y) &= \frac{(x_2-x)(y_2-y)}{(x_2-x_1)(y_2-y_1)}I(x_1,y_1) + \frac{(x-x_1)(y_2-y)}{(x_2-x_1)(y_2-y_1)}I(x_2,y_1) \\ &\quad + \frac{(x_2-x)(y-y_1)}{(x_2-x_1)(y_2-y_1)}I(x_1,y_2) + \frac{(x-x_1)(y-y_1)}{(x_2-x_1)(y_2-y_1)}I(x_2,y_2) \\ &= w_1 I(x_1,y_1) + w_2 I(x_2,y_1) + w_3 I(x_1,y_2) + w_4 I(x_2,y_2) \end{aligned} \tag{3.1}$$

where $I(x_1, y_1)$, $I(x_1, y_2)$, $I(x_2, y_1)$, and $I(x_2, y_2)$ are the intensity values of the 4 neighbouring pixels. From the above equation and Fig. 1, it is interesting to note that each weight is equivalent to the normalized area between the point of interest P and the diagonally opposite pixel. For example, (x_2, y_2) is diagonally opposed to (x_1, y_1) and weight for I $I(x_1, y_1)$ is w_1.

Bicubic Interpolation

The bicubic interpolation uses sixteen (4×4) neighboring pixels for estimation. It approximates the local intensity values using a bicubic polynomial surface. The general form for a bicubic interpolation is as follows:

$$\begin{aligned} I_{xy} &= \sum_{i=0}^{3}\sum_{j=0}^{3} a_{ij}x^i y^j \\ &= a_{00} + a_{10}x + a_{01}y + a_{20}x^2 + a_{11}xy + a_{02}y^2 + a_{21}x^2y \\ &\quad + a_{12}xy^2 + a_{22}x^2y^2 + a_{30}x^3 + a03y^3 + a13xy^3 + a_{32}x^3y^2 + a_{23}x^2y^3 + a_{33}x^3y^3 \end{aligned} \tag{3.2}$$

In order to do a bicubic interpolation within a grid square, one need to calculate the gradients (the first derivatives) in both the x and y directions and the cross derivative at each of the four corners of the square. This gives 16 equations that determine the 16 coefficients (a_{ij}), as explained in [14].

These are few image interpolation techniques that are being used on images for different applications such as image resizing, image re-sampling, digital zooming, image magnification or enhancement etc. In the previous chapter DACE model is discussed, which uses kriging technique for building the model and this concept can be applied as an interpolation technique for images by building a model and using the model interpolating the pixel values of the image. Thus this concept is used as an interpolating technique for images which are specified in spatial domain. The next chapter discusses the algorithm for representing image in DACE model describing each and every step involved for building the model. It also gives flow chart and later it explains the model representation for image.

Chapter 4

Image Representation: DACE Model

In the previous two chapters we discussed about modelling DACE and also existing image representation technique and this chapter combines both the chapters concepts and explains the entire framework how the DACE Model is modelled as representation for images. It explains the algorithm specifying each and every step involved in representation in different sections.

4.1 Initial Data

The first step includes specifying the initial conditions taken before starting the computer experiment. This is a crucial step which decides the field where the experiment is done. As we are dealing with Image Processing the data to be collected is from an input image. The input image can be considered as a gray image with size $M \times N$ or a color image with size $M \times N \times 3$ specifying the three colors RGB. The collection of input data points are the respective coordinates of an image which are considered as design sites in a spatial domain. The respective responses of these design sites in this experiment are the pixel elements taken at respective design site and they are considered as deterministic. For color image three responses for RGB are considered. The notations for design sites and responses are:

Design sites : $S = [s_1 \dots s_m]^T]$

Response : $Y = [y_1 \dots y_m]^T]$

Initial data are represented as matrix with dimension $S_{m\times 2}$ and response with dimension $Y_{m\times 1}$ for gray images and $Y_{m\times 3}$ for color images.

4.2 Normalization

For modelling, the design sites are normalized by satisfying the condition that mean is equal to zero and covariance is one. The normalized condition is shown in below equation

$$\begin{aligned} \mu[S_{:,j}] = 0, \quad V[S_{:,j}, S_{:,j}] = 1, \quad j = 1, ..., n, \\ \mu[Y_{:,j}] = 0, \quad V[Y_{:,j}, Y_{:,j}] = 1, \quad j = 1, ..., q, \end{aligned} \tag{4.1}$$

where X:, j is the vector given by the jth column in matrix X, and $\mu[.]$ and V[. , .] denote respectively the mean and the covariance.

4.3 Initial Parameters

The initial parameters to be considered include the values for the few parameters that are to be initialized before building the model. The initial parameters include the theta0 value, upper and lower bounds on theta0. The theta0 value is an initial guess for correlation function parameters. The upper and lower bounds are an optional and if present then they are upper and lower bounds on theta0 otherwise theta0 is used for theta.
The initial parameter values taken are:

$$\text{theta0} = [0.5\ 0.25]$$
$$\pi = [1\ 1]$$
$$\text{lob} = 0.1 \times \pi$$
$$\text{upb} = 1 \times \pi$$

The other input parameters include handle functions that are provided to build the model. The two handle functions are Polynomial Regression functions and Correlation functions. The polynomial regression function considered has been implemented with degree ranging from 0 to 5 and nearly seven different types of correlation functions are considered to build a DACE model. All the above sections deals about the inputs given for building the model.

4.4 Building DACE Model

The entire frame work for building the DACE Model considering the inputs as mentioned in above three sections is shown as a block diagram which explains clearly all the blocks involved for modelling. In the Figure 4.1 the high resolution image is represented as the signal **X** given as input is down sampled which gives an output as **S**. The sampling of the image entirely depends on how best the model is built with the required input samples of the image and also on the initial parameters taken before building the model. This

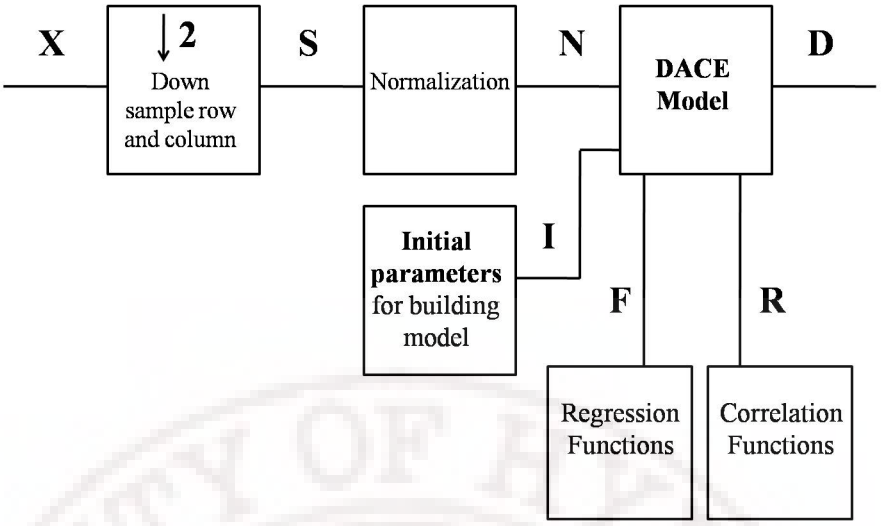


Figure 4.1: Block Diagram for building DACE model

sampled output is sent for normalization which gives **N** as output. The initial parameters to be considered before building model gives an output as **I** which is a combination of all initial values taken for modelling which are discussed in above sections.

To build DACE model four inputs are considered among which the first one includes the input sampled pixel values along with its respective coordinates that are sent into normalization block for satisfying the normalization condition which is been represented as **N** in the block diagram. The second input considered are the initial parameters required for building a model such as the theta value, upper and lower bounds which are represented as **I**. The third and fourth inputs that are provided for building DACE model include functions such as regression functions and correlation functions which are represented as **F** and **R** respectively. The polynomial regression function considered has been implemented with degree ranging from 0 to 5 and nearly seven different types of correlation functions are considered to build a DACE model. By considering all these different input parameters modelling is done which results giving a DACE model which gives an approximate model or surrogate for the given inputs. The output of DACE model is represented as **D** in the block diagram. This completes the modelling of the DACE model.

4.5 Evaluating the DACE Model

The purpose involved in this section is to use the DACE model and to evaluate it by giving the untried sites and estimating the responses using kriging predictor. The inputs to the kriging predictor are the trial sites at which the values are unknown and the DACE model which is built in the above section. The entire setup is shown in the below block diagram. The block diagram 4.2 shows two inputs for the kriging predictor or interpolator which

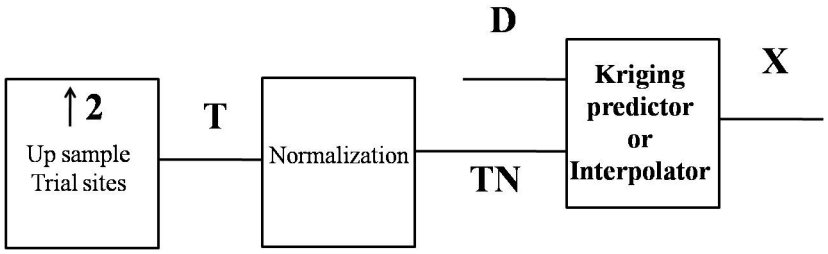


Figure 4.2: Block Diagram evaluating the DACE model

are represented as **D** and **TN** ie the DACE model and the untried trial sites respectively. The trial sites considered are denoted as **T** which is allowed to sent into normalization block which gives an output **TN**. By upsampling the unknown normalized trial sites are given as input to the predictor. The DACE model here is given as a structure and in turn the functions regression polynomial and correlation are also given as input to the predictor where the respective values are calculated for the two functions for trial sites. These inputs are not shown separately in the block diagram but given as a handle functions in the DACE model. By providing these inputs to the predictor model, the pixel values at the unknown sites are evaluated. The model is evaluated by specifying the required size of the image and is been represented using the fixed DACE model parameters. The estimation and the evaluation of the model is discussed briefly in the chapter 2 ie DACE.

4.6 Flow Chart

In all the above sections we discussed about the initial conditions, initial parameters, building the model by constructing a DACE model and then evaluating the model using the built DACE model. This section summarizes all the above steps which is shown as a flowchart. This gives a brief idea about what is done and how the image is represented using the DACE model and also shows the evaluation of the model using predictor. The flowchart begins with START and an input image is given as input. The input can be in any format and the respective pixel values are given to sampling stage where depending on the sample rate the pixels are collected and are given to next stage ie normalization where the coordinates of the image are normalized. Externally the input parameters and other functions are given to the DACE Model. The output parameters of DACE Model give a surrogate model for the given input. The next stage shown in the flow chart is the predictor stage for which the trial sites are calculated and are sent for normalization. This normalized trial sites along with DACE Model is given to the predictor model and an approximate pixel values of the image are predicted. The selection trial sites depends on the size required for the end user.

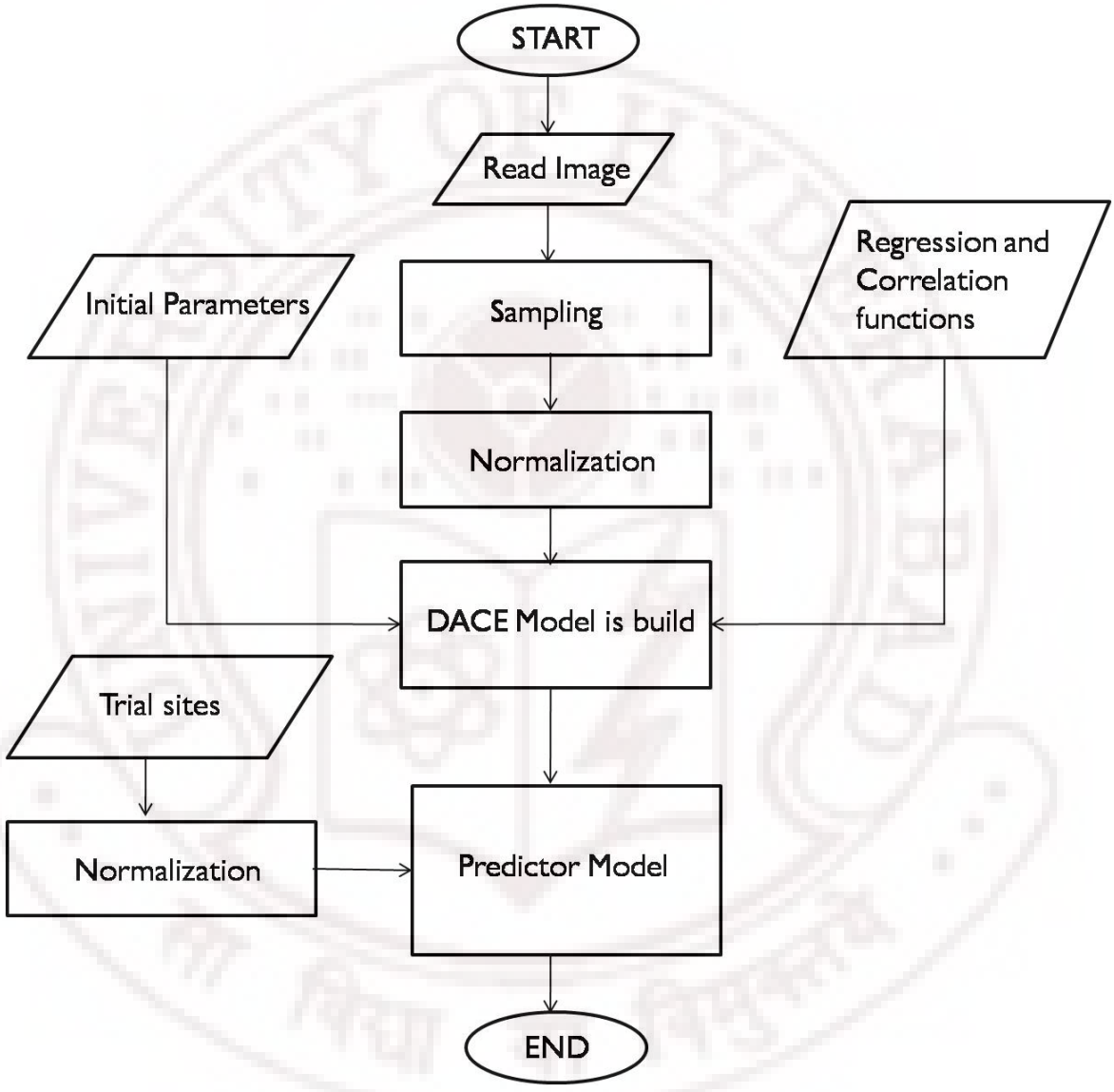


Figure 4.3: Flow Chart for DACE Approach for Image Representation

4.7 Model Representation

This section explains about the end equations for building model and gives the expression for parameters of the DACE model. The end expression gives an expression for representing an image. From the above sections we can get the initial values and also the initial parameters.

The given set of input parameters are:

Design sites : $S = [s_1 \dots s_m]^T]$ which are the coordinates of the image in spatial domain and is represented as a matrix with dimension $S_{m\times 2}$.

Response : $Y = [y_1 \dots y_m]^T]$ ie for images responses are the pixel elements or pixel values which are static and are considered as deterministic. For gray images the dimension of the response is $Y_{m\times 1}$ and when representing a color image response of dimension is considered as $Y_{m\times 3}$ denoting the three color spaces RGB(Red, Green and Blue). The other initial parameters and the functions are taken as mentioned in above sections.

Considering all these inputs DACE Model is built and the parameters that are calculated are $\beta, \gamma and \theta$ and these are calculated using the below respective equations:

$$\begin{aligned} \beta &= (F^T R^{-1} F)^{-1} F^T R^{-1} Y \\ \gamma &= R^{-1}(Y - F\beta) \\ \theta &= -1/2(dln\sigma^2 + lnR) \end{aligned} \tag{4.2}$$

where,

Regression function : $\mathbf{F} = A + Bs + Cs^2 + Ds^3 + \dots$

Correlation function : $\mathbf{R} = exp(-\theta_i * d_i^2)$, $d_i = (s_i - s_j)$

DACE Model equation is written as:

$$Y(x) = f^T(x)\beta + r^T(x)\gamma \tag{4.3}$$

where,

$$\begin{aligned} \text{Y (x)} &= \text{output response} \\ \text{f (x)} &= A + Bx + Cx^2 + Dx^3 + \dots \\ \text{r (x)} &= exp(-\theta_i * d_i^2), d_i = (x_i - x_j) \end{aligned}$$

The end equation expresses the image in the DACE Model that can be used as a new format for representing an image in a pixel based form considering in spatial and static domain.The next chapter discusses about the demonstration of DACE image representation specifying few constraints and describes about the changes occurred when parameters are varied.

Chapter 5

Demonstration of DACE Image Representation

This chapter describes about the demonstrating DACE model for representing an image using DACE Model. In previous chapter we have seen the mathematical notation for representing an image in DACE Model. In this chapter we shall observe the robustness of such type of representation when varying the parameters for building the model.

5.1 Gray Image Representation

For representing an image in gray initially the size of the image is fixed to $M \times N$ having M rows and N columns. The gray image can be written as matrix having spatial coordinates with respective image elements or picture elements or shortly known as pixel values. The gray input is given to different stages as discussed in previous chapter and the results of the important blocks are shown in the below figure. The figure shows gray input image with respective sample output image followed by it structured DACE model parameters and at the end it shows the approximate representation of the gray image which is predicted by considering DACE Model.

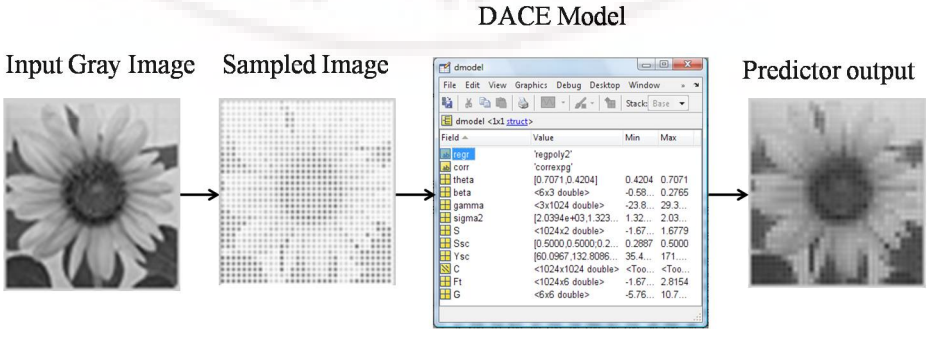


Figure 5.1: Steps for representing image in gray

5.2 Color Image Representation

For representing an image in color it requires higher dimensions nearly three to four times greater than the number employed for gray. The size of the color image is also fixed to $M \times N \times 3$ with M rows and N columns and the three specifies the three colors Red, Green and Blue (RGB) in the color model. The color image is also same as gray written as matrix having spatial coordinates with respective pixel values for all the three colors (RGB). The below figure explains the steps involved and the respective results at each stage are shown. In the figure it explains that color image is given as input which is then split into respective RGB color spaces and are sent for sampling separately and the color sampled output is shown as the next result. This sampled values are normalized and sent for building model along with other input parameters. the built DACE model predicts the respective RGB pixel values and then are combined for representing an approximate color image.

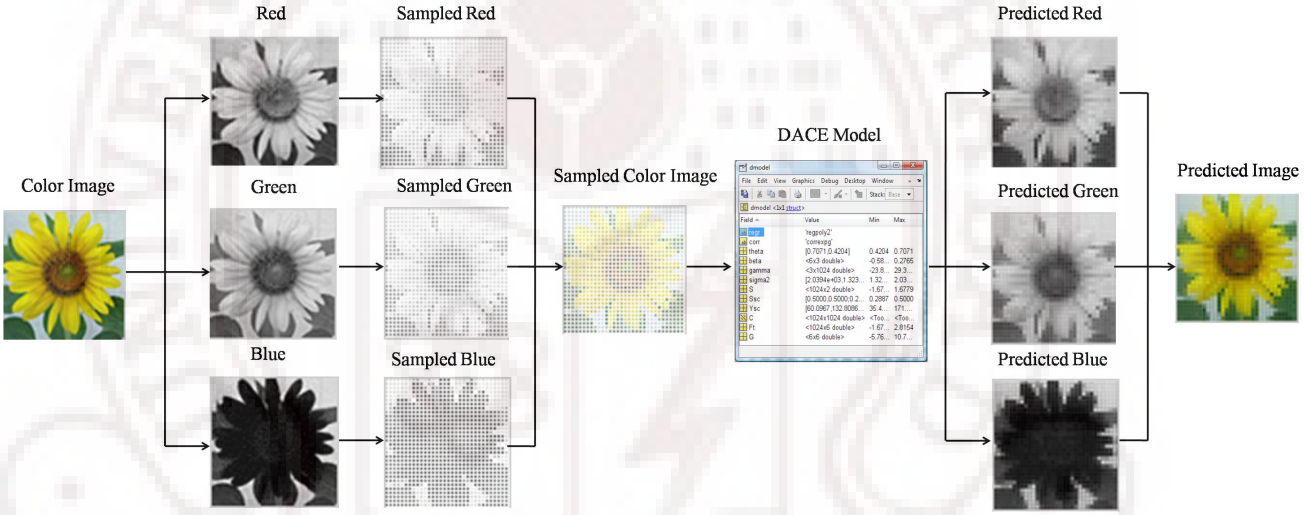


Figure 5.2: Steps for representing image in color

5.3 Representation for Zooming and Shrinking

In this section concentration is more on taking the trial sites that are to be predicted using fixed DACE model. The figure shows that input considered is color image and considering constant input parameters model is built and at the predictor side we can specify the size of the image to be represented by varying the trial sites given to predictor model. The size of the image can be resized by specifying the number of trial sites to be given to the predictor model. If the input image has to reduce its size then by specifying only few samples as trial sites to predictor model gives representation for smaller sized image. If we

want to zoom the input image then by specifying more trial sites to the predictor model gives a zoomed output. This is shown clearly in the figure specifying the fixed DACE model and predicting the zoomed or shrinked predicted outputs.

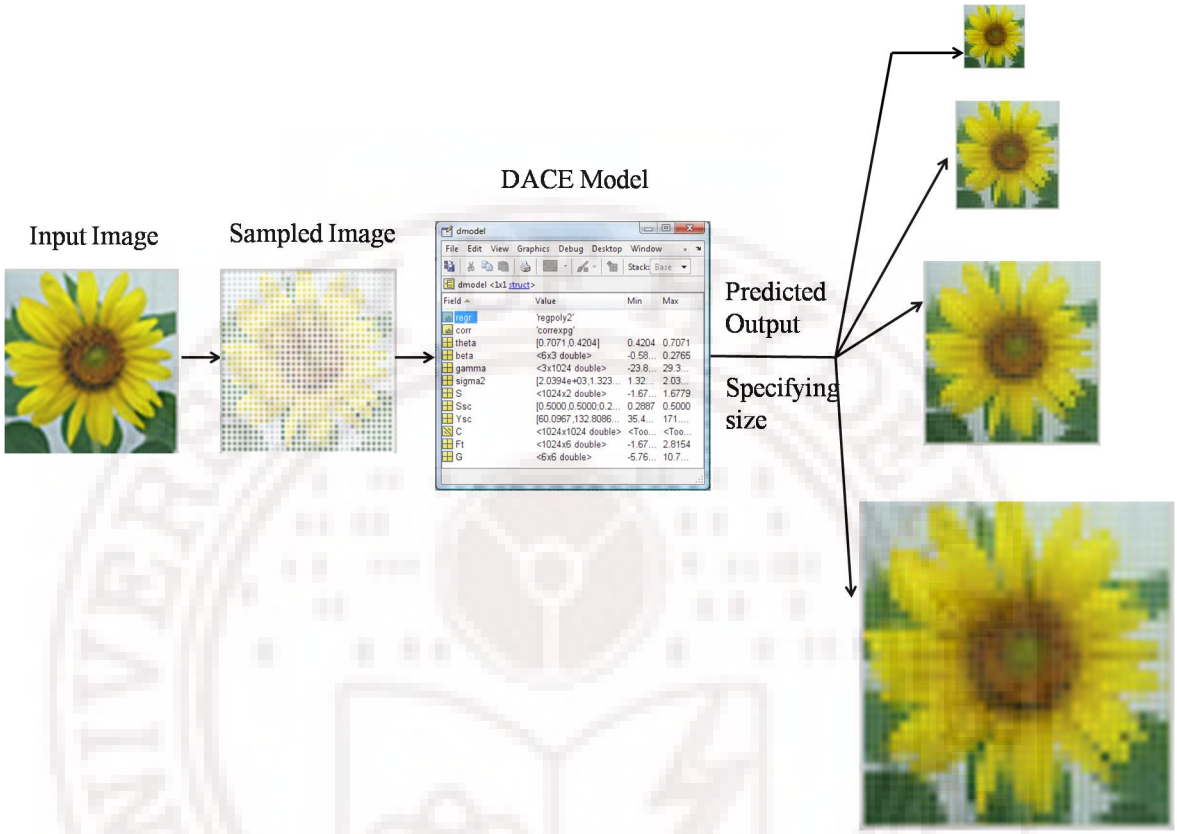


Figure 5.3: Predicted outputs for zooming and shrinking the input image using DACE Model

5.4 Representation by varying Sample Rate

This section concentrates more on how the image is represented by varying the sample rate at the input. The sampling is done depending on the sample rate taken at the input before given to the normalization block. The DACE model varies when sample rate is varied and thus the representation of image varies even though the same input is considered. This happens because when we specify the sample rate the sampling of the image ie picking the pixel values of image may varying and that leads to building different DACE model for same input image. The figure shows the predicted outputs when sampling rate is increased from 2 to 5. From the outputs we can observe that when sample rate is 2 ie considering more sample points for building model the predicted output resembles the input. As the sample rate increases then model is built with reduced number of sample points which

lead to represent the image with some loss. Thus it shows that for building DACE model selection of sample rate plays an important role.

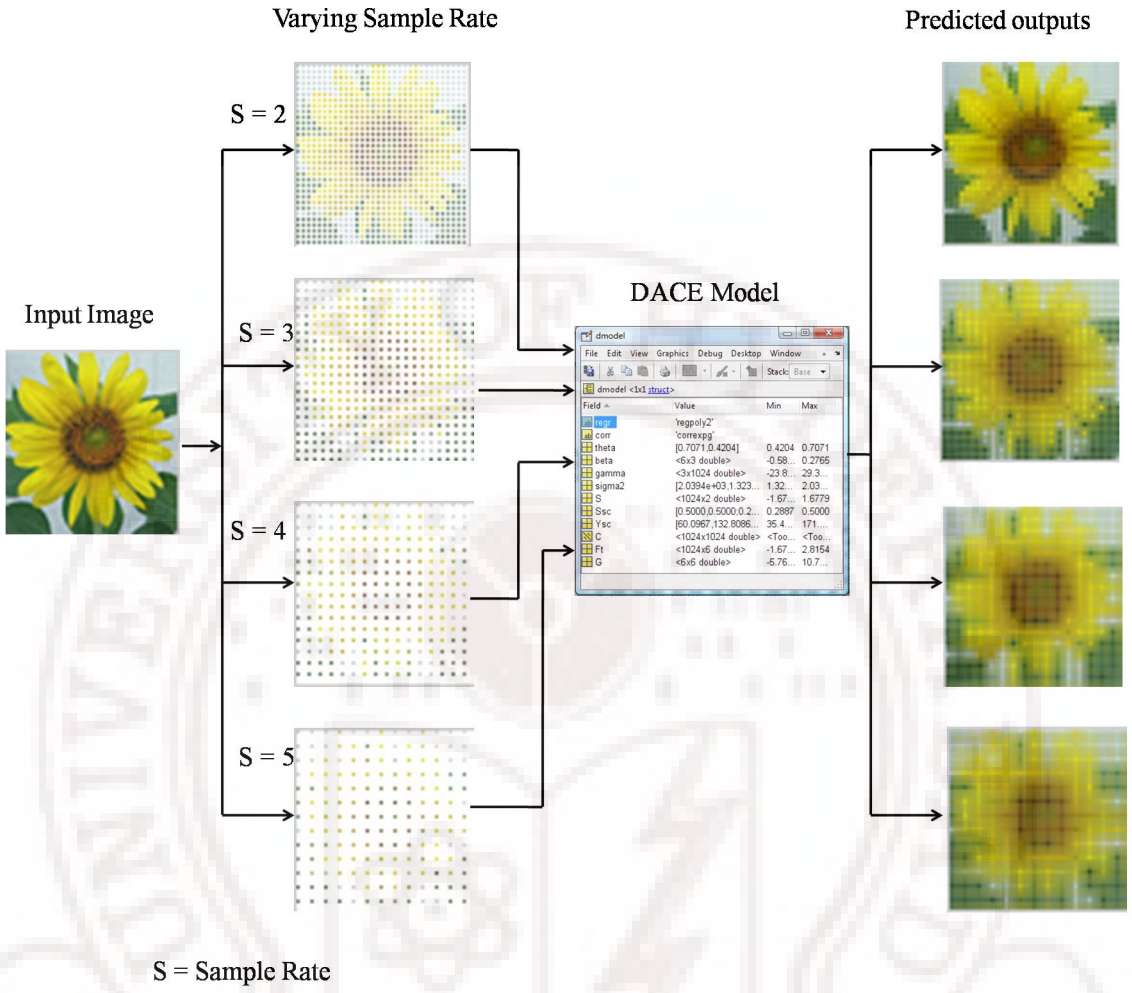


Figure 5.4: Results when sample rate is varied

5.5 Representation by varying Polynomial Regression

This section concentrates more on how the image is represented by varying the polynomial regression function parameters by varying the degree of the function. The concept of Regression function is discussed in DACE chapter giving the expression for the functions. The degree of the polynomial is varied from zero to five and DACE model is built. When the predicted outputs are observed it does not show much difference in representing an image built by varying regression function degree. This can be clearly observed in the below figure which shows the predicted image. The figure shows the predicted outputs when degree of polynomial regression is varied by fixing the other parameters. Thus it shows that while building DACE model selection of polynomial regression function degree

does not effect much on representing an image.

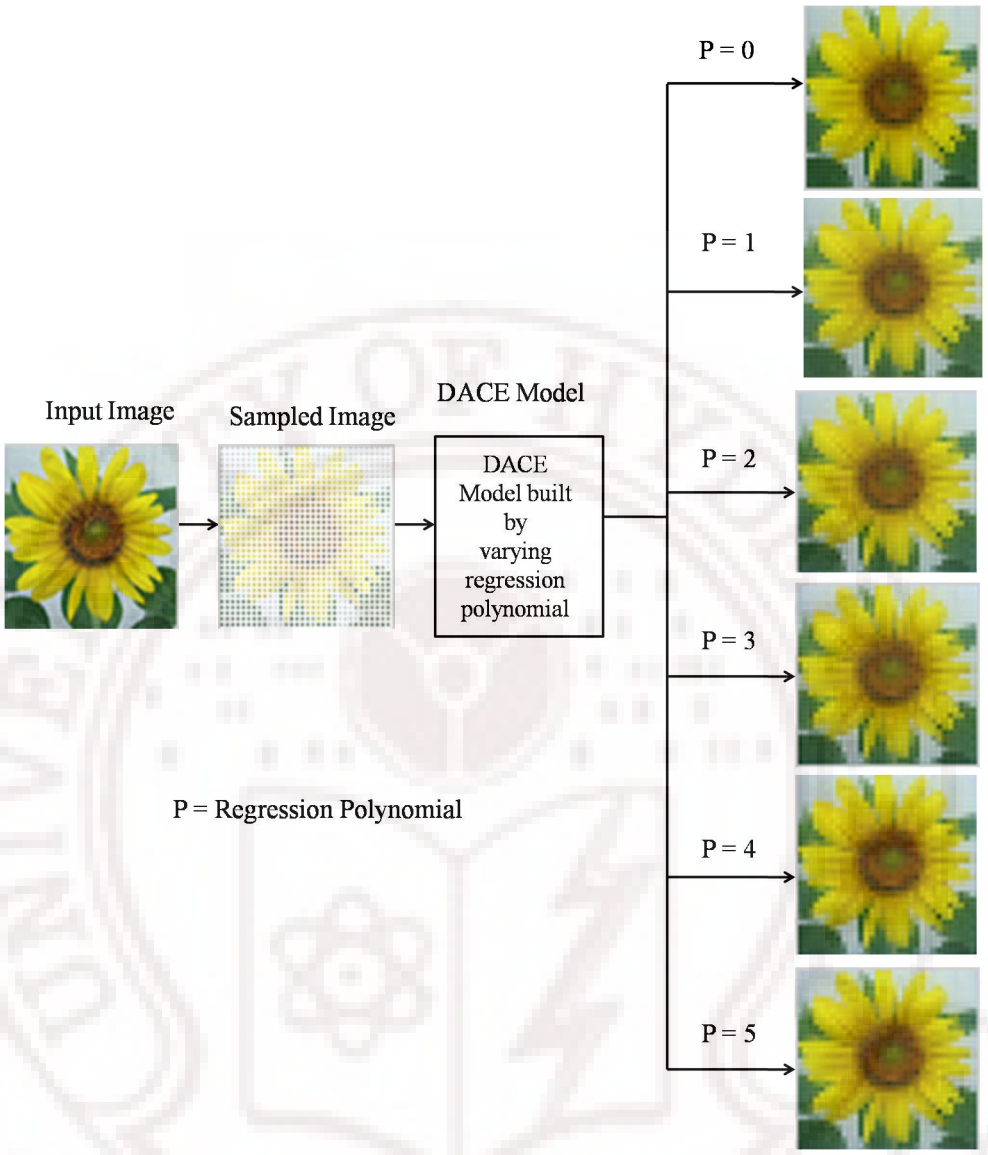


Figure 5.5: Results when degree of polynomial regression function is varied

5.6 Representation by varying both sample rate and polynomial regression

This section concentrates more on how the image is represented by varying the both polynomial regression function parameters by varying the degree of the function as well as varying the sample rate. In the above two sections it explains about varying these separately keeping any one fixed so this section gives a combination of both varying parameters. The figure shows the predicted outputs obtained at the predictor output. It shows the variation of degree of polynomial regression which is denoted as P varying from 0 to 5. It also

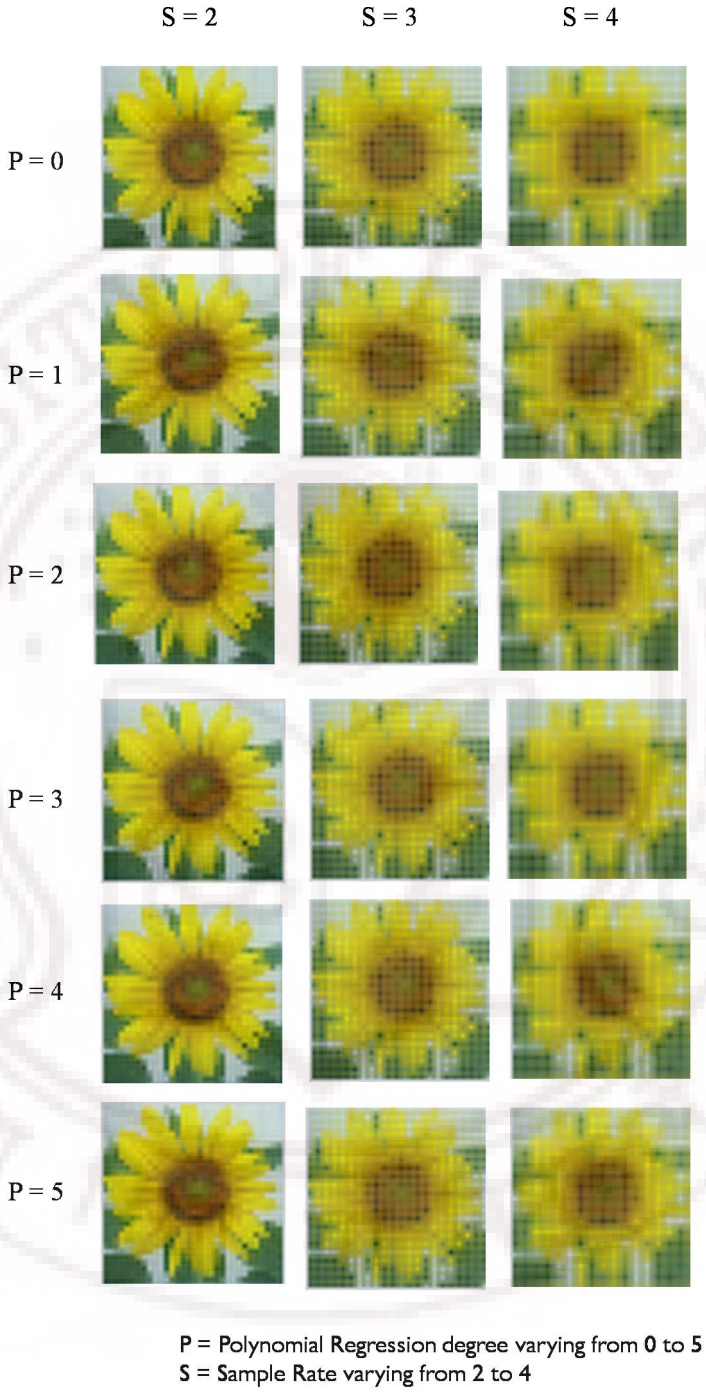


Figure 5.6: Predicted output results when both polynomial regression and sample rate are varied

gives the respective predicted output for varying sample rate S from 2 to 4. From the figure it clearly shows that as the degree of polynomial increases there is slight change in representation of image which is observable in the output so we can take the slight change as minor change in representation. While varying sample rate predicted output is differentiated from that of other outputs. Thus a comparative study is done in this section by varying few parameters.

The next chapter talks about the comparative study done on DACE representation describing the memory requirements and constraints.

Chapter 6

Comparative Study of DACE Representation

This chapter discusses a comparative study of DACE representation than compared with other representations. As we have seen in previous chapters that image is mathematically represented in DACE model and image is represented through a feature set which are the corresponding parameter values of the model. The memory required for storing the image with DACE model and comparing it with other image formats is discussed with an example. The specifications mentioned in example are considered as standardized tool for DACE Representation.

Given an input image of size $M \times N$ and sample rate(s) for sampling image is considered. The sample rate value depends on the type of input image considered ie when image has very few colors or shades in image then only few pixel values are required to represent such an image where as for representing an image which has more level of shades require more number of samples to be considered. The input parameters and initial conditions are same as discussed in previous chapters.

Representing an image of size $M \times N$ needs = M * N pixels for gray image
where as for representing a color image needs = M * N * 3 pixels
Memory required for representing image **I** = M * N (or) M * N * 3 in bytes
Memory required for representing in DACE **D** = (B + G + T) in bytes
where,

B = memory required for beta parameters in bytes
G = memory required for gamma parameters in bytes
T = memory required for theta parameters in bytes

Compression Ratio(CR) is calculated as

$$CR = \left[\frac{I-D}{I} * 100 \right] \tag{6.1}$$

Example

This is explained with an example taking an input image of size 64×64 for gray and 64×64×3 for color image. The example is explained for color image giving compression ratio for color image.

Input Image:

I = 64 * 64 * 3 = 12,288

DACE model parameters:

Memory required for β : **B** = 6 * 3 = 18

Memory required for γ : **G** = 1024 * 3 = 3072

Memory required for θ : **T** = 2 * 1 = 2

Memory required for DACE model parameters : **D** = B + G + T = 18 + 3072 + 2 = 3092

Compression Ratio :

$$\begin{aligned} CR &= \left[\frac{I-D}{I} * 100 \right] \\ &= \left[\frac{12{,}288-3092}{12{,}288} * 100 \right] \\ &= 74.83\% \end{aligned}$$

When representing an image DACE model parameters the compression ratio is nearly 75 % and the predicted output resembles the input hence this model can be used for representing an image and when the predicted output is after predicting all the pixel values can be represented into any image format and we cannot differentiate the predicted outputs represented in different image formats. This is observed in figures 6.1 and 6.2 where in the predicted outputs are represented in different image formats. In figure 6.1 it shows

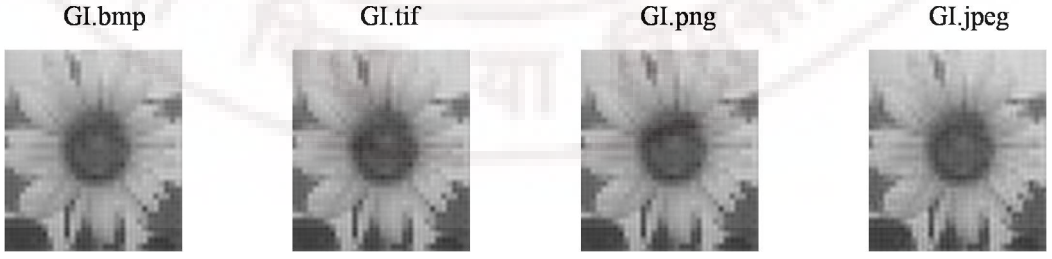


Figure 6.1: Predicted gray image outputs represented in different Image formats

the predicted outputs for gray input image ie after building the DACE model and then given to the predictor model where in the image pixel values are predicted using the model. In the figure the image is represented in four different formats ie in .bmp, .tif, .png and

.jpeg. There is no difference in representing them in different image formats expect the size required for representing them. In figure 6.2 it shows the predicted outputs for color

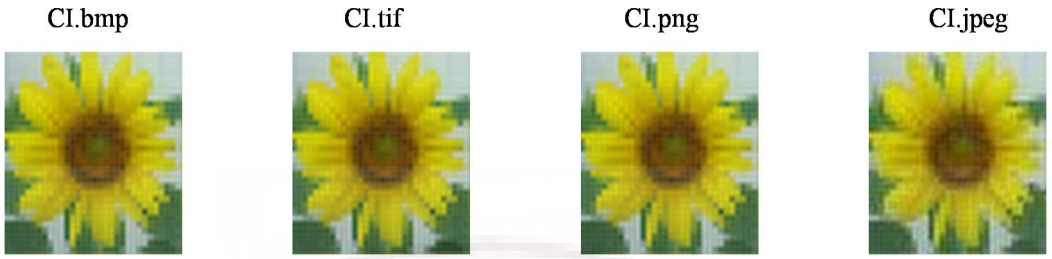


Figure 6.2: Predicted color image outputs represented in different Image formats

input image. They are predicted same as how gray image are predicted and the predicted outputs are represented in different image formats. The size of the input image is 64×64 with sample rate two and the polynomial regression function of degree two. This specification is same for images in Figure 6.1. There is no difference in representing predicted outputs in different formats except the size required for representing the image.
The memory required for representing in different image formats is discussed by considering the same input image as considered above and the memory required is to store in different image formats is explained in the below table. The above table 6.1 shows the

Table 6.1: Memory required for different image formats

Image Formats	Gray image memory (kb)	Color image memory (kb)
.bmp	6	13
.tif	5	13
.png	4	10
.jpeg	2	2

different image formats giving the respective memory required to store the image. The table gives the memory in kilobytes(kb) for the images which are shown in Figures 6.1 and 6.2 ie predicted outputs using DACE model. The memory required for representing DACE model for gray image is 1kb where as for color image it is 3kb. Once the parameters are calculated they can be used for different image processing applications. Instead of representing the entire pixel values of image and is represented as a surrogate model giving high accuracy for representing. Thus the concept DACE is used as a representation technique for images. This work is presented in an **"INTERNATIONAL CONFERENCE ON OPERATION RESEACH FOR GROWING NATION"** paper entitled **"Image Representation: DACE Approach"** held at 15 - 17 December 2008, presented by me

and P.Shalini . The next chapter discusses conclusion and gives the final tool for DACE Representation.

Chapter 7

Conclusion

The work mainly concentrated on developing and use of DACE as surrogate model for representing an image. it gives a naval representation methodology for image using DACE approach. Image is represented through a feature set which are corresponding parameter values of a DACE model. By building the model image is represented with fewer number of samples thus giving a good ratio for compression.
The merits involved in such type of representation are:

1. Representation accuracy is high, when considered few samples resulting in saving memory space for storage.

2. once the model is built the parameters of the model are used for zooming and shrinking of the image.

3. It is an effective representation in Web based service oriented systems.

The limitation involved in such type of representation is when size of input image is large then computation for building model increases. It is computationally intense though the work is done offline.
Final Tool for DACE Image Representation Input :
Consider input image of any format
Size : 64×64 (for gray)
$64 \times 64 \times 3$ (for color)
Sample rate : 2
theta : [0.5 0.25]
$\pi = [11]; lob = 0.1 * \pi; upb = 1 * \pi$
Regression function : degree varying from 0 to 5
Correlation function : EXPG (Exponential Gaussian)

Output :

We get predicted output in any formate

Size : To predict input image consider the size as

64×64 (for gray)

$64 \times 64 \times 3$ (for color)

To predict output of any size

$m \times n$ (for gray)

$m \times n \times 3$ (for color)

These are the specified specification for Final tool: DACE Image Representation. The scope of the project is to use DACE representation for image processing applications such as image retrieval, image matching, image verification, image feature analysis etc.

Bibliography

[1] T. J. M. Jerome Sacks, William J. Welch and H. P. Wynn, "Design and analysis of computer experiments," *Statistical Science*, vol. 4, no. 4, pp. 409–423, 1989.

[2] M. . G, "Principles of geostatistics," *Economic Geology*, no. 58, pp. 1296–1266, 1963.

[3] F. . P, "Global modeling for optimization," *SIAG / OPT Views-and-News*, vol. 7, pp. 9–12, 1995.

[4] Booker.A.J, "case studies in design and analysis of computer experiments," *in proceedings of the section on physical and Engineering Sciences*, 1996.

[5] "Adobe cs2 software." http://www.adobe.com/products/photoshop/.

[6] "Irfanview graphic viewer." http://www.irfanview.com/.

[7] S. N. Lophaven, H. B. Nielsen, and J. Søndergaard, "Dace - a matlab kriging toolbox," tech. rep., Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, 2002.

[8] T. J. Santner, W. B., and N. W., *The Design and Analysis of Computer Experiments*. Springer-Verlag, 2003.

[9] T. Acharya and P.-S. Tsai, "Computational foundations of image interpolation algorithms," *Ubiquity*, vol. 8, no. 42, pp. 1–17, 2007.

[10] R. C. Gonzalez and R. E. Woods, *Digital Image Processing (3rd Edition)*. Prentice Hall, August 2007.

[11] R. Jin and T. W. S. Wei Chen, "Comparative studies of meta-modelling techniques under multiple modelling criteria," *Structural and Multidisciplinary Optimization*, vol. 23, pp. 1–13, 2000.

[12] L. Etman, "Design and analysis of computer experiments: The method of sacks et al," Engineering Mechanics report WFW 94.098, 1994.

[13] A. Isaacs, *Direct-search methods and DACE*. PhD thesis, Department of Aerospace Engineering , Indian Institute of Technology Bombay, May 2003.

[14] D. Spinellis, "Coding for numbers," *IEEE Software*, vol. 22, pp. 95–96, January/February 2005. Book Review: Numerical Recipes in C++: The Art of Scientific Computing, 2nd edition.

Appendix1

Maximum Likelihood Estimation

The maximum likelihood estimation (MLE) that we shall consider is a parametric form of density estimation problem. Suppose $x_1, x_2, \cdots, x_m$ are independent and identically distributed (iid) with the common probability density function ψ. Suppose ψ is parameterized with respect to θ. The maximum likelihood estimation problem is to find θ such that conditional probability of $x_1, x_2, \cdots, x_m$ given θ, i.e. $P(x_1, x_2, \cdots, x_m \mid \theta)$ is maximum. Using that x_is are iid,

$$\theta_{MLE} = argmax(P(x_1, x_2, \cdots, x_m \mid \theta))$$

Since $x_i s$ are iid,

$$\theta_{MLE} = argmax\left(\prod_1^m p(x_i \mid \theta)\right)$$

$$\theta_{MLE} = argmax\left(\prod_1^m \psi(x_i, \theta)\right)$$

In our case, we frame an MLE problem to find all of θ,σ,β such that the joint probability distribution given by $\prod_1^m \psi\left(z(s_i), \theta, \sigma, \beta\right)$ is maximized, where $z(s_i) = y(s_i) - f(s_i)^T\beta$ is the error at site s_i appearing out of a stochastic process. The kriging model is a combination of multivariate normal model and a linear model. If the stochastic process is taken as Gaussian then the probability density function is given by

$$\prod_1^m \psi\left(z(s_i), \theta, \sigma, \beta\right) = \frac{1}{((2\pi)\sigma^2 det(R))^{m/2)}} exp\left[\frac{-(Y-F\beta)^T R^{-1}(Y-F\beta)}{2\sigma^2}\right]$$

Where R the correlation matrix is function of θ , $Y = [y(s_1), y(s_2), \cdots, y(s_m)]^T$ is vector of outputs at the chosen sites, and $F = [f(s_1), f(s_2), \cdots, f(s_m)]^T$ is an $m \times p$ matrix holding the regression functions evaluated at the chosen sites and β is as defined in Eq.(4.6). Hence MLE problem now is,

$$(\theta, \sigma, \beta)_{MLE} = argmax\left[\frac{1}{((2\pi)\sigma^2 det(R))^{m/2}} exp\left[\frac{-(Y-F\beta)^T R^{-1}(Y-F\beta)}{2\sigma^2}\right]\right]$$

Taking natural logarithms we get log likelihood problem

$$(\theta, \sigma, \beta)_{MLE} = argmax\left[\frac{-1}{2}\left[\text{m } \ln\sigma^2 + \ln \det(\text{R}) + \frac{(Y-F\beta)^T R^{-1}(Y-F\beta)}{\sigma^2}\right]\right]$$

To solve this we must differentiate the log likelihood with respect to θ, σ, and β and put the respective partial derivatives equal to 0.σ and R are independent of β so the MLE of β denoted by β^* is obtained as

$$\frac{\delta}{\delta\beta}\left((Y - F\beta)^T R^{-1}(Y - F\beta)\right) = 0$$

$$\beta^* = (F^T R^{-1} F)^{-1} F^T R^{-1} Y$$

This MLE β^* is the same as the generalized least squares estimate of the regression problem. Similarly, the MLE of σ^2, denoted by σ^{*2} is obtained as

$$\frac{\delta}{\delta\sigma_2}\left[\text{m } \ln\sigma^2 + \frac{(Y - F\beta^*)^T R^{-1}(Y - F\beta^*)}{\sigma^2}\right] = 0$$

$$\sigma^{*2} = \frac{1}{m}\left((Y - F\beta^*)^T R^{-1}(Y - F\beta^*)\right)$$

We see that parameters β and σ^2 are decoupled. In fact it is clear that both σ^{*2} and β^* are both essentially functions of θ. Thus the MLE problem posed above is in fact a problem of finding the MLE of only θ. Although the least squares solution for β and the MLE β^* are identical, this fact could not have been established by finding least squares solution first and then imposing MLE separately on θ.
On solving the MLE for θ, σ^{*2} and β^* get fixed as a consequence. This means that by choosing n critical values in the vector θ, the model and as we shall see in the next section, the predictor can be determined. This remarkable simplicity is attributed to the choice of the stochastic process as Gaussian. MLE for θ is reposed as

$$\theta_{MLE} = argmax\left[\frac{-1}{2}(\text{m } \ln\sigma^{*2} + \ln \det(\text{R})\right]$$

This is an optimization problem that has to be solved numerically. Having solved for θ and having found σ^{*2} and β^*we are in a position now to create a predictor for the function y(x) to predict its value over the domain D.

Appendix2

Deriving the BLUP using the method of Lagrange Multipliers

The BLUP (best linear unbiased predictor) requires that we solve:

1. Minimize MSE=$E(\mid \hat{y}(x) - y(x) \mid^2)$ with respect to c(x)
2. Subject to $E(\hat{y}(x))$ =E(y(x))

The mean square error at untried x is

$$MSE(x) = E\left\{\mid c(x)^T(F\beta^* + Z) - f(x)^T\beta^* - z(x) \mid^2\right\}$$

Where

$$Z = [z(s_1), z(s_2), \cdots, z(s_m)]^T$$

Since E(z)=E(Z)=0 the unbiasedness condition becomes $F^T c(x) = f(x)$

$$MSE(x) = \sigma^2\left(1 + c(x)^T Rc(x) - 2c(x)^T r(x)\right)$$

Where r(x) is a vector that holds the correlations between the untried x and sites in S . $r(x) = [\rho(s_1, x), \rho(s_2, x), \cdots, \rho(s_m, x)]^T$
Thus the constrained optimization problem stated above is to minimize MSE (x) subject to $F^T c(x) = f(x)$. This can be solved using the method of Lagrange multipliers. The Lagrange equation is with a vector Lagrange multipliers λ is

$$L(c, \lambda) = MSE(x) - \lambda^T(F^T c(x) - f(x))$$

$$\frac{\delta L}{\delta \lambda} = 0 \Rightarrow F^T c(x) - f(x) = 0 \& \frac{\delta L}{\delta c} = 0 \Rightarrow 2\sigma^2(Rc(x) - r(x)) - F\lambda = 0$$

$$c(x) = R^{-1}(r(x) + F(F^T R^{-1} F)^{-1}(f(x) - F^T R^{-1} r(x)))$$

Thus the BLUP can now presented as below. Using that R is symmetric,

$$\hat{y}(x) = r(x)^T R^{-1} Y - (F^T R^{-1} r(x) - f(x))^T (F^T R^{-1} F)^{-1} F^T R^{-1} Y$$

Title:

Sampling Strategies for Computer Experiments: Design and Analysis

Authors:

Timothy W. Simpson
Assistant Professor
The Pennsylvania State University
Departments of Mechanical & Nuclear Engineering
and Industrial & Manufacturing Engineering
University Park, Pennsylvania 16802

Dennis K. J. Lin
Professor
The Pennsylvania State University
Department of Management Science and Information Systems
University Park, PA 16802-1913

Wei Chen
Associate Professor
Department of Mechanical Engineering
University of Illinois at Chicago
Chicago, IL 60607-7022

Corresponding Author:

Dennis K. J. Lin
314 Beam Building
Penn State University
University Park, PA 16802
email: DKL5@psu.edu
phone: (814) 865-0377
fax: (814) 863-2381

Submitted to:

International Journal of Reliability and Applications

Submitted: December 18, 2000
Revised: August 14, 2001

Sampling Strategies for Computer Experiments: Design and Analysis

Timothy W. Simpson, Dennis K. J. Lin, and Wei Chen

ABSTRACT

Computer-based simulation and analysis is used extensively in engineering for a variety of tasks. Despite the steady and continuing growth of computing power and speed, the computational cost of complex high-fidelity engineering analyses and simulations limit their use in important areas like design optimization and reliability analysis. Statistical approximation techniques such as design of experiments and response surface methodology are becoming widely used in engineering to minimize the computational expense of running such computer analyses and circumvent many of these limitations. In this paper, we compare and contrast five experimental design types and four approximation model types in terms of their capability to generate accurate approximations for two engineering applications with typical engineering behaviors and a wide range of nonlinearity. The first example involves the analysis of a two-member frame that has three input variables and three responses of interest. The second example simulates the roll-over potential of a semi-tractor-trailer for different combinations of input variables and braking and steering levels. Detailed error analysis reveals that uniform designs provide good sampling for generating accurate approximations using different sample sizes while kriging models provide accurate approximations that are robust for use with a variety of experimental designs and sample sizes.

Keywords: design of experiment, kriging, Latin hypercube, multivariate adaptive regression spline, radial basis functions, response surface, uniform design.

NOMENCLATURE

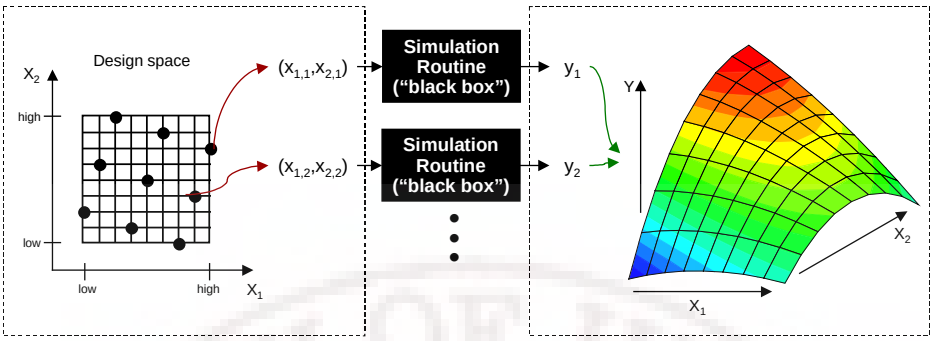
APPROX	approximation model type:
krg	kriging approximation
mar	multivariate adaptive regression splines
rbf	radial basis functions
rs2	second-order polynomial response surface
DOE	experimental design type:
hss	Hammersley sequence sampling
lhd	Latin hypercube
oay	orthogonal array
rnd	random design
uni	uniform design
FCN	function number for first example
MAX	maximum absolute error
RMSE	root mean square error
SAMP	number of sample points in an experimental design
x	design (input) variable
y	actual output (response) value
$\hat{y}$	predicted output (response) value from approximation model

1. FRAME OF REFERENCE: COMPUTER EXPERIMENTS

Computer-based simulation and analysis is used extensively in engineering to predict the performance of a system or product. For example, engineers use finite element models to predict the performance of a structure, computational fluid dynamics models to visualize the flow over a body, and Monte Carlo simulation to estimate the reliability of a product due to uncertainty in loading conditions or material parameters. Despite the steady and continuing growth of computing power and speed, single evaluations of aerodynamic or finite element analyses can take minutes to hours, if not longer. The high computational costs of performing these analyses limit their use in design optimization and reliability analysis.

Design of experiments (e.g., Montgomery, 1997) and statistical approximation techniques such as response surface methodology (e.g., Box and Draper, 1987; Box, et al., 1978; Myers and Montgomery, 1995) are becoming widely used in engineering to minimize the computational expense of running such computer analyses (Barthelemy and Haftka, 1993; Barton, 1992; Barton, 1994; Barton, 1998; Simpson, et al., 2001b; Sobieszczanski-Sobieski and Haftka, 1997). The basic approach is to construct approximations of the computationally expensive simulation and analysis codes to provide surrogate models that are sufficiently accurate to replace the original code. These surrogate models are then used in lieu of the original analysis or simulation code, facilitating design space exploration, optimization, and reliability analysis.

Building approximations for these computer simulations involves (a) choosing an experimental design to sample the region of interest and (b) constructing an approximation model to the observed sample data as shown in Figure 1. As shown in Figure 1a, the region of interest is often referred to as the "design space," which is bounded by the upper and lower limits of each of the design (input) variables being studied. Design of experiments strategies are often used to sample the design space to generate sample data to fit an approximate model to each of the output variables (responses) of interest. Experimental designs can also be used for "screening" experiments to identify significant factors and reduce the dimensionality of the problem (Box and Draper, 1987; Gangadharan, et al., 1995; Goldsman and Nelson, 1998; Koch, et al., 1997; Welch, et al., 1992). In Figure 1b, a second-order response surface is used to approximation the relationship between the design (input) variables x_1 and x_2, and y, the output variable (response).



(a) Sample Region of Interest (b) Construct Approximation Model

Figure 1. Design and Analysis of Computer Experiments

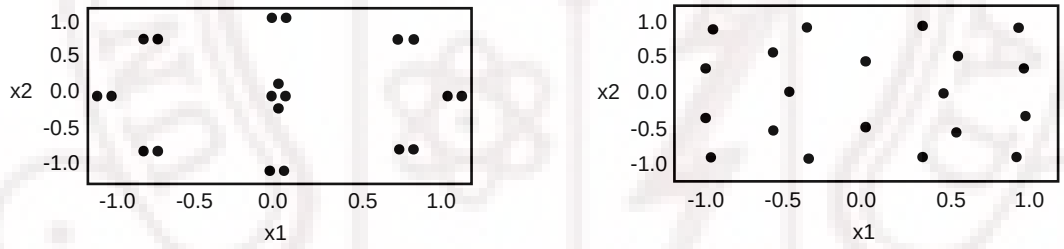
The growing use of computers in design optimization has given rise to considerable research in the design and analysis of computer experiments. The primary research thrusts are to improve:

1. the *efficiency* with which the design space is sampled either by using fewer sample points or seeking better coverage of the design space, and
2. the *accuracy* of the resulting surrogate model by using more complex approximations that are capable of fitting both linear and non-linear functions.

In this paper, we systematically compare several experimental design types and surrogate model types, which are widely used in the engineering design community, in terms of their capability to generate accurate approximations for computer experiments. In the next section, an overview of experimental design strategies for computer experiments is offered. This is followed in Section 1.2 with a summary of the different types of surrogate models that are being used for approximating computer experiments. Descriptions of the experimental design types and surrogate models employed in this study are given in Sections 2.1 and 2.2, respectively. Sections 3 and 4 contain two example problems wherein the different experimental design types and surrogate models are compared in terms of their capability to generate accurate approximations. The first example involves the design of a two-member frame subject to out-of-plane loading—it is a small three variable problem that is relatively inexpensive to analyze yet is characteristic of many engineering analyses used in structural optimization. The second example simulates roll-over of a semi-tractor-trailer for a given design and driving conditions—the computational expense and size of this example (14 variables) yields a large complex problem, involving a highly nonlinear response. Based on the results of these examples, recommendations for efficient and accurate approximation model building are given in Section 5.

1.1. Experimental Designs for Computer Experiments

Properly designed experiments are essential for effective computer utilization. Experimental design techniques, which were initially developed for physical experiments, are finding considerable use for the design of computer experiments to increase the efficiency of these analyses. In the "classical" design and analysis of physical experiments (i.e., using central composite and factorial designs), random variation is accounted for by spreading the sample points out in the design space and by taking multiple data points (replicates) as shown in Figure 2a. Sacks, et al. (1989) state that the "classical" notions of experimental blocking, replication, and randomization are irrelevant when it comes to *deterministic* computer experiments; thus, sample points should be chosen to fill the design space for computer experiments. Consequently, many researchers advocate the use of "space filling" designs when sampling deterministic computer analyses to treat all regions of the design space equally. For instance, Sacks, et al. (1989) suggest minimizing the integrated mean squared error (IMSE) over the design region by using an IMSE-optimal design as shown in Figure 2b.



(a) "Classical" Design (b) "Space Filling" Design

Figure 2. "Classical" and "Space Filling" Designs (Booker, 1998)

Koch, Mavris and Mistree (1998) investigate the use of a modified central composite design (CCD) that combines half-fractions of an inscribed CCD with a face-centered CCD to distribute points more evenly throughout the design space. Koehler and Owen (1996) describe several Bayesian and Frequentist "space filling" designs, including maximum entropy designs, mean squared-error designs, minimax and maximin designs, Latin hypercubes, randomized orthogonal arrays, and scrambled nets. Minimax and maximin designs were originally proposed by Johnson, Moore and Ylvisaker (1990) specifically for use with computer experiments. Sherwy and Wynn (1987; 1988) and Currin, et al. (1991) use the maximum entropy principle to develop

designs for computer experiments. Tang (1993; 1994) describes orthogonal array-based Latin hypercubes which he asserts are more suitable for computer experiments than general Latin hypercubes. Park (1994) introduces optimal Latin hypercube designs for computer experiments which either minimize IMSE or maximize entropy, spreading the points out over the design region. Beattie and Lin (1997) present a method to construct Latin hypercubes via rotated factorial designs. Fang and his co-authors (Fang, et al., 2000; Fang and Wang, 1994) use number-theoretic methods to develop uniform designs for use with computer experiments. Morris and Mitchell (Mitchell and Morris, 1992; Morris and Mitchell, 1992) propose maximin distance designs found within the class of Latin hypercube arrangements since they "offer a compromise between the entropy/maximin criterion, and good projective properties in each dimension." Owen (1992) advocates the use of orthogonal arrays as suitable designs for computer experiments, numerical integration, and visualization.

1.2. Approximation Models for Computer Experiments

As with experimental designs, a variety of approximation models and techniques exist for constructing "surrogates" of computationally expensive computer analysis and simulation codes. Response surface methodology (see, e.g., Box and Draper, 1987; Box, et al., 1978; Draper and Lin, 1990; Myers and Montgomery, 1995) and artificial neural network methods (see, e.g., Cheng and Titterington, 1994; Haykin, 1994; Smith, 1993) are two well-known approaches for constructing simple and fast approximations of complex computer analyses. An interpolative model known as kriging is also becoming widely used for the design and analysis of computer experiments (see, e.g., Barton, 1998; Booker, 1998; Currin, et al., 1991; Sacks, et al., 1989). Multivariate adaptive regression splines (Friedman, 1991) and radial basis function approximations (Dyn, et al., 1986; Powell, 1987) are also beginning to draw the attention of many researchers. Radial basis functions and multivariate adaptive regression splines are discussed in more detail in Section 2.2 along with response surface and kriging models.

In other work, Rasmussen (1990) offers an accumulated approximation technique for structural optimization which refines the approximation of objective and constraint functions by accumulating the function values of previously visited points. Similarly, Balling and Clark (1992) describe weighted and gradient-based approximations for use with optimization which utilize weighted sums of exact function values at sample points. Wavelet modeling uses a

special form of a basis function which is especially effective in modeling sharp jumps in a response surface (Mallet, 1998). Friedman and Steutzle (1981) introduce projection pursuit regression which works well in high-dimensional (< 50) data and with large data sets (can handle 200,000+ data points). Projection pursuit regression takes the data and generates different projections of it along linear combinations of the variables; an optimizer finds the best projections and builds a predictor by summing them together with arbitrary levels of precision. Multivariate Hermite approximations for multidisciplinary design optimization are introduced in (Wang, et al., 1996). A comprehensive review of applications of approximation models and techniques in mechanical and aerospace systems can be found in (Simpson, et al., 2001b); a review of metamodeling applications in structural optimization can be found in (Barthelemy and Haftka, 1993) while applications in multidisciplinary design optimization can be found in (Sobieszczanski-Sobieski and Haftka, 1997).

Despite the variety of approximations that are available, comparative studies of these approaches are limited. Kriging methods are compared against polynomial regression models for the multidisciplinary design optimization of an aerospike nozzle in (Simpson, et al., 2001a); kriging models and polynomial regression models are compared using two 5 and 10 variable test problems in (Giunta and Watson, 1998). In (Varadarajan, et al., 2000), artificial neural network methods are compared with polynomial regression models for modeling the nonlinear thermodynamic behavior of an engine design problem. In (Yang, et al., 2000), four approximation methods—enhanced multivariate adaptive regression splines, stepwise regression, neural networks, and the moving least squares—are compared for the construction of safety related functions in automotive crash analysis, for a relative small sampling size. In (Jin, et al., 2000), response surface models, radial basis functions, kriging models, and multivariate adaptive regression splines are systematically compared on a variety of test problems based on multiple measures of merit (e.g., accuracy, robustness, transparency, etc.). While no one approximation model dominated, recommendations based on problem size, degree of nonlinearity, and availability of sample data are given. The study conducted by Jin, et al. (2000) did not account for different types of experimental design strategies—only different sample sizes; therefore, our objective in this paper is to compare both experimental design strategies and approximation model types. The details of our approach are described next.

2. TECHNICAL APPROACH

Our objective in this paper is to systematically compare several experimental design types and surrogate modeling techniques in terms of their capability to generate accurate approximations of complex engineering analyses. In total, five experimental design types and four surrogate model types are utilized to build approximations for two example problems.

2.1. Experimental Designs

Four different types of "space filling" experimental design strategies are considered in this study: (1) Latin hypercubes, (2) Hammersley sequence sampling, (3) orthogonal arrays, and (4) uniform designs. A fifth type of design, namely, a set of randomly generated points, is also considered for each example. To enable direct comparisons, each design will employ comparable sample sizes. An overview of each experimental design type is offered next.

2.1.1. Latin Hypercubes

Latin hypercubes were the first type of design proposed specifically for computer experiments (McKay, et al., 1979). A Latin hypercube is a matrix of n rows and k columns where n is the number of levels being examined and k is the number of design (input) variables. Each of the k columns contains the levels 1, 2, ..., n, randomly permuted, and the k columns are matched at random to form the Latin hypercube. Latin hypercubes offer flexible sample sizes while ensuring stratified sampling, i.e., each of the input variables is sampled at n levels. These designs can have relatively small variance when measuring output variance (Sacks, et al., 1989).

2.1.2. Hammersley Sequence Sampling

Latin hypercubes are designed for uniformity along a single dimension where subsequent columns are randomly paired for placement on a k-dimensional cube. Hammersley sequence sampling provides a low-discrepancy experimental design for placing n points in a k-dimensional hypercube (Kalagnanam and Diwekar, 1997), providing better uniformity properties over the k-dimensional space than Latin hypercubes. A low discrepancy implies a uniform distribution of points in space.

2.1.3. Orthogonal Arrays

An orthogonal array is a matrix of n rows and k columns with every element being one of q symbols: $0, ..., q-1$ (Owen, 1992). An orthogonal array has an associated strength t depending on

the number of combinations of l levels appearing in any of the r columns of the array. The strength of the array and the number of levels combine to form the number of samples within the array. Orthogonal arrays provide an attractive class of sparse designs because they provide balanced (full factorial) designs for any projection into r factors (Barton, 1994).

2.1.4. Uniform Designs

A uniform design provides uniformly scatter design points in the experimental domain. A uniform design is a type of fractional factorial design with an added uniformity property; they have been popularly used since 1980 (see, Fang, 1980). If the experimental domain is finite, uniform designs are very similar to Latin hypercubes. When the experimental domain is continuous, the fundamental difference between these two designs is that in Latin hypercubes, points are selected at random from cells, whereas in a uniform design, points are selected from the center of cells. Furthermore, a Latin hypercube requires one-dimensional balance of all levels for each factor, while a uniform design requires one-dimensional balance and n-dimensional uniformity. Thus these designs are similar in one-dimension, but they can be very different in higher dimensions. Several uniform designs can be obtained from the website: <http://www.math.hkbu.edu.hk/UniformDesign>; for a recent review of uniform designs and their applications, see (Fang, et al., 2000).

In addition to the four specific types of experimental designs, sets of randomly generated points of equal sample size are considered for each example. The sample sizes for each experimental design are chosen based on the number of design variables and are discussed when each example problem is introduced. The approximation models employed in this study are discussed next.

2.2. Approximation Models

Four types of approximation models are investigated in this study: (1) polynomial response surfaces, (2) kriging models, (3) radial basis functions, and (4) multivariate adaptive regression splines. An overview of each type of approximation model is given in the following sections.

2.2.1 Response Surfaces

Originally developed for the analysis of physical experiments (Box and Wilson, 1951), polynomial response surface models have been used effectively for building approximations in a variety of applications. A second-order polynomial response surface model has the form:

$$\hat{y} = \beta_o + \sum_{i=1}^{k} \beta_i x_i + \sum_{i=1}^{k} \beta_{ii} x_i^{\,2} + \sum_i \sum_j \beta_{ij} x_i x_j \ , \tag{1}$$

where the ß parameters are computed using least squares regression. Least squares regression minimizes the sum of the squares of the deviations of predicted values, $\hat{y}(\mathbf{x})$, from the actual values, $y(\mathbf{x})$, using the equation:

$$ß = [\mathbf{X}'\mathbf{X}]^{-1}\mathbf{X}'\mathbf{y} \tag{2}$$

where **X** is the design matrix of sample data points, **X**' is its transpose, and **y** is a column vector that contains the values of the response at each sample point. Polynomial response surface models can be easily constructed, and the smoothing capability allows quick convergence of noisy functions in optimization; however, there is always a drawback when applying polynomial response surfaces to model highly nonlinear or irregular behaviors.

2.2.2 Kriging

Originally developed for applications in geostatistics (see, e.g., Cressie, 1989; Cressie, 1993), a kriging model postulates a combination of a polynomial model and departures of the form:

$$\hat{y} = \sum_{j=1}^{k} \beta_j f_j(x) + Z(x) \,, \tag{3}$$

where $Z(\mathbf{x})$ is assumed to be a realization of a stochastic process with mean zero and spatial correlation function given by:

$$\mathrm{Cov}[Z(\mathbf{x}_i), Z(\mathbf{x}_j)] = \sigma^2 \mathbf{R}(\mathbf{x}_i, \mathbf{x}_j), \tag{4}$$

where σ^2 is the process variance and R is the correlation. A variety of correlation functions can be chosen; however, the Gaussian correlation function proposed in (Sacks, et al., 1989) is the most frequently used. Furthermore, $f_j(\mathbf{x})$ is typically taken as a constant term. In our study, we use a constant term for $f_j(\mathbf{x})$ and a Gaussian correlation function with p=2 and k θ parameters, one θ for each of the k dimensions in the design space. Determining the maximum likelihood estimates of the k θ parameters used to fit the model is a k-dimensional optimization problem, which can require significant computational time if the sample data set is large, see (Simpson, et al., 1998; Simpson, et al., 2001a) for more details. The correlation matrix, **R**, can also become singular if multiple sample points are spaced close to one another or if the sample points are generated from particular designs. Fitting problems have been observed with some factorial

designs and central composite designs when using kriging models (Meckesheimer, et al., 2001; Wilson, et al., 2001). Kriging methods are extremely flexible, however, due to the wide range of correlation functions that have small amount of unknown coefficients. They can provide accurate predictions of highly nonlinear or irregular behaviors.

2.2.3. Radial Basis Functions

Radial basis functions were developed by Hardy (1971) and use linear combinations of a radially symmetric function based on Euclidean distance or similar metric to build approximation models. A simple radial basis function form is:

$$\hat{y} = \phi(\mathbf{x}) = \sum_i \beta_i \left\| \mathbf{x} - \mathbf{x}^i \right\| \tag{5}$$

where $\| \bullet \|$ represents the Euclidean norm, and the sum is taken over an observed set of system responses, $\{(\mathbf{x}^i, f(\mathbf{x}^i))\}$, $i = 1, \ldots, n$. Replacing $\phi(\mathbf{x})$ with $f(\mathbf{x}^i)$, and solving the resulting linear system yields the β_i coefficients. As commonly applied, the method is an interpolating approximation. Radial basis function approximations have produced good fits to arbitrary contours of both deterministic and stochastic response functions (Powell, 1987).

2.2.4. Multivariate Adaptive Regression Splines

Multivariate Adaptive Regression Splines (MARS) adaptively selects a set of basis functions for approximating the response function through a forward/backward iterative approach (Friedman, 1991). A MARS model can be written as:

$$\hat{y} = \sum_{m=1}^{M} a_m B_m(\mathbf{x}) \tag{6}$$

where a_m is the coefficient of the expansion, and B_m, the basis functions, can be represented as:

$$B_m(\mathbf{x}) = \prod_{k=1}^{K_m} \left[s_{k,m} \left(x_{v(k,m)} - t_{k,m} \right) \right]_+^q \tag{7}$$

where K_m is the number of factors (interaction order) in the m-th basis function, $s_{k,m}$=+/-1, $x_{v(k,m)}$ is the v-th variable, $1 \le v(k,m) \le n$, and $t_{k,m}$ is a knot location on each of the corresponding variables. The subscript '+' means the function is a truncated power function:

$$\left[s_{k,m}\left(x_{v(k,m)}-t_{k,m}\right)\right]_{+}^{q}=\begin{cases}\left[s_{k,m}\left(x_{v(k,m)}-t_{k,m}\right)\right]^{q} & s_{k,m}\left(x_{v(k,m)}-t_{k,m}\right)>0\\ 0 & otherwise\end{cases} \tag{8}$$

The major advantages of using the MARS procedure, however, appear to be accuracy and a major reduction in computational cost associated with constructing the approximation model. Compared to other techniques, the use of MARS for engineering design applications is relatively new. The algorithm described in (Chen, 1999) is utilized to build MARS models in this paper.

2.3. Assessing Model Accuracy

Since many of these approximation models interpolate the sample data, additional validation points are collected for each example to assess the accuracy of each approximation model over the region of interest. For each set of validation points, the maximum absolute error (MAX) and root mean square error (RMSE) are computed as:

$$\text{MAX} = \max\ \{|y_i - \hat{y}_i|\}_{i=1,\ldots,n_{error}} \tag{9}$$

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n_{error}}(y_i-\hat{y}_i)^2}{n_{error}}} \tag{10}$$

where n_{error} is the number of additional validation points. While RMSE provides good estimates of the "global" error over the region of interest, MAX gives a good estimate of the "local" error by measuring the worst error within the region of interest, where a good approximation will have low RMSE and low MAX values. Finally, the average absolute error and the correlation determination (R^2) were also computed using the additional validation points. Error analysis revealed that average absolute error and R^2 were both highly correlated with RMSE for these two examples; therefore, neither measure is included in this paper, and only RMSE and MAX are used when analyzing the results.

3. EXAMPLE 1: STRUCTURAL ANALYSIS OF A TWO-MEMBER FRAME

3.1. Overview of Two-Member Frame Example

Our first example for testing the five experimental design and four surrogate modeling types comes from (Arora, 1989) and is a typical engineering analysis conducted during structural optimization. This example involves the design of a two-member frame subject to out-of-plane

loads as shown in Figure 3. There are three design variables of interest: frame width (d), height (h) and wall thickness (t). The length, L, of each member is 100 in, and the load at node n_2 is P = -10,000 lbs. The stresses are calculated using the finite element method where U_1 is the vertical displacement at node n_2, U_2 is the rotation about bar n_3-n_2 and U_3 is the rotation about bar n_1-n_2.

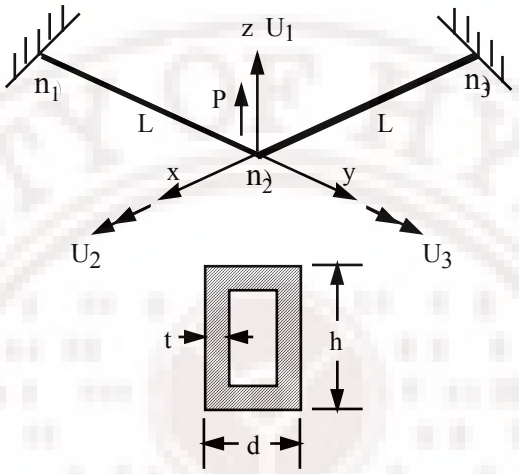


Figure 3. Two-Member Frame

The objective is to minimize the volume of the frame subject to stress constraints and bounds:

$$\underset{d,h,t}{\textbf{Min.}} \quad V(d, h, t) = 2L(2dt + 2ht - 4t^2) \tag{11}$$

$$\textbf{s.t.} \quad g_1(d, h, t) = 1 - \sigma_{e,n_1}/\sigma_{max} \geq 0 \tag{12}$$

$$g_2(d, h, t) = 1 - \sigma_{e,n_2}/\sigma_{max} \geq 0 \tag{13}$$

$$2.5 \text{ in.} \leq d \leq 10 \text{ in.}$$

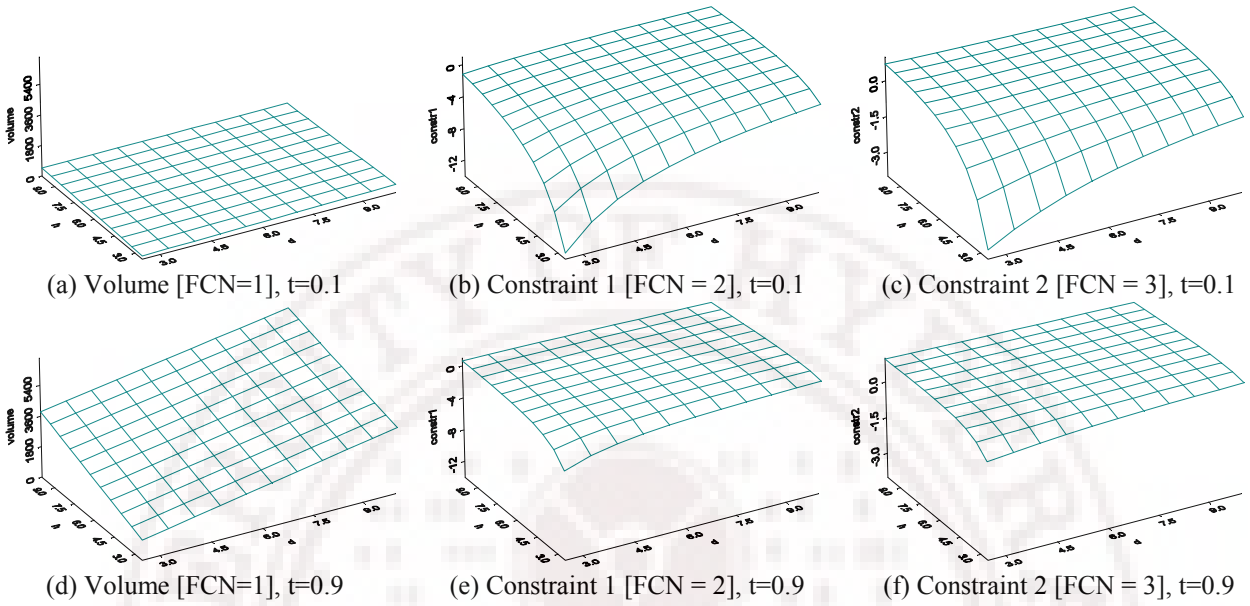
$$2.5 \text{ in.} \leq h \leq 10 \text{ in.}$$

$$0.1 \text{ in.} \leq t \leq 1.0 \text{ in.}$$

The maximum allowable stress σ_{max} = 40,000 psi, and the effective stresses at nodes n_1 and n_2, σ_{e,n_1} and σ_{e,n_2}, are determined using finite element analysis as detailed in (Arora, 1989).

The objective is to build surrogate approximations of the objective function, Eqn. 11, and the two stress constraints, Eqns. 12 and 13, that are sufficiently accurate to be used in place of the original finite element analyses. Three-D grid plots of each equation are shown in Figure 4 to gain insight into their behavior over the region of interest. In all six plots, the horizontal axes are

h and d varied over their range of interest (i.e., 2.5 to 10) while t is fixed at its lower (0.1) and upper bound (0.9).



(a) Volume [FCN=1], t=0.1 (b) Constraint 1 [FCN = 2], t=0.1 (c) Constraint 2 [FCN = 3], t=0.1

(d) Volume [FCN=1], t=0.9 (e) Constraint 1 [FCN = 2], t=0.9 (f) Constraint 2 [FCN = 3], t=0.9

Figure 4. 3-D Grid Plots of Objective Function and Constraints

From Figure 4, we see that the volume is a fairly smooth function, increasing gradually as t increases. Meanwhile, the two stress constraints are fairly flat for large values of t (i.e., the constraints are well satisfied since the stresses are low when t is large); however, for small values of t, there is a steep drop-off in one corner of the design space. The magnitude of the drop-off for the first stress constraint is nearly double that of the second (compare Figure 4b and 7c).

3.2. Experimental Set-Up for Two-Member Frame

As stated previously, the objective in this first example is to construct approximation models for the volume and the two constraints. There are three design variables—h, d, and t—whose ranges of interest are listed in the previous section. The experimental designs and approximations used in this first example are summarized as follows.

- *Experimental Design* (DOE): 5 types – Hammersley sequence (hss), Latin hypercube design (lhd), orthogonal array (oay), random set of points (rnd), uniform design (uni).
- *Sample size* (SAMP): 6 sizes – 9, 16, 25, 32, 49, 64.
- *Approximation Model* (APPROX): 4 types – kriging model (krg), radial basis function (rbf), second-order response surface (rs2), multivariate adaptive regression splines (mar).

- *Function* (FCN): 3 types – volume, stress constraint 1, stress constraint 2.

Note that a total of (5)(6)(4)(3) = 360 approximation models are constructed for this example based on the number of experimental design types, sample sizes, approximation model types, and functions being approximated. The number of sample sizes is based on available sample sizes of the orthogonal arrays and the minimum number of points needed to fit a second-order polynomial response surface. For the 9 point designs, the response surface models consist of only first-order effects and two-factor interactions since there is insufficient data to fit a full second-order model. So for each SAMP size and each DOE type, four different APPROX models are constructed for each of the three functions. To validate each approximation a set of 8000 additional validation points is used to compute MAX and RMSE, using Eqns. 9 and 10. The entire data set is available on the web at <http://edog.me.psu.edu/IJORA/>. In the next section, each function is examined independently of the other functions in terms of MAX and RMSE due to the different magnitudes of each response.

3.3. Analysis of Results

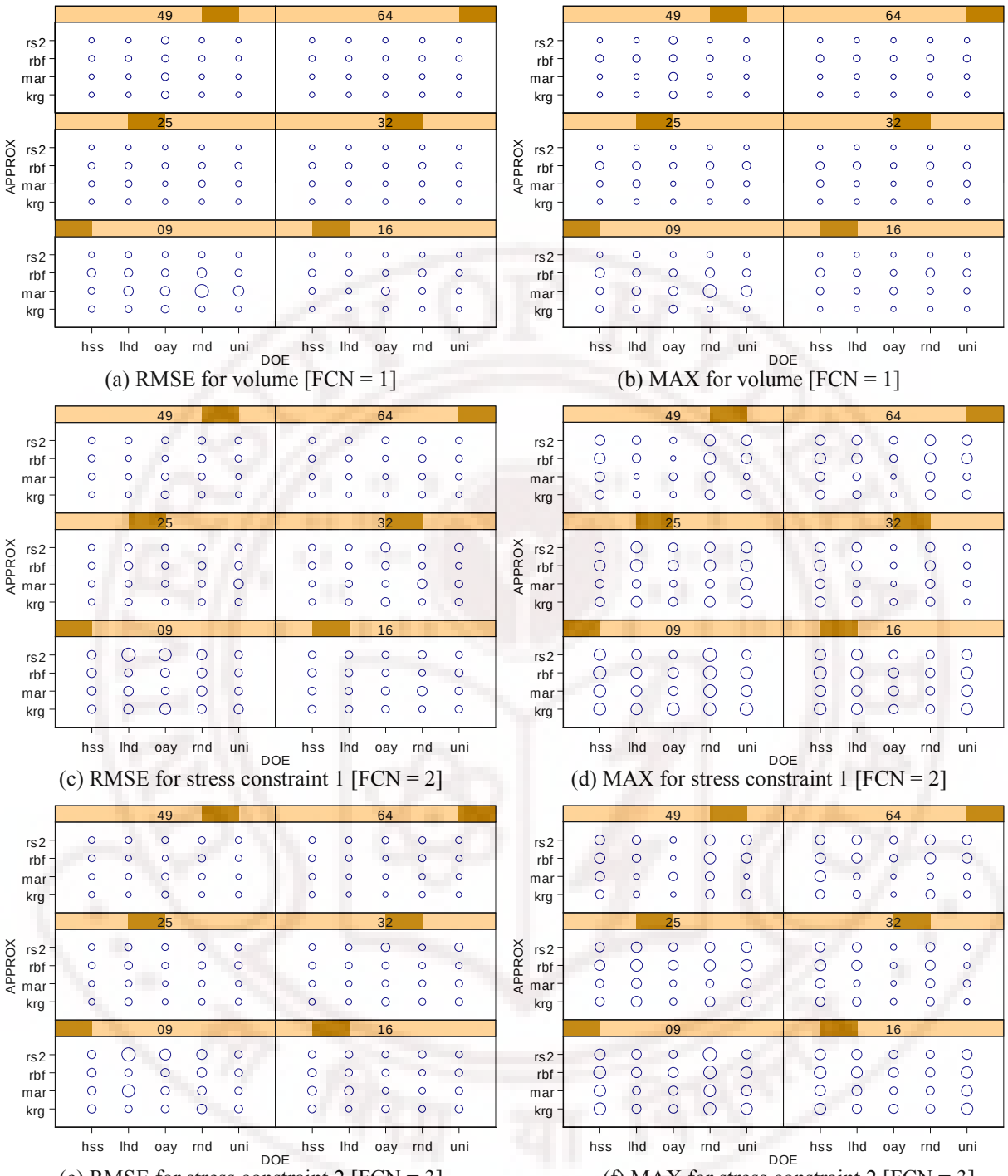
Bubble plots for RMSE and MAX are given in Figure 5. In these plots, the strip across the top of each quadrant indicates the conditioning factor (i.e., SAMP size), and the size of the circle qualitatively depicts the magnitude of the corresponding value of RMSE or MAX. Since we desire minimum values of both error measures, smaller circles indicate a more accurate fit. We will first make some preliminary observations from the bubble plots and then provide in-depth interpretations through detailed analyses.

Consistent trend of sample size: As an initial consistency check, we note in Figure 5 that metamodel accuracy improves as sample size increases—the size of the circles in each graph get smaller and smaller as sample size increases from 9 to 64. This trend is most noticeable in the plots for RMSE, but also exists in the plots for MAX as the smallest circles occur at the large sample sizes, namely, 49 and 64. Overall, this trend is consistent with intuition as the accuracy of the metamodel should improve as more sample data becomes available.

Comparison of RMSE and MAX results: In Figure 5a and Figure 5b, we note that the combinations of SAMP size, DOE type, and APPROX type that yield low RMSE values for

volume (FCN=1) also yield low MAX values. The same does not hold true for the other two functions, however. For instance, while the RMSE values for the 49 and 64 point designs for FCN=2 are low regardless for all DOE and APPROX types, only the orthogonal array (oay) designs yield consistently low MAX values for the 49 and 64 point designs as indicated by the smaller circles in Figure 5d. Meanwhile, the remaining designs yield sporadic results in that no DOE type dominates, nor does any combination of DOE type and APPROX type dominate.

Factors contributing to accuracy: Regarding APPROX type, it appears that all four types yield low RMSE and MAX values for FCN=1 except for the multivariate adaptive regression splines with the lowest sample size. Based on the smoothness of the volume as noted in Figure 4a, it is not surprising to have most APPROX types accurately model this function. However, the performance of the metamodels for the two stress constraints is not nearly as good. As noted earlier, SAMP size has a very strong impact on the accuracy. The type of DOE also appears to have some impact on accuracy; the Hammersley sampling sequence (hss) designs and uniform (uni) design tend to yield the smallest circles, while the random (rnd) sets of points and Latin hypercube designs (lhd) yield some of the largest circles. The response surface (rs2) models tend to do well regardless of SAMP size and DOE type, except for the smallest sample size. The kriging (krg) models approximate the stress constraints well, particularly in terms of RMSE; however, the MAX values appear to be about the same as those obtained from the other model types. The multivariate adaptive regression splines (mar) appear to perform very well for large sample sizes, particularly in terms of RMSE. The radial basis functions (rbf), on the other hand, give spotted performance and yield some of the largest MAX values even for FCN=1, see Figure 5b. More detailed analysis of the impact of each factor and their interactions on metamodel accuracy is presented in Figures 6-9.



(a) RMSE for volume [FCN = 1]

(b) MAX for volume [FCN = 1]

(c) RMSE for stress constraint 1 [FCN = 2]

(d) MAX for stress constraint 1 [FCN = 2]

(e) RMSE for stress constraint 2 [FCN = 3]

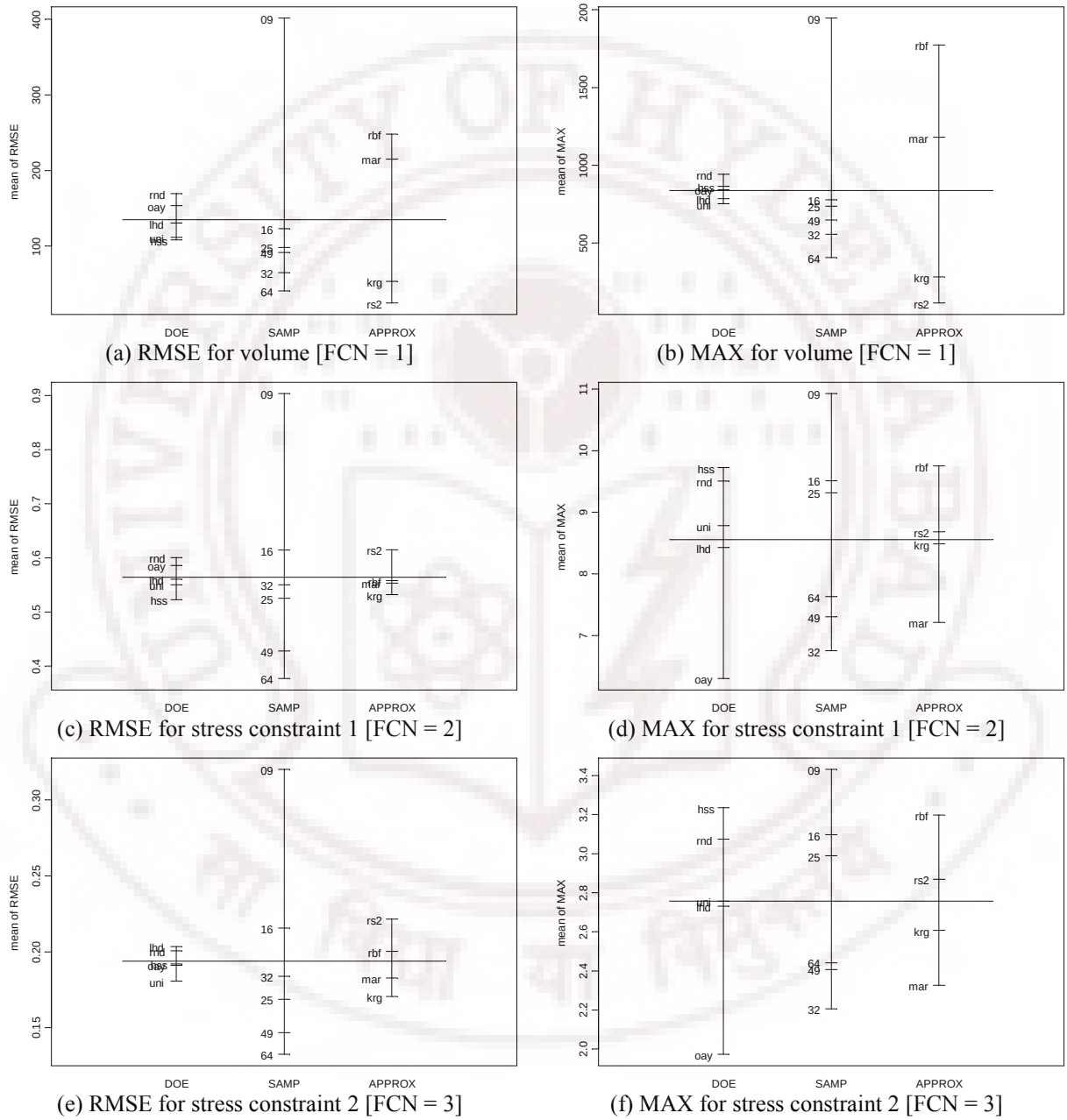
(f) MAX for stress constraint 2 [FCN = 3]

Figure 5. Effects of DOE, APPROX, and SAMP on RMSE and MAX

Contributions of individual factors:

The individual factor contributions for each of the three functions are plotted in Figure 6. On the left-hand side of Figure 6, we plot the average effect of each factor on RMSE; on the right-hand

side, we plot the average effect of each factor on MAX. For both measures, a lower value indicates a better fit. Our observations on the trend of sample size, the consistency between RMSE and MAX results, and the impact of various factors on accuracy from Figure 6 are consistent with the trends observed in Figure 5.



(a) RMSE for volume [FCN = 1]

(b) MAX for volume [FCN = 1]

(c) RMSE for stress constraint 1 [FCN = 2]

(d) MAX for stress constraint 1 [FCN = 2]

(e) RMSE for stress constraint 2 [FCN = 3]

(f) MAX for stress constraint 2 [FCN = 3]

Figure 6. Individual Factor Contributions on RMSE and MAX

Impact of DOE type: We notice that DOE types are tightly spaced in terms of RMSE for all functions and MAX for FCN=1; however, a much wider spread exists in the impact of DOE type on MAX for two stress constraints. The uniform (uni) designs and Hammersley sampling sequences (hss) are consistently among the best performers at providing accurate models with low RMSE. However, as noted in the plots for MAX in Figure 6, the orthogonal arrays (oay) tend to yield the most accurate approximations with the Hammersley sampling sequences giving the worst or next to worst performance in all cases. The uniform (uni) designs tend to be average performers for the stress constraints when measured by MAX value. As expected, the random sets of points (rnd) give some of the worst results, particularly for RMSE. This is primarily because the random points do not guarantee good coverage of the design space when compared to a uniform design, Hammersley sampling sequence, or orthogonal array. Meanwhile, while a uniform design and Hammersley sampling sequence provide good coverage of the design space, and hence low RMSE, they do not position points at the corners as one finds in an orthogonal array. This is the main reason why the resulting approximations from these two designs do not capture the drop-offs of the two stress functions which lead to higher MAX errors, even though the global accuracy indicated by the low RMSE values is still good. This also accounts for the fairly tight grouping for RMSE—all of the DOE types provide reasonably accurate approximations from a global perspective (i.e., low RMSE value) with specialized designs (e.g., uniform designs) offering slight improvements over random sets of points and Latin hypercubes.

Impact of sample size: The overall effect of SAMP size is consistent with intuition (i.e., larger sample sizes yield more accurate models). A few discrepancies among the 32, 49, and 64 sample sizes are more difficult to explain and are investigated in more detail when discussing the interactions in Figure 8 and Figure 9. In Figure 6 we observe that the 9 and 16 point designs tend to yield the worst RMSE values for all three functions while the 64 point designs yield the best. The improvement gains in RMSE appear to lessen as sample size increases above 25 points, with the biggest gains occurring when moving from 9 to 16 to 25 points. The 25, 32, and 49 point designs alternate their rank ordering in terms of their impact on RMSE and MAX. It is also interesting to note that the 32 point designs yield some of the lowest MAX values. This is primarily due to the superior performance of the orthogonal arrays for this sample size as indicated by the small circles in the bubble plot Figure 5. We believe that the poor performance

of the 49 and 64 point designs is due primarily to over-fitting the functions because they are so smooth—had the volume and stress constraints been more nonlinear, taking more sample points would most likely continue to improve the accuracy of the approximation.

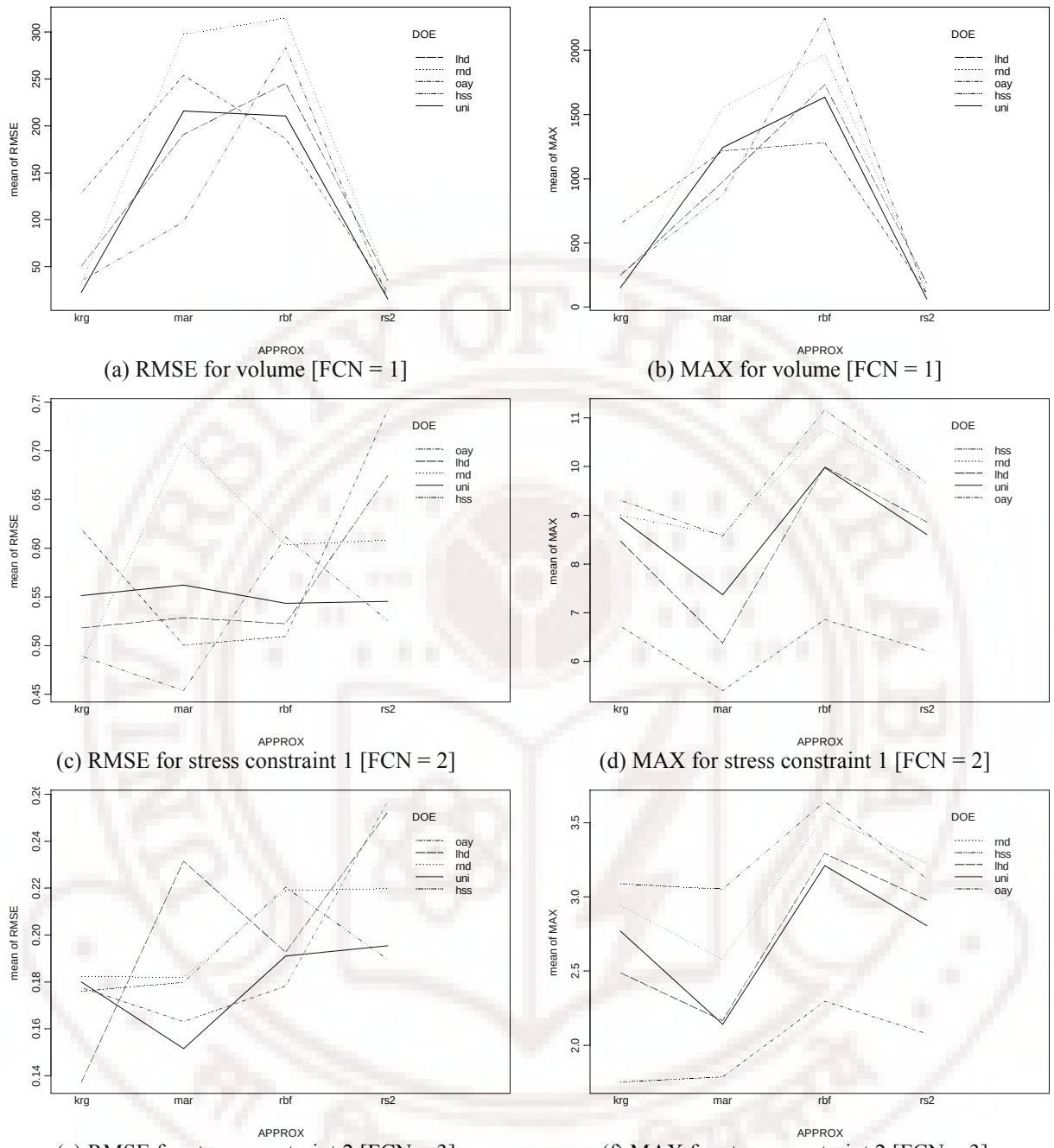
Impact of approximation type: From Figure 6 we note that the kriging (krg) models tend to yield the most accurate approximations as measured by both RMSE and MAX. The second-order response surface (rs2) models yield very accurate predictions for FCN=1 since Eqn. 25 is quadratic; however, as expected, their performance in modeling the stress constraints is not as good, particularly in terms of MAX. The radial basis functions (rbf) yield some of the worst approximations, compared to the other APPROX types, especially when MAX is considered. Finally, it is interesting to note that the multivariate adaptive regression splines (mar) provide much better approximations for the stress constraints (FCN=2 and 3) than they do for volume, and they yield the lowest MAX values for both stress constraints. This is consistent with previous observations regarding the bubble plots in Figure 5.

Contributions of Interactions between Factors:

Having looked at the individual factor contributions, the next step is to examine the interactions between different pairs of factors. Three interactions are studied: (1) DOE and APPROX, (2) DOE and SAMP, and (3) APPROX and SAMP, and the effect of each interaction is plotted in Figure 7, Figure 8, and Figure 9, respectively. The first set of interactions—APPROX type and DOE type—plotted in Figure 7, shows the impact of each APPROX type for a given DOE type, averaged over all sample sizes. The results are segmented by RMSE value and MAX value and plotted independently for each function to be consistent with previous graphs.

Interaction between DOE and APPROX type (Figure 7): We first notice the tight grouping of DOE types for the response surface (rs2) models in Figures 7a and 7b. This indicates that the response surface models accurately approximate the volume (FCN=1) independent of DOE type, which is not surprising given the quadratic nature of Eqn. 25. No other DOE-APPROX combination is as tightly grouped as found for volume. Looking at DOE types, the uniform designs (uni) appear to provide the least variation among APPROX types, yielding a nearly horizontal line for stress constraint 1 (Figure 7c). All of the DOE types appear to fluctuate when looking at MAX values; however, the orthogonal arrays are consistently among the best

performers, independent of APPROX type, as seen previously in Figure 6. It also appears that the Hammersley sampling sequences (hss) work particularly well with the multivariate adaptive regression splines (mar) to yield fairly low RMSE and MAX values for all three functions. The parallel lines in Figure 7d and Figure 7f indicate that the interaction effect of DOE and APPROX type on MAX values is very small for both constraint functions. We also notice that the multivariate adaptive regression splines (mar) method is very DOE type dependent, indicating that it is the least robust method with respect to different DOE types.



(a) RMSE for volume [FCN = 1]

(b) MAX for volume [FCN = 1]

(c) RMSE for stress constraint 1 [FCN = 2]

(d) MAX for stress constraint 1 [FCN = 2]

(e) RMSE for stress constraint 2 [FCN = 3]

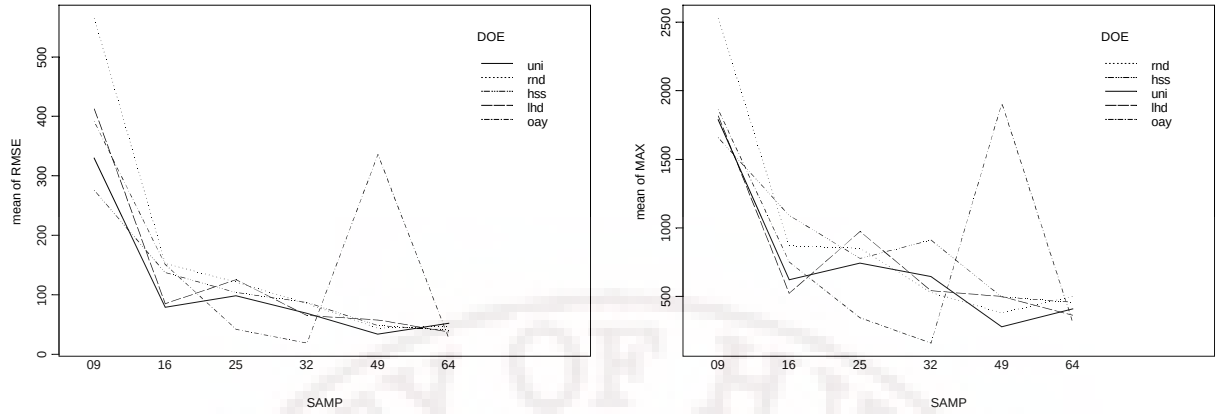
(f) MAX for stress constraint 2 [FCN = 3]

Figure 7. Interaction of DOE and APPROX on RMSE and MAX

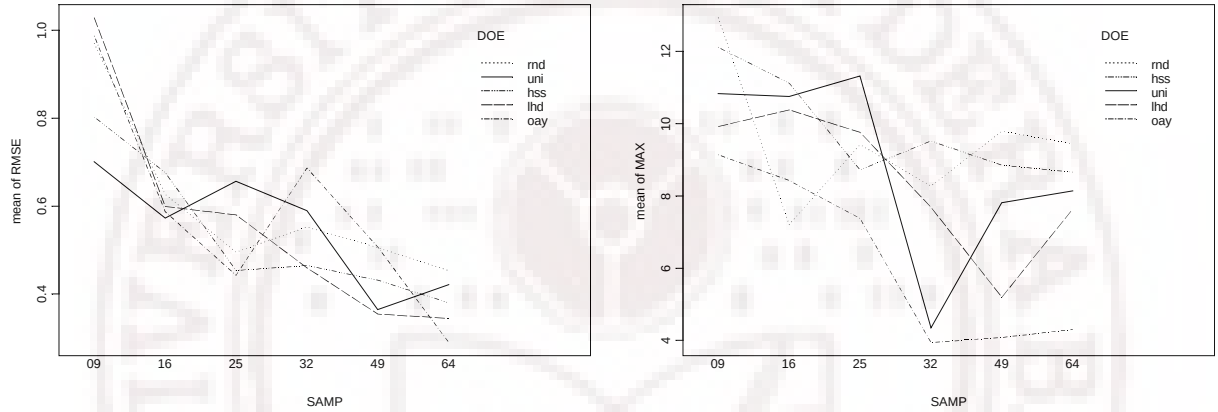
Interactions between DOE and SAMP (Figure 8): Despite some jumpiness, the general trend for each DOE type in Figure 8 is to improve accuracy as the SAMP size increases; this is particularly noticeable in the RMSE plots for all three functions. The sharp spike for the 49 point OA seen in Figure 8a is primarily due to poor randomization within the orthogonal array

and subsequently poor approximations as indicated by the slightly larger circles in Figure 6a. The Latin hypercube designs (lhd) and random sets of points (rnd) yield some of the worst RMSE and MAX values, regardless of sample size. This can be attributed to the random positioning of points in both types of designs. For instance, we see that the random sets of points (rnd) perform very poorly for low sample sizes when MAX is considered; they also yield high MAX values even with large SAMP sizes for the two stress constraints. Meanwhile, the orthogonal arrays (oay) yield the lowest MAX values for the two stress constraints. In Figure 8, the uniform (uni) designs offer some of the lowest RMSE values for many SAMP sizes; however, their performance in terms of MAX is not as good due mainly to the positioning of points within the design space as discussed previously. The Hammersley sampling sequences (hss) are average performers for low sample sizes, but they start to perform quite well for large SAMP sizes (25 and above). Finally, we note that capability of most designs to yield further improvements in RMSE tends to start to level off at 25 and 32 points, i.e., taking more than 25 or 32 points does not yield significant gains in the overall accuracy of the resulting approximation.

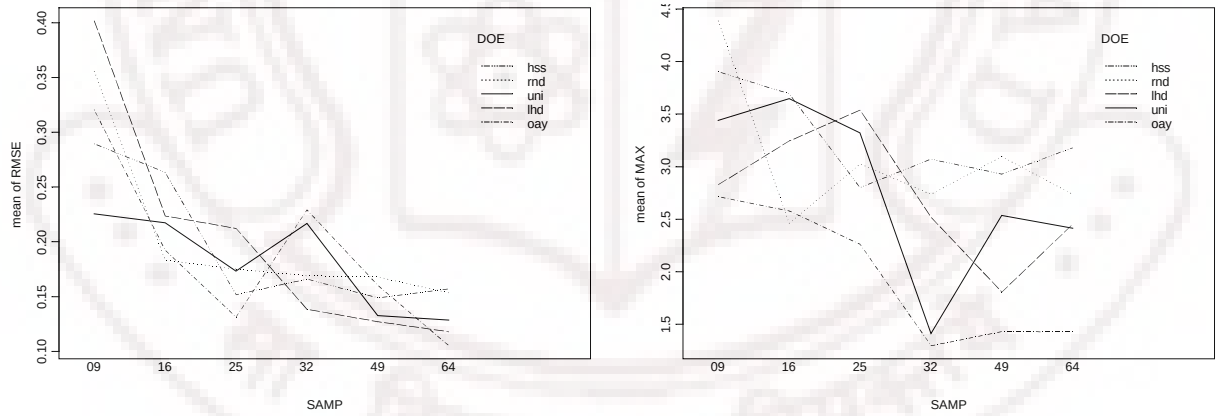
Interaction between APPROX type and SAMP size (Figure 9): As a first check, all APPROX types seem to improve as SAMP increases in Figure 9, providing consistency with previous findings and intuition. Of all the APPROX type, however, the multivariate adaptive regression splines (mar) are the most affected by SAMP size—the multivariate adaptive regression splines are among the worst performers at small SAMP sizes but are among the best performers at large SAMP sizes based on RMSE and MAX. We also notice that the kriging models perform well at low SAMP sizes as do the radial basis functions (rbf), which yield the most accurate RMSE values for the two stress constraints at the lowest SAMP size. Meanwhile, the performance of the response surface (rs2) models tend to level off for SAMP sizes greater than 9 when looking at RMSE values, and this is due to the inability to fit a full second-order model when only 9 points are available. Only ten sample points are required to fit a full second-order model for 3 variables, but it appears that once we are able to fit a full second-order response surface (rs2) model that little improvement in RMSE is obtained by increasing SAMP size further.



(a) RMSE for volume [FCN = 1] (b) MAX for volume [FCN = 1]

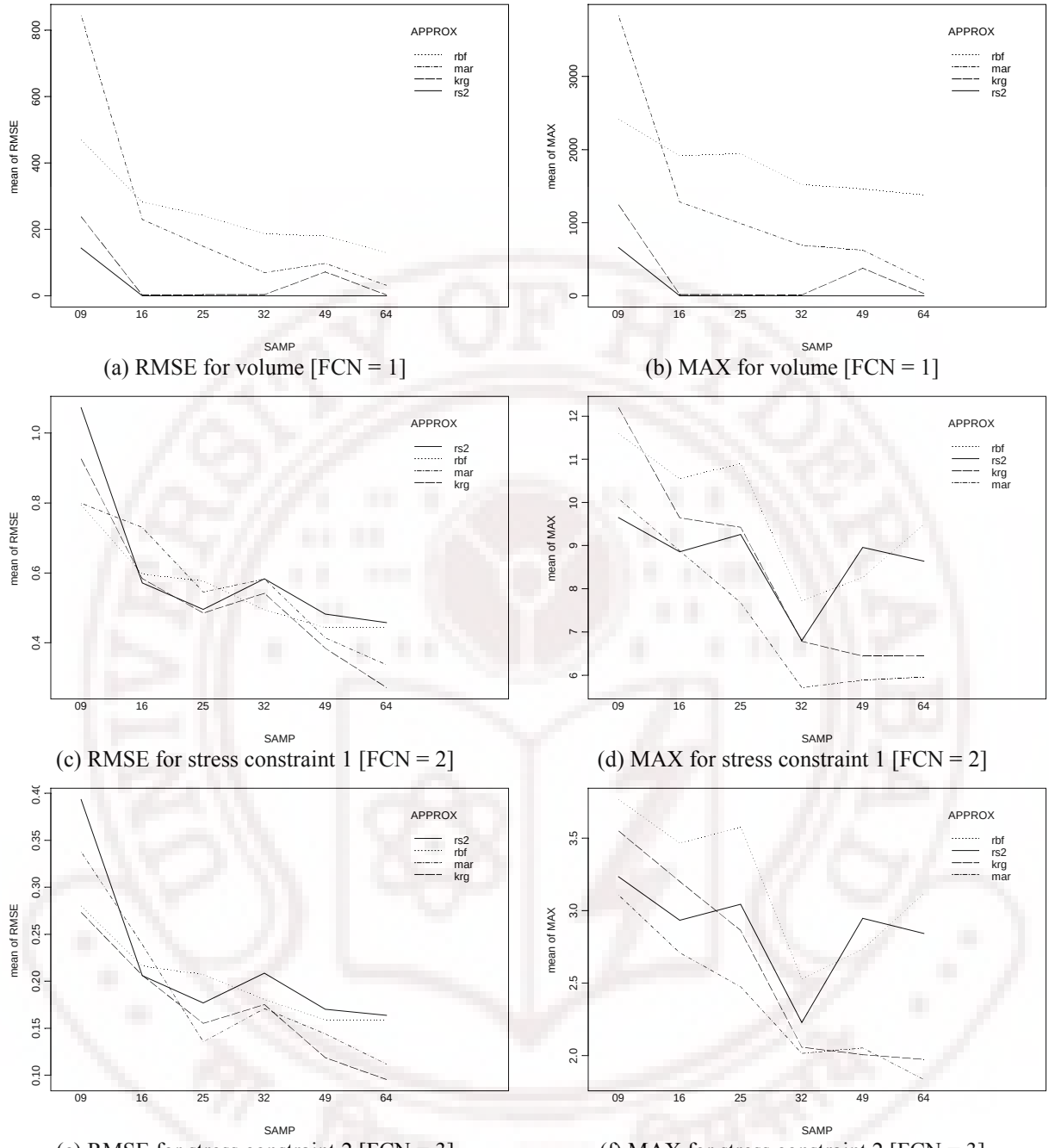


(c) RMSE for stress constraint 1 [FCN = 2] (d) MAX for stress constraint 1 [FCN = 2]



(e) RMSE for stress constraint 2 [FCN = 3] (f) MAX for stress constraint 2 [FCN = 3]

Figure 8. Interaction of DOE and SAMP on RMSE and MAX



(a) RMSE for volume [FCN = 1]

(b) MAX for volume [FCN = 1]

(c) RMSE for stress constraint 1 [FCN = 2]

(d) MAX for stress constraint 1 [FCN = 2]

(e) RMSE for stress constraint 2 [FCN = 3]

(f) MAX for stress constraint 2 [FCN = 3]

Figure 9. Interaction of SAMP and APPROX on RMSE and MAX

To determine if similar trends exits for larger problems containing many design variables (>10) and a highly nonlinear response, a fourteen variable problem involving the analysis of semi-tractor trailer roll-over is next.

4. EXAMPLE 2: ROLL-OVER ANALYSIS OF A SEMI-TRACTOR TRAILER

Our second example is a real engineering problem that analyzes vehicle design to improve a vehicle's handling characteristics, particularly the prevention of roll-over (Chen, et al., 1999). The simulator used is the integrated computer tool ArcSim (ArcSim, 1997; Sayers and Riley, 1996) developed at the University of Michigan for simulating and analyzing the dynamic behavior of 6-axle tractor-semitrailers. Each simulation takes more than three minutes to run on a Sun UltraSparc 1 workstation; therefore, using ArcSim during optimization imposes heavy computational costs. An overview of ArcSim is given in the next section; details of the example and experimental set-up are discussed in Section 4.2 with result analysis in Section 4.3.

4.1. Overview of ArcSim Example

ArcSim can simulate responses of tractor-semitrailers to user-defined steering and braking inputs on both flat and inclined surfaces (ArcSim, 1997). The program contains a nonlinear 3-D mathematical model with 91 state variables, a nonlinear tire model, and a detailed steering system model with major compliance effects. ArcSim also considers solid-axle suspensions and major suspension effects. In this study, 14 input variables are considered which include nine suspension and vehicle parameters as design variables and five uncontrollable (i.e., noise) factors for steering and braking. The response of interest is the vehicle handling performance, which is measured by the roll-over metric. The previous studies (Chen, et al., 1999) indicate that the roll-over metric has a highly nonlinear dependence on the control and noise variables, especially for different combinations of brake and steering levels. A description and the range of interest for each of the 14 input parameters are summarized in Table 1. All of the variables except brake_end have a range of +/- 20 % from their nominal values (i.e., the values for the baseline design); brake_end varies by +/- 15% to avoid overlap with the steering parameters.

In most cases, roll-over occurs due to extreme steering and braking inputs; therefore, the steering and braking parameters are taken as noise factors. Five noise factors are chosen: three corresponding to the braking inputs, and two corresponding to the steering inputs. Their ranges are also listed in Table 1. The ranges of brake_start and brake are +/- 15% from their nominal values to avoid overlap of the two parameters, whereas the other parameter ranges are +/- 20% from their nominal values. The level of braking is the amount of braking pressure applied. The

level of steering is the angle the steering wheel is turned. The starting and ending times define total time of braking and steering, respectively.

Table 1. ArcSim Variables and Ranges of Interest

Design Variable	Description	Lower Bound	Upper Bound
HH1	Height of Hitch above ground	51.2 in	76.8 in
KHX1	Hitch roll torsional stiffness	8e5 in-lb/deg	1.2e6 in-lb/deg
LTS11	Distance between springs on Axle 1	30.4 in	45.6 in
LTS123	Distance between springs on Axles 2 & 3	30.4 in	45.6 in
LTS2123	Distance between springs on Axles 4,5 & 6	30.4 in	45.6 in
M11	Laden load for Axle 1	11540 lbm	17310 lbm
M2123	Laden load for Axles 4, 5 and 6	16274.4 lbm	24411.6 lbm
KT2123	Axles 4, 5 & 6 tire stiffness	4139.20 lb/in	6208.80 lb/in
SCFS11	Axle 1 spring stiffness scale factor	0.8	1.2
Noise Variables	**Description**	**Lower Bound**	**Upper Bound**
brake_start	Time at which braking is applied	1.02 sec	1.38 sec
brake_level	Level of braking that is applied	70 psi	100 psi
brake_end	Time after which braking is no longer applied	1.53 sec	2.07sec
steer_level	Level of steering that is applied	60 deg	100 deg
steer_end	Time after which steering is no longer applied	2.16 sec	3.24 sec

In terms of the vehicle handling response, it is assumed that if the roll-over angle becomes greater than 45^{o}, roll-over will occur. The roll-over metric is one of ArcSim's outputs and is defined as the square root of the integral of the square of the roll-over angle in a 5-second period:

$$R=\sqrt{\int_0^5 roll\_angle^2\, dt} \tag{14}$$

Five seconds is chosen as the upper limit of the integration based on previous studies (Chen, et al., 1999). The square of the roll-over angle is used as the metric since the roll-over angle can take negative or positive values depending on whether roll-over is to the left or the right. Based on this definition, we note that the value of the roll-over metric is desired to be as small as possible. The plot of roll-over metric versus brake level and steering level shown in Figure 10 illustrates the high non-linearity, which is further complicated by the high dimensionality of the problem. The experimental design set-up used for this example is described next.

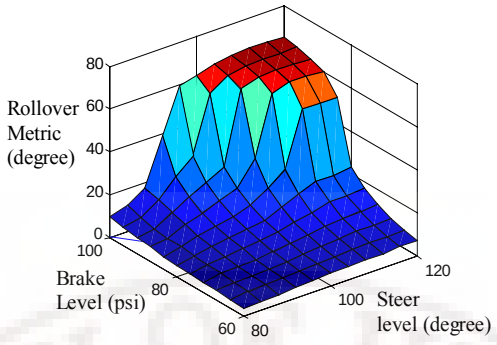


Figure 10. 3-D Plot of Roll-Over Metric Versus Brake Level and Steering Level

4.2. Experimental Set-Up for ArcSim Example

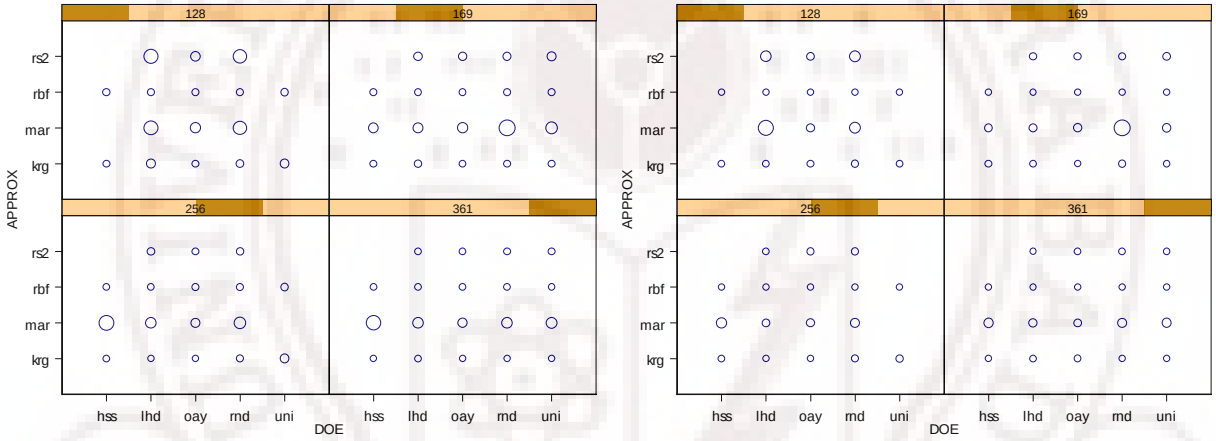
The objective in this second example is to construct an accurate approximation model for the roll-over metric computed by ArcSim. There are a total of fourteen input variables as listed in Table 1, and the ranges of interest for each variable are also listed in Table 1. The factors and levels considered in this example are summarized as follows.

- *Experimental Design* (DOE): 5 types – Hammersley sequence (hss), Latin hypercube design (lhd), orthogonal array (oay), random set of points (rnd), uniform design (uni)
- *Sample size* (SAMP): 4 sizes – 128, 169, 256, 361
- *Approximation Model* (APPROX): 4 types – kriging model (krg), radial basis function (rbf), second-order response surface (rs2), multivariate adaptive regression splines (mar)
- *Function* (FCN): 1 type – roll-over metric

Notice that a total of (4)(5)(4)(1) = 80 approximation models are constructed for this example based on the number of experimental design types, sample sizes, approximation model types, and functions being approximated. The sample sizes are based on available sample sizes of the orthogonal arrays and the minimum number of points needed to fit a second-order polynomial response surface. For each approximation of the roll-over metric, a set of 1000 additional validation points is used to compute MAX and RMSE, using Eqns. 9 and 10. The analysis of the results is next; the complete data set is available on the web at <http://edog.me.psu.edu/IJORA/>.

4.3. Analysis of ArcSim Results

Following a similar order to that in Section 3.3, a bubble plot of the effects of DOE, APPROX, and SAMP on RMSE and MAX is shown in Figure 11. Blank spaces in the bubble plot indicate that an approximation model could not be fit for that particular combination of DOE and SAMP. For instance, all of the Hammersley sequence sampling (hss) designs yielded singular design matrices (X'X), preventing a second-order response surface (rs2) model from being fit. Meanwhile, several of the multivariate adaptive regression splines yielded extremely poor approximations due to numerical round-off error and were consequently removed from the data set. This trend is consistent with previous results wherein large sample sizes are needed in order to fit accurate multivariate adaptive regression splines.



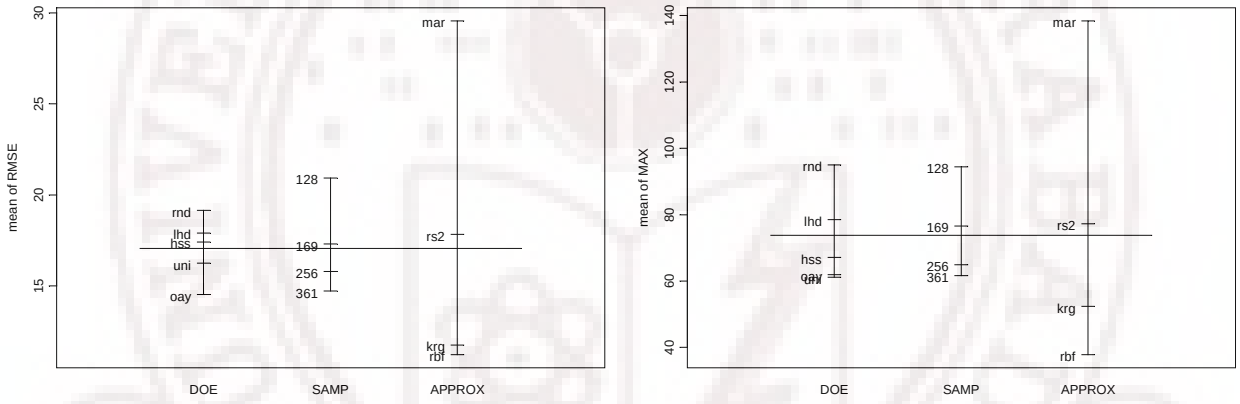
(a) for RMSE (b) for MAX

Figure 11. Effects of DOE, APPROX, and SAMP on RMSE and MAX

Factors contributing to accuracy: Looking at the data in Figure 11, the radial basis functions (rbf) and kriging (krg) models appear to offer good approximations for a wide variety of DOE and SAMP sizes. These two approximation models provide the majority of the lowest RMSE and MAX values (i.e., smallest circles) for the roll-over metric. Meanwhile, the multivariate adaptive regression splines (mar) and response surface models only yield accurate approximations for large sample sizes. The Latin hypercube designs (lhd) and random sets of points (rnd) yield some of the least accurate models as indicated by the large circles associated with many of the approximations constructed from these two DOE types. The orthogonal arrays (oay) appear to be the most robust, yielding accurate approximations for a variety of SAMP sizes

and APPROX types. The orthogonal arrays are followed closely by the uniform (uni) designs, and the Hammersley sampling sequences based on the sizes of the circles for RMSE and MAX. To further understand the impact of each DOE type, APPROX type, and SAMP size on modeling accuracy, the individual factor contributions for modeling the roll-over metric are shown in Figure 12.

Impact of DOE type: Since we want to minimize both MAX and RMSE, the orthogonal array (oay) and uniform designs (uni) appear to be the experimental designs of choice, followed closely by Hammersely sampling sequences (hss) and Latin hypercubes (lhd). The worst possible DOE types, as expected, is the random sets of points (rnd). It is again reassuring to note that both MAX and RMSE decrease as the sample size increases.



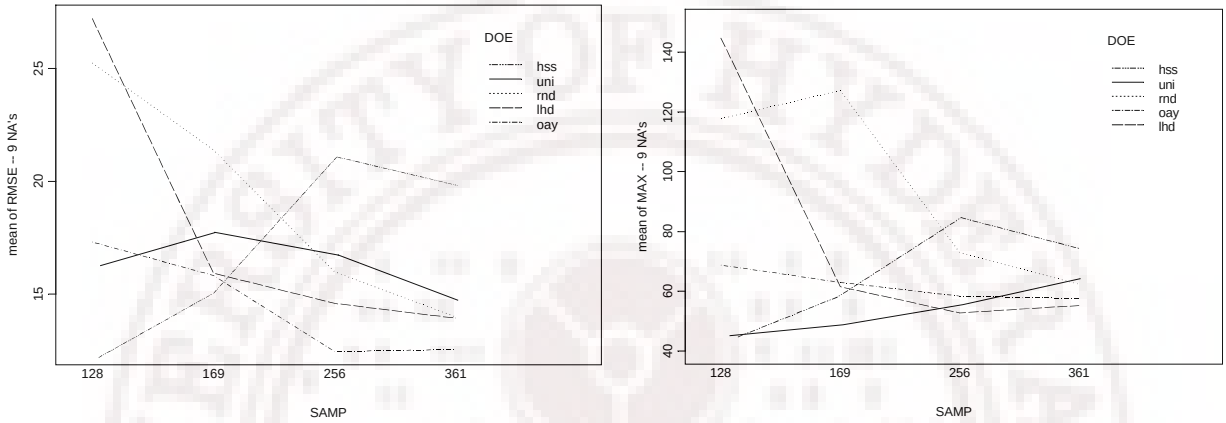
(a) for RMSE (b) for MAX

Figure 12. Factor Contributions on Model Accuracy

Impact of APPROX type: The radial basis function (rbf) and kriging models (krg) both perform well in terms of MAX and RMSE as shown in Figure 12. The radial basis function models offer a slight improvement in MAX error over the kriging (krg) models, while both approximations yield similar RMSE values. The second-order response surface (rs2) models yield average results for both error measures while the multivariate adaptive regression splines (mar) are the most inaccurate, particularly when small sample sizes are used. This is consistent with previous results (Jin, et al., 2000).

Interaction between DOE and SAMP (Figure 13): Consistent with the previous figure, most of the lines in Figure 13 are negatively sloped, indicating increased accuracy as sample size

increases; however, there are some exceptions. Most notably, the MAX values for the uniform design (uni) rise slightly as sample size increases, but the corresponding RMSE values tend to decrease. Meanwhile, the Hammersley sampling sequences (hss) show the most pronounced increase in MAX and RMSE when moving to the 361 point designs, while the other DOE types level off at the higher sample sizes.

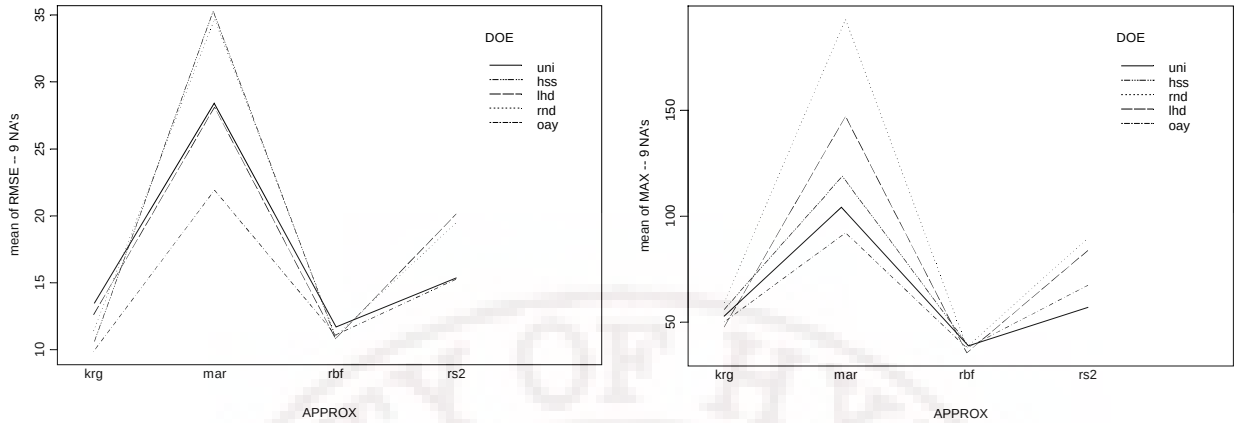


(a) for RMSE (b) for MAX

Figure 13. Interaction of DOE and SAMP

Interaction between DOE and APPROX (Figure 14): We see from Figure 14 that both the kriging (krg) and radial basis function (rbf) models yield good results, regardless of DOE type; the same does not hold true for the response surface (rs2) models or the multivariate adaptive regression splines (mar). Of the five design types available, the uniform designs (uni) and orthogonal arrays (oay) seem to work well with both multivariate adaptive regression splines (mar) and radial basis functions (rbf) while the random points provide the worst data set for fitting accurate approximations.

Interaction between APPROX type and SAMP size: The interaction between APPROX and SAMP is not plotted for this example because the only interaction that exists has already been captured, namely, as SAMP size increases, each APPROX yields more accurate results. There are no big jumps in accuracy for any particular APPROX type as observed previously.



(a) for RMSE (b) for MAX

Figure 14. Interaction of DOE and APPROX

5. RECOMMENDATIONS AND CLOSING REMARKS

Based on the comparative study results for two engineering example problems, one small dimension ($k = 3$) with both low-order and higher-order nonlinear functions, and the other large dimension ($k = 14$) with higher-order nonlinear behavior, some general conclusions can be drawn regarding the selection of DOE type, the approximation type and the sample size.

For DOE type, we find that good design space coverage afforded by the uniform designs and Hammersley sampling sequences tend to yield more accurate approximations globally as indicated by the consistently low RMSE values associated with them. The uniform designs tend to perform well at low sample sizes while the Hammersley sampling sequences tend to fair better when large sample sizes can be afforded, but both offer improvements over standard Latin hypercube designs and random sets of points. The orthogonal arrays do particularly well at giving low MAX values because these designs place points at the corners of the design space which is critical when trying to approximate the two stress constraints in the first example. This type of behavior may not always be present in a system, and we recommend a design that provides good overall coverage (and therefore lower RMSE) be chosen over one that yields low MAX—validation of the approximation during use can always help correct large MAX values.

For APPROX type, the kriging (krg) and radial basis function (rbf) models tend to offer more accurate approximations over a wide range of DOE types and SAMP sizes. The performance of the multivariate adaptive regression splines (mar) is the least stable; its performance varies quite

a lot when different sample sizes or DOE types are available. In both examples, large sample sizes are needed to fit accurate multivariate adaptive regression splines. The second-order response surfaces yield average results and perform particularly well when approximating low-order non-linear functions.

For SAMP size, larger sizes generally improve the accuracy, however, for low-order non-linear functions, we also find that taking large samples for many approximation types, with the exception of the multivariate adaptive regression splines, does not improve accuracy that much.

ACKNOWLEDGEMENTS

We thank the reviewers for their helpful comments. Dr. Simpson acknowledges support from Dr. Kam Ng, ONR 333, through the Naval Sea Systems Command under Contract No. N00014-00-G-0058. The work by Dr. Lin is partially supported by the NSF/DMS-9704711 and National Science Council of ROC via Contract NSC 87-2119-M-001-007. Dr. Chen acknowledges support from NSF/DMII 9896300.

REFERENCES

ArcSim, 1997, "ArcSim User's Guide," *PDF version can be downloaded from http://arc.engin.umich.edu/sw_distri/ ARCSIM/ arcsim.html,* The University of Michigan, Ann Arbor, MI.

Arora, J. S., 1989, *Introduction to Optimum Design*, McGraw-Hill, New York.

Balling, R. J. and Clark, D. T., 1992, September 21-23, "A Flexible Approximation Model for Use with Optimization," *4th AIAA/USAF/NASA/OAI Symposium on Multidisciplinary Analysis and Optimization,* Cleveland, OH, AIAA, Vol. 2, pp. 886-894. AIAA-92-4801-CP.

Barthelemy, J.-F. M. and Haftka, R. T., 1993, "Approximation Concepts for Optimum Structural Design - A Review," *Structural Optimization,* Vol. 5, pp. 129-144.

Barton, R. R., 1992, December 13-16, "Metamodels for Simulation Input-Output Relations," *Proceedings of the 1992 Winter Simulation Conference (Swain, J. J., Goldsman, D., et al., eds.),* Arlington, VA, IEEE, pp. 289-299.

Barton, R. R., 1994, December 11-14, "Metamodeling: A State of the Art Review," *Proceedings of the 1994 Winter Simulation Conference (Tew, J. D., Manivannan, S., et al., eds.),* Lake Beuna Vista, FL, IEEE, pp. 237-244.

Barton, R. R., 1998, December 13-16, "Simulation Metamodels," *Proceedings of the 1998 Winter Simulation Conference (WSC'98) (Medeiros, D. J., Watson, E. F., et al., eds.),* Washington, DC, IEEE, pp. 167-174.

Beattie, S. D. and Lin, D. K. J., 1997, "Designing Computer Experiments: Rotated Factorial Designs," *Technical Report No. 97-06,* Department of Statistics, The Pennsylvania State University, University Park, PA.

Booker, A. J., 1998, September 2-4, "Design and Analysis of Computer Experiments," *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis & Optimization,* St. Louis, MO, AIAA, Vol. 1, pp. 118-128. AIAA-98-4757.

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

Box, G. E. P., Hunter, W. G. and Hunter, J. S., 1978, *Statistics for Experimenters: An Introduction to Design, Data Analysis, and Model Building*, John Wiley & Sons, New York.

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

Chen, V. C. P., 1999, "Application of MARS and Orthogonal Arrays to Inventory Forecasting Stochastic Dynamic Programs," *Computational Statistics and Data Analysis,* Vol. 30, pp. 317-341.

Chen, W., Garimella, R. and Michelena, N., 1999, September 12-15, "Robust Design for Improved Vehicle Handling under a Range of Maneuver Conditions," *Advances in Design Automation,* Las Vegas, NV, ASME, Paper No. 99-DETC/DAC-8580.

Cheng, B. and Titterington, D. M., 1994, "Neural Networks: A Review from a Statistical Perspective," *Statistical Science,* Vol. 9, No. 1, pp. 2-54.

Cressie, N., 1989, "Geostatistics," *The American Statistician,* Vol. 43, No. 4, pp. 197-202.

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

Currin, C., Mitchell, T., Morris, M. and Ylvisaker, D., 1991, "Bayesian Prediction of Deterministic Functions, With Applications to the Design and Analysis of Computer Experiments," *Journal of the American Statistical Association,* Vol. 86, No. 416, pp. 953-963.

Draper, N. R. and Lin, D. K. J., 1990, "Connections Between Two-Level Designs of Resolutions III and V," *Technometrics,* Vol. 32, No. 3, pp. 283-288.

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

Fang, K. T., 1980, "Experimental Design By Uniform Distribution," *Acta Mathematice Applicatae Sinica,* Vol. 3, No. 363-372, pp.

Fang, K.-T., Lin, D. K. J., Winker, P. and Zhang, Y., 2000, "Uniform Design: Theory and Application," *Technometrics,* Vol. 42, pp. 237-248.

Fang, K.-T. and Wang, Y., 1994, *Number-theoretic Methods in Statistics*, Chapman & Hall, New York.

Friedman, J. H., 1991, "Multivariate Adaptive Regression Splines," *The Annals of Statistics,* Vol. 19, No. 1, pp. 1-67.

Friedman, J. H. and Steutzle, W., 1981, "Projection Pursuit Regression," *Journal of the American Statistical Association,* Vol. 76, No. 376, pp. 817-823.

Gangadharan, S. N., Haftka, R. T. and Fiocca, Y. I., 1995, April 10-13, "Variable-Complexity-Modeling Structural Optimization Using Response Surface Methodology," *36th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference and AIAA/ASME Adaptive Structures Forum,* New Orleans, LA, AIAA, AIAA-95-1164.

Giunta, A. and Watson, L. T., 1998, September 2-4, "A Comparison of Approximation Modeling Techniques: Polynomial Versus Interpolating Models," *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis & Optimization,* St. Louis, MO, AIAA, Vol. 1, pp. 392-404. AIAA-98-4758.

Goldsman, D. and Nelson, B. L., 1998, December 13-16, "Statistical Screening, Selection, and Multiple Comparison Procedures in Computer Simulation," *Proceedings of the 1998 Winter Simulation Conference (WSC'98) (Medeiros, D. J., Watson, E. F., et al., eds.),* Washington, DC, IEEE, pp. 159-166.

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

Haykin, S., 1994, *Neural Networks: A Comprehensive Foundation*, Macmillan Publishing, New York.

Jin, R., Chen, W. and Simpson, T. W., 2000, September 6-8, "Comparative Studies of Metamodeling Techniques under Multiple Modeling Criteria," *8th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization,* Long Beach, CA, AIAA, AIAA-2000-4801.

Johnson, M. E., Moore, L. M. and Ylvisaker, D., 1990, "Minimax and Maximin Distance Designs," *Journal of Statistical Planning and Inference,* Vol. 26, No. 2, pp. 131-148.

Kalagnanam, J. R. and Diwekar, U. M., 1997, "An Efficient Sampling Technique for Off-Line Quality Control," *Technometrics,* Vol. 39, No. 3, pp. 308-319.

Koch, P. N., Allen, J. K., Mistree, F. and Mavris, D., 1997, September 14-17, "The Problem of Size in Robust Design," *Advances in Design Automation,* Sacramento, CA, ASME, Paper No. DETC97/DAC-3983.

Koch, P. N., Mavris, D. and Mistree, F., 1998, September 2-4, "Multi-Level, Partitioned Response Surfaces for Modeling Complex Systems," *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis & Optimization,* St. Louis, MO, AIAA, Vol. 3, pp. 1954-1968. AIAA-98-4958.

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

Mallet, C. G., 1998, *A Wavelet Tour of Signal Processing*, Academic Press, Boston, MA.

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

Meckesheimer, M., Barton, R. R., Simpson, T. W., Limayem, F. and Yannou, B., 2001, "Metamodeling of Combined Discrete/Continuous Responses," *AIAA Journal,* pp. to appear.

Mitchell, T. J. and Morris, M. D., 1992, "Bayesian Design and Analysis of Computer Experiments: Two Examples," *Statistica Sinica,* Vol. 2, pp. 359-379.

Montgomery, D. C., 1997, *Design and Analysis of Experiments*, Fourth Edition, John Wiley & Sons, New York.

Morris, M. D. and Mitchell, T. J., 1992, "Exploratory Designs for Computer Experiments," *ORNL/TM-12045,* Oak Ridge National Laboratory, Oak Ridge, TN.

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

Owen, A. B., 1992, "Orthogonal Arrays for Computer Experiments, Integration and Visualization," *Statistica Sinica,* Vol. 2, pp. 439-452.

Park, J.-S., 1994, "Optimal Latin-Hypercube Designs for Computer Experiments," *Journal of Statistical Planning and Inference,* Vol. 39, No. 1, pp. 95-111.

Powell, M. J. D., 1987, "Radial Basis Functions for Multivariable Interpolation: A Review," *Algorithms for Approximation (Mason, J. C. and Cox, M. G., eds.),* Oxford University Press, London, pp.

Rasmussen, J., 1990, "Accumulated Approximation-A New Method for Structural Optimization by Iterative Improvement," *3rd Air Force/NASA Symposium on Recent Advances in Multidisciplinary Analysis and Optimization,* San Francisco, CA, pp. 253-258.

Sacks, J., Welch, W. J., Mitchell, T. J. and Wynn, H. P., 1989, "Design and Analysis of Computer Experiments," *Statistical Science,* Vol. 4, No. 4, pp. 409-435.

Sayers, M. W. and Riley, S. M., 1996, "Modeling Assumptions for Realistic Multibody Simulations of the Yaw and Roll Behavior of Heavy Trucks," *SAE Paper No. 960173,* Society of Automotive Engineers, Warrendale, PA.

Shewry, M. C. and Wynn, H. P., 1987, "Maximum Entropy Sampling," *Journal of Applied Statistics,* Vol. 14, No. 2, pp. 165-170.

Shewry, M. C. and Wynn, H. P., 1988, "Maximum Entropy Sampling with Application to Simulation Codes," *Proceedings of the 12th World Congress on Scientific Computation,* IMAC88, Vol. 2, pp. 517-519.

Simpson, T. W., Allen, J. K. and Mistree, F., 1998, September 13-16, "Spatial Correlation Metamodels for Global Approximation in Structural Design Optimization," *Advances in Design Automation,* Atlanta, GA, ASME, Paper No. DETC98/DAC-5613.

Simpson, T. W., Mauery, T. M., Korte, J. J. and Mistree, F., 2001a, "Kriging Metamodels for Global Approximation in Simulation-Based Multidisciplinary Design Optimization," *AIAA Journal,* to appear: Vol. 40, No. 1.

Simpson, T. W., Peplinski, J., Koch, P. N. and Allen, J. K., 2001b, "Metamodels for Computer-Based Engineering Design: Survey and Recommendations," *Engineering with Computers,* Vol. 17, No. 2, pp. 129-150.

Smith, M., 1993, *Neural Networks for Statistical Modeling*, von Nostrand Reinhold, New York.

Sobieszczanski-Sobieski, J. and Haftka, R. T., 1997, "Multidisciplinary Aerospace Design Optimization: Survey of Recent Developments," *Structural Optimization,* Vol. 14, pp. 1-23.

Tang, B., 1993, "Orthogonal Array-Based Latin Hypercubes," *Journal of the American Statistical Association,* Vol. 88, No. 424, pp. 1392-1397.

Tang, B., 1994, "A Theorem for Selecting OA-Based Latin Hypercubes Using a Distance Criterion," *Communications in Statistics, Theory and Methods,* Vol. 23, No. 7, pp. 2047-2058.

Varadarajan, S., Chen, W. and Pelka, C., 2000, "The Robust Concept Exploration Method with Enhanced Model Approximation Capabilities," *Engineering Optimization,* Vol. 32, No. 3, pp. 309-334.

Wang, L., Grandhi, R. V. and Canfield, R. A., 1996, "Multivariate Hermite Approximation for Design Optimization," *International Journal for Numerical Methods in Engineering,* Vol. 39, No. 5, pp. 787-803.

Welch, W. J., Buck, R. J., Sacks, J., Wynn, H. P., Mitchell, T. J. and Morris, M. D., 1992, "Screening, Predicting, and Computer Experiments," *Technometrics,* Vol. 34, No. 1, pp. 15-25.

Wilson, B., Cappelleri, D. J., Frecker, M. I. and Simpson, T. W., 2001, "Efficient Pareto Frontier Exploration Using Surrogate Approximations," *Optimization and Engineering,* Vol. 2, No. 1, pp. 31-50.

Yang, R. J., Gu, L., Liaw, L., Gearhart, C., Tho, C. H., Liu, X. and Wang, B. P., 2000, September 10-13, "Approximations for Safety Optimization of Large Systems," *ASME 2000 Design Engineering Technical Conferences - Design Automation Conference (Renaud, J. E., ed.),* Baltimore, MD, ASME, Paper No. DETC-2000/DAC-14245.

Statistical Science
1989, Vol. 4, No. 4, 409–435

Design and Analysis of Computer Experiments

Jerome Sacks, William J. Welch, Toby J. Mitchell and Henry P. Wynn

Abstract. Many scientific phenomena are now investigated by complex computer models or codes. A computer experiment is a number of runs of the code with various inputs. A feature of many computer experiments is that the output is deterministic—rerunning the code with the same inputs gives identical observations. Often, the codes are computationally expensive to run, and a common objective of an experiment is to fit a cheaper predictor of the output to the data. Our approach is to model the deterministic output as the realization of a stochastic process, thereby providing a statistical basis for designing experiments (choosing the inputs) for efficient prediction. With this model, estimates of uncertainty of predictions are also available. Recent work in this area is reviewed, a number of applications are discussed, and we demonstrate our methodology with an example.

Key words and phrases: Experimental design, computer-aided design, kriging, response surface, spatial statistics.

1. INTRODUCTION

Computer modeling is having a profound effect on scientific research. Many processes are so complex that physical experimentation is too time consuming or too expensive; or, as in the case of weather modeling, physical experiments may simply be impossible. As a result, experimenters have increasingly turned to mathematical models to simulate these complex systems. Advances in computational power have allowed both greater complexity and more extensive use of such models. Virtually every area of science and technology is affected. Our direct experience has been with applications in combustion, VLSI-circuit design, controlled-nuclear-fusion devices, plant ecology, and thermal-energy storage, but this is only a small sample.

Jerome Sacks is Professor and Head, Department of Statistics, University of Illinois, Champaign, Illinois 61820. William J. Welch is Associate Professor, Department of Statistics and Actuarial Science, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1. Toby J. Mitchell is Senior Research Staff Member, Oak Ridge National Laboratory, Oak Ridge, Tennessee 37831-8083. Henry P. Wynn is Dean, School of Mathematics, Actuarial Science and Statistics, City University, London EC1V 0HB, England.

Computer models (or codes) often have high-dimensional inputs, which can be scalars or functions. The output may also be multivariate. In particular, it is common for the output to be a time-dependent function from which a number of summary responses are extracted. For simplicity here, we shall assume that interest is focused on a relatively small set of scalar inputs, x, and on a single scalar response, y. Making a number of runs at various input configurations is what we call a computer experiment. The design problem is the choice of inputs for efficient analysis of the data.

The computer models we address in this article are deterministic; replicate observations from running the code with the same inputs will be identical. It is this lack of random error that makes computer experiments different from physical experiments, calling for distinct techniques.

In the next section we describe some applications. An understanding of the scientific background and objectives will be helpful in Section 3 where the role of statistics in modeling deterministic systems is discussed. This organization also parallels our research program, which has largely responded to a number of examples. Our statistical model, adopted from kriging in the spatial statistics literature and described in Section 4, treats the response as if it were a realization of a stochastic process. This provides a statistical basis for computing an efficient predictor of the response

at untried inputs and allows estimates of uncertainty of predictions. Within this framework, Section 5 discusses design criteria and algorithms for construction of designs. Applying these methods to an electronic-circuit simulator in Section 6 demonstrates what is already possible. On the other hand, one of the purposes of this paper is to highlight open problems and questions. Some of these are discussed and summarized in Section 7.

2. EXAMPLES AND OBJECTIVES

Kee, Grcar, Smooke and Miller (1985) described a fluid-dynamics model for flames which solves a complex set of partial differential equations. In an ongoing study with M. Frenklach and H. Wang, the input vector is taken to be five rate constants controlling five of the chemical reactions, and the response is the flame velocity. The numerous other inputs to the code are set at standard values based on knowledge of the chemistry. The ultimate objective here is to tune the computer model, that is find rate constants yielding a flame velocity that matches physical data. A physical analog of this experiment is impossible, because the rate constants are indeed *constants* and cannot be manipulated in reality. The need for careful design of the inputs is underlined by the fact that a single run of the code takes up to 20 minutes on a Cray X-MP.

Following Frenklach and Rabinowitz's work, Sacks, Schiller and Welch (1989) discussed examples of methane combustion based on the solution of a large system of (ordinary) differential equations arising from chemical kinetics. Although the objective is similar to that of the above flame example, the system of equations is simpler and the numerical complexity is less, allowing statistical design and analysis for a larger set of inputs.

Another important application area is quality improvement of integrated circuits. This can involve simulation of both the manufacturing process and the circuit. For example, Nassif, Strojwas and Director (1984) described the FABRICS II simulator and applied it to the processing of a ring oscillator. In these applications, the inputs are circuit parameters such as nominal transistor sizes and/or process parameters such as reagent doses, and the response might be a circuit delay time. Often, process variability is incorporated in these models by Monte Carlo sampling of a noise distribution (e.g., Singhal and Pinel, 1981). Conditional on the noise inputs, however, the simulator is deterministic. The usual objective is to find settings of the engineering or process parameters that make the response insensitive to noise, as emphasized in recent years by Taguchi (1986) and others.

Following Taguchi, the input variables x can often be divided into control factors x_{con} and noise factors x_{noise}. In a circuit-simulator example studied by Welch, Yu, Kang and Sacks (1988), the control factors were transistor dimensions and the noise factors corresponded to manufacturing-process variability. The response y was a measure of the asynchronization of two clocks, ideally zero. Generally, given a loss function $L(y)$, a "parameter design" problem can be formalized as minimizing expected loss

$$\int L[y(x_{\text{con}}, x_{\text{noise}})]d\Gamma(x_{\text{noise}})$$

over x_{con}. Here $\Gamma(x_{\text{noise}})$ is an assumed distribution of the noises. In the example, $L(y)$ was y^2 and Γ was approximated by a uniform distribution on five noise combinations to represent typical and extreme noise conditions.

Another example is a thermal-energy storage model, TWOLAYER, created by A. Solomon and colleagues at Oak Ridge National Laboratory. This simulates heat transfer into and out of a wall containing two layers of phase-change materials. Currin, Mitchell, Morris and Ylvisaker (1988) described a simple experiment with melting temperature and layer thickness as inputs. The response was a heat-storage-utility index, and the main objective was to determine configurations of the input parameters yielding high values of the index. The computational time for a single run, normally several minutes on a Cray X-MP, was reduced by Currin, Mitchell, Morris and Ylvisaker (1988) for the purposes of their experiment by requiring only a coarse solution to the differential equations of the computer model.

These examples illustrate that the computer experimenter, like the physical experimenter, can have many purposes in mind. We see three primary objectives:

- Predict the response at untried inputs.
- Optimize a functional of the response.
- Tune the computer code to physical data.

These objectives prompt basic statistical questions:

- *The design problem: At which input "sites" $S = \{s_1, \cdots, s_n\}$ should data $y(s_1), \cdots, y(s_n)$ be collected?*
- *The analysis problem: How should the data be used to meet the objective?*

In this article we concentrate on the prediction objective, as it is plausibly the most basic. If a sufficiently precise predictor can be found, the experimenter then has a cheap surrogate for the simulator. "What if" questions can be explored, optimization can be performed on the predictor, etc.

3. THE ROLE OF STATISTICS

These deterministic computer experiments differ substantially from the physical experiments per-

formed by agricultural and biological scientists of the early 20th century. Their experiments had substantial random error due to variability in the experimental units. Relatively simple models were often successful. The remarkable methodology for design of experiments introduced by Fisher (1935) and the associated analysis of variance is a systematic way of separating important treatment effects from the background noise (as well as from each other). Fisher's stress on blocking, replication and randomization in these experiments reduced the effect of random error, provided valid estimates of uncertainty, and preserved the simplicity of the models.

The above deterministic examples also differ from codes in the simulation literature (e.g., Kleijnen, 1987), which incorporate substantial random error through random number generators. It has been natural, therefore, to design and analyze such stochastic simulation experiments using standard techniques for physical experiments.

Apparently, McKay, Conover and Beckman (1979) were the first to explicitly consider experimental design for deterministic computer codes. They introduced Latin hypercube sampling, an extension of stratified sampling which ensures that each of the input variables has all portions of its range represented. Latin hypercubes are computationally cheap to generate and can cope with many input variables. These designs are aimed at an objective different from those we discussed in Section 2: namely, how a known distribution of the inputs propagates through to the output distribution. (Of course, conditional on the inputs, the output is still deterministic.) For this purpose, Iman and Helton (1988) compared Latin hypercube sampling with Monte Carlo sampling of a response surface replacement for the computer model. The response surface was fitted by least squares to data from a fractional-factorial design. They found in a number of examples that the response surface could not adequately represent the complex output of the computer code but could be useful for ranking the importance of the input variables. Because Latin hypercube sampling exercises the code over the entire range of each input variable, it can also be a systematic way of discovering scientifically surprising behavior, as noted in Iman and Helton (1988).

In the absence of independent random errors, the rationale for least-squares fitting of a response surface is not clear. Of course, least squares can be viewed as curve fitting and not necessarily employing or relying on the assumption that the departures (differences between the response and the regression model) behave like white noise. The usual problem of choosing the regression model is compounded if the response is complex. Moreover, the fit will not generally interpolate the observed data (where the function is known exactly) unless there are as many estimable coefficients in the regression as there are runs.

Despite some similarities to physical experiments, then, the lack of random (or replication) error leads to important distinctions. In deterministic computer experiments:

- The adequacy of a response-surface model fitted to the observed data is determined solely by systematic bias.
- The absence of random error allows the complexity of the computer model to emerge.
- Usual measures of uncertainty derived from least-squares residuals have no obvious statistical meaning. Though deterministic measures of uncertainty are available (e.g., $\max |\hat{y}(x) - y(x)|$ over x and a class of y's), they may be very difficult to compute.
- Classical notions of experimental unit, blocking, replication and randomization are irrelevant.

While the pioneering work of Box and Draper (1959) has relevance to the first of these points, it is unclear that current methodologies for the design and analysis of physical experiments [e.g., Box and Draper, 1987; Box, Hunter and Hunter, 1978; Fisher (1935); and Kiefer (1985)] are ideal for complex, deterministic computer models. Lest the reader wonder whether statistics has *any* role here, we assert that:

- The selection of inputs at which to run a computer code is still an experimental design problem.
- Statistical principles and attitudes to data analysis are helpful however the data are generated.
- There is uncertainty associated with predictions from fitted models, and the quantification of uncertainty is a statistical problem.
- Modeling a computer code as if it were a realization of a stochastic process, the approach taken below, gives a basis for the quantification of uncertainty and a statistical framework for design and analysis.

4. MODELING AND PREDICTION

This section discusses models for computer experiments and efficient prediction. Experimental design for this predictor is the subject of the next section.

The model we adopt here treats the deterministic response $y(x)$ as a realization of a random function (stochastic process), $Y(x)$, that includes a regression model,

$$Y(x) = \sum_{j=1}^{k} \beta_j f_j(x) + Z(x). \tag{1}$$

The random process $Z(\cdot)$ is assumed to have mean zero and covariance

$$V(w, x) = \sigma^2 R(w, x)$$

between $Z(w)$ and $Z(x)$, where σ^2 is the process variance and $R(w, x)$ is the correlation. One rationale is that departures of the complex response from the simple regression model, though deterministic, may resemble a sample path of a (suitably chosen) stochastic process $Z(\cdot)$. Alternatively, $Y(\cdot)$ in (1) may be regarded as a Bayesian prior on the true response functions, with the β's either specified a priori or given a prior distribution.

The use of a stochastic process as a prior on a class of functions has a long history. Diaconis (1988) gave an interesting account of early uses (back to H. Poincaré in the 19th century) in one-dimensional interpolation and integration. Suldin (1959, 1960) used Brownian motion and integrals of Brownian motion to develop quadrature formulas in one dimension. Sacks and Ylvisaker (1970) independently considered the same problem for a wider class of processes, and the Brownian motion model has re-emerged in Smale (1985). Corresponding efforts in d dimensions began in Ylvisaker (1975). See Ylvisaker (1987) for a more recent discussion. Sacks and Ylvisaker (1985) used models of the form (1) with added independent measurement error for one-dimensional (physical) experimental design and analysis. Sacks, Schiller and Welch (1989) employed such models (without measurement error) for prediction in computer experiments with multi-dimensional inputs.

One method of analysis for such models is known as kriging (Matheron, 1963). Given a design $S = \{s_1, \cdots, s_n\}$ and data $y_S = \{y(s_1), \cdots, y(s_n)\}'$, consider the linear predictor

$$\hat{y}(x) = c'(x) y_S$$

of $y(x)$ at an untried x. Taking a classical frequentist stance, we can replace y_S by the corresponding random quantity $Y_s = [Y(s_1), \cdots, Y(s_n)]'$, treat $\hat{y}(x)$ as random, and compute the mean squared error of this predictor averaged over the random process. The best linear unbiased predictor (BLUP) is obtained by choosing the $n \times 1$ vector $c(x)$ to minimize

$$\text{MSE}[\hat{y}(x)] = E[c'(x) Y_S - Y(x)]^2 \tag{2}$$

subject to the unbiasedness constraint

$$E[c'(x) Y_S] = E[Y(x)]. \tag{3}$$

Alternatively, a Bayesian approach would predict $y(x)$ by

$$\hat{y}(x) = E[Y(x) \mid y_S], \tag{4}$$

the posterior mean. The frequentist and Bayesian viewpoints will generally lead to different methods and results, except in the special case of a Gaussian process for $Z(\cdot)$ and improper uniform priors on the β's. It is an old result that the BLUP in the Gaussian case is the limit of the Bayes predictor as the prior variances on the β's tend to infinity (e.g., Parzen, 1963, Section 6).

Kimeldorf and Wahba (1970) investigated classes of prior processes for which the Bayes estimate (4) is a smoothing spline. Blight and Ott (1975) used a stochastic process as a Bayesian prior on the departure function for one-dimensional x. Steinberg (1985) and Young (1977) mitigated the effects of model inadequacy by representing $y(x)$ as a polynomial of arbitrarily-high or infinite degree and assigning a Bayesian prior to the coefficients. O'Hagan (1978, Section 3) formulated a general Bayesian approach, in which the prior on $y(x)$ is a general multidimensional Gaussian process. For a more detailed discussion of the Bayesian viewpoint applied to computer experiments see Currin, Mitchell, Morris and Ylvisaker (1988).

In this article, we shall focus mainly on the kriging predictor, partly for ties with methodology in use in other areas and partly to simplify the exposition. Moreover, the use of Gaussian spatial processes provides a bridge to the Bayesian viewpoint. Where the Bayesian view provides additional insight, however, it will be mentioned.

To give some technical details connected with implementing the BLUP of the response at an untried input we use the notation

$$f(x) = [f_1(x), \cdots, f_k(x)]'$$

for the k functions in the regression,

$$F = \begin{pmatrix} f'(s_1) \\ \vdots \\ f'(s_n) \end{pmatrix}$$

for the $n \times k$ expanded design matrix,

$$R = \{R(s_i, s_j)\}, \quad 1 \le i \le n; \ 1 \le j \le n,$$

for the $n \times n$ matrix of stochastic-process correlations between Z's at the design sites, and

$$r(x) = [R(s_1, x), \cdots, R(s_n, x)]'$$

for the vector of correlations between the Z's at the design sites and an untried input x. With these definitions, the MSE (2) is

$$\sigma^2[1 + c'(x) R c(x) - 2c'(x) r(x)], \tag{5}$$

and the unbiasedness constraint (3) is $F'c(x) = f(x)$. Introducing Lagrange multipliers $\lambda(x)$ for the constrained minimization of the MSE, the coefficient $c(x)$ of the BLUP must satisfy

$$\begin{pmatrix} 0 & F' \\ F & R \end{pmatrix} \begin{pmatrix} \lambda(x) \\ c(x) \end{pmatrix} = \begin{pmatrix} f(x) \\ r(x) \end{pmatrix}. \tag{6}$$

Then, by inverting the partitioned matrix, the BLUP can be written as

$$\hat{y}(x) = f'(x)\hat{\beta} + r'(x)R^{-1}(Y_S - F\hat{\beta}), \tag{7}$$

where $\hat{\beta} = (F'R^{-1}F)^{-1}F'R^{-1}Y_S$ is the usual generalized least-squares estimate of β. Under the model, the two terms on the right of (7) are uncorrelated, and the second can be interpreted as a smooth of the residuals. Therefore, the fit can be viewed as two stages: obtain the generalized least-squares predictor and then interpolate the residuals as if there were no regression model.

A convenient representation for the MSE (2) is obtained by substituting (6) in (5) to give

$$\mathrm{MSE}[\hat{y}(x)] = \sigma^2\left[1 - (f'(x)\ r'(x))\begin{pmatrix} 0 & F' \\ F & R \end{pmatrix}^{-1}\begin{pmatrix} f(x) \\ r(x) \end{pmatrix}\right]. \tag{8}$$

Equations (7) and (8) are also the limiting posterior mean and variance of $Y(x)$ when a diffuse prior is placed on the β's.

Of course, the correlation $R(w, x)$ has to be specified to compute any of these quantities. It should reflect the characteristics of the output of the computer code. For a smooth response a covariance function with some derivatives would be preferred, whereas an irregular response might call for a function with no derivatives.

A natural class is the stationary family $R(w, x) = R(w - x)$, presuming that any anticipated nonstationary behavior can be modeled via the regression component. Within this family we restrict attention to correlations $R(w, x) = \prod R_j(w_j - x_j)$, which are products of one-dimensional correlations. Of special interest to us are those of the form

$$R(w, x) = \prod \exp(-\theta_j |w_j - x_j|^p), \tag{9}$$

where $0 < p \leq 2$. (We can also permit p to vary with j.) The case $p = 1$ is the product of Ornstein-Uhlenbeck processes; these are continuous but otherwise not very smooth. The case $p = 2$ gives a process with infinitely differentiable paths (mean square sense) and is useful when the response is analytic.

An alternative correlation function, related to (9) with $p = 1$, is the product of linear correlation functions,

$$R(w, x) = \prod(1 - \theta_j |w_j - x_j|)_+. \tag{10}$$

The predicted response $\hat{y}(x)$ using this correlation is a linear spline. From a one-dimensional correlation function $R_j(x_j, w_j)$, a smoothed correlation can be obtained by integrating,

$$\tilde{R}_j(w_j, x_j) = \int^{w_j}\int^{x_j} R_j(u, v)\,du\,dv.$$

Such correlations are not stationary. However, as shown by Mitchell, Morris and Ylvisaker (1988), stationary versions can be produced by a modified technique. In particular, the cubic correlation on the unit cube

$$\prod[1 - a_j(w_j - x_j)^2 + b_j|w_j - x_j|^3], \tag{11}$$

for certain choices of a_j and b_j, is the stationary version of integrating (10) and produces cubic spline predictors.

The product form of the correlations is especially convenient for some of our computations. This rules out correlations like

$$R(w, x) = \exp(-\theta\|w - x\|), \tag{12}$$

where $\|\cdot\|$ is Euclidean distance in d dimensions, but we are optimistic that the product families already provide enough flexibility for adequate prediction in most cases.

Given the family of correlations, there still remains the question of selecting or estimating the parameters of the family [θ_j and p in (9) say]. In Currin, Mitchell, Morris and Ylvisaker (1988) and Sacks, Schiller and Welch (1989), we have found that cross validation and maximum likelihood estimation (MLE) are useful at the analysis stage (i.e., after data have been collected) and in data-adaptive sequential design (see Section 5).

Assuming a Gaussian process, the likelihood is a function of the β's in the regression model, the process variance σ^2, and the correlation parameters. Given the correlation parameters, the MLE of the β's is the generalized least-squares estimate, and the MLE of σ^2 is

$$\hat{\sigma}^2 = \frac{1}{n}(y_S - F\hat{\beta})'R^{-1}(y_S - F\hat{\beta}).$$

With these definitions of $\hat{\beta}$ and $\hat{\sigma}^2$, the problem is to minimize $(\det R)^{1/n}\hat{\sigma}^2$, which is a function of only the correlation parameters and the data.

5. EXPERIMENTAL DESIGN

5.1. Introduction

The design of deterministic computer experiments has been partly addressed in the literature. For example, Sacks and Ylvisaker (1984, 1985), Welch (1983) and references mentioned therein have considered nonparametric systematic departures from regression models. Random error is also included, but the resulting sampling-variance contribution to mean squared error can be set to zero, and these approaches have helped shape our formulation. For the most part, however, the designs used for fitting predictors have been those developed for physical experiments. Such

designs typically have appealing features of symmetry and are often optimal in one or more senses in settings which include random noise. Their appropriateness for computer experiments, however, is by no means clear. Latin hypercube sampling, discussed in Section 3, is aimed at objectives different from those we have in mind.

There has also been some work in design for numerical integration, where function evaluations can be viewed as a computationally cheap computer experiment. Much is known about design for one-dimensional quadrature. In particular, Sacks and Ylvisaker (1970) constructed good designs (finite n) from asymptotically ($n \to \infty$) optimal designs. These methods, however, do not carry over to $d > 1$ dimensions (see Ylvisaker, 1975). Similarly, in the numerical analysis literature (Davis and Rabinowitz, 1984) results for $d = 1$ offer little guide to $d > 1$.

5.2. Design Criteria

For a fixed number of runs, n, and for specified correlation structure R, we need a criterion for choosing a design that predicts the response well at untried inputs in the experimental region $\mathscr{X}$. Here, we consider functionals of the MSE matrix or kernel

$$M = \{E[Y(w) - \hat{y}(w)][Y(x) - \hat{y}(x)]\}$$

for all w and x in $\mathscr{X}$. The diagonal elements are the $\mathrm{MSE}[\hat{y}(x)]$ given in (8). In the Bayes case when the β's in (1) are known constants, M is just the posterior covariance matrix of the process. When the β's have prior variances that tend to infinity, M is the limiting posterior covariance matrix of $Y(\cdot)$. We now list various criteria based on M.

Integrated Mean Squared Error (IMSE). The IMSE criterion chooses the design S to minimize

$$\int_{\mathscr{X}} \mathrm{MSE}[\hat{y}(x)]\phi(x)\,dx$$

for a given weight function $\phi(x)$. From (8) the IMSE can be written as

$$\sigma^2\left\{1 - \mathrm{trace}\left[\begin{pmatrix} 0 & F' \\ F & R \end{pmatrix}^{-1} \cdot \int \begin{pmatrix} f(x)f'(x) & f(x)r'(x) \\ r(x)f'(x) & r(x)r'(x) \end{pmatrix}\phi(x)\,dx\right]\right\}. \tag{13}$$

These integrals simplify to products of one-dimensional integrals if $\mathscr{X}$ is rectangular and the elements of $f(x)$ and $r(x)$ are products of functions of a single input factor. Thus, polynomial regression models and product correlations can be numerically convenient.

The IMSE criterion is essentially the trace of M (suitably normalized). We assume that $\phi(x)$ is uniform, though other weights cause no real difficulty. This criterion has proved to be effective in terms of *actual* squared error of prediction in test examples reported by Sacks, Schiller and Welch (1989).

Maximum Mean Squared Error (MMSE). Instead of integrating the MSE of prediction, MMSE is a minimax criterion which chooses the design to minimize

$$\max_{x \in \mathscr{X}} \mathrm{MSE}[\hat{y}(x)].$$

Sacks and Schiller (1988) compared IMSE and MMSE for discrete regions. For continuous regions, however, this criterion is computationally complex. It involves a d-dimensional optimization of a function with numerous local optima at every iteration of a given design-optimization algorithm.

Entropy. A criterion advanced by Lindley (1956) in his work on Bayesian design is the minimization of the expected posterior entropy. Shewry and Wynn (1987, 1988) applied it to spatial sampling, and Currin, Mitchell, Morris and Ylvisaker (1988) applied it to the design of computer experiments. It quantifies the "amount of information" in an experiment. In the present setting, if the experimental region $\mathscr{X}$ is discrete, the entropy criterion chooses the design S to minimize $E(-\log g)$, where g is the conditional density of $Y(\cdot)$ on $\bar{S} = \mathscr{X} - S$ given Y_S. Using a classical decomposition of entropy, Shewry and Wynn (1987) showed that minimizing the expected posterior entropy on $\bar{S}$ is equivalent to maximizing the *prior* entropy on S. When $Y(\cdot)$ is Gaussian, this is the same as choosing S to maximize the determinant of V_S, the covariance matrix for $Y(\cdot)$ on S. Straightforward algebra also shows that, in the limiting Bayes case as the prior variances of the β's tend to infinity, maximization of $\det V_S$ is equivalent to maximizing $\det R \cdot \det(F'R^{-1}F)$. If the β's are regarded as fixed (as in Currin, Mitchell, Morris and Ylvisaker, 1988, for the case of a constant prior mean), the last determinant disappears and the entropy criterion reduces to maximization of $\det R$.

5.3. Algorithms

There is no way to implement the ideas set forth above without a method of constructing designs. The utility of D-optimal designs for standard analysis of variance and regression problems with independent experimental errors has only been realized by the development of accessible algorithms (Fedorov, 1972; Mitchell 1974; Welch, 1985; and Wynn, 1970).

Because standard designs can be inefficient or even inappropriate for deterministic computer codes, the need for computer software is even greater. Of course, efficiency has to be weighed against computational cost and convenience. Computer models like the flame code in Section 2, which themselves are expensive to

run on supercomputers, justify the cost of supercomputing in constructing good designs. It is these models we have in mind here. Less effort would be warranted to design for a code that runs on a workstation, say, and so there is also a need for cheap, less sophisticated algorithms.

We now describe some of the algorithms we have used. They can be classified as single-stage methods, sequential methods without adaption to the data, and sequential methods with adaption.

Single-stage design fixes n in advance, and all n design sites are simultaneously optimized according to one (or perhaps a combination) of the above criteria. In addition to standard optimization routines, such as quasi-Newton, a number of exchange algorithms have been tried, primarily when the experimental region is a large, finite grid. At each iteration, an exchange replaces a site in the design by a site that improves the criterion. Currin, Mitchell, Morris and Ylvisaker (1988) adapted Mitchell's (1974) DETMAX excursion algorithm for the entropy criterion. The exchange algorithms used by Shewry and Wynn (1987) exchange sites by adding a random candidate site to the design and deleting the worst site. When the design is close to a (possibly local) optimum, the random candidates are restricted to neighborhoods of the current sites. A simulated annealing algorithm was found useful by Sacks and Schiller (1988) in problems with a small, finite experimental region. For larger problems, the time taken for the annealing process to converge to the optimum was far too long. Simulated annealing algorithms typically require many exchanges and are therefore feasible only when exchanges are cheap. Unfortunately, in our context each exchange may require substantial linear algebra. For continuous regions, we currently prefer standard optimization routines, at least for $n \times d < 100$.

Sequentially designing one site at a time reduces the computational burden from a single $n \times d$-dimensional optimization to a sequence of d-dimensional optimizations. Unlike physical experiments, sequential schemes for computer experiments are no more difficult to organize than a single stage. The design can also adapt to information gathered about the regression model and $R(w, x)$. Furthermore, there is the option of allowing n to be determined as data accumulate, stopping the algorithm as soon as there is sufficient information. Fully sequential design is, therefore, the most natural for computer experiments; unfortunately, it is also the most difficult to treat theoretically.

A sequential design algorithm devised for the IMSE criterion, though *ad hoc*, avoids some pitfalls (see Section 7) encountered in using simple one step look ahead schemes. It starts by dividing the experimental region into a number of subregions or boxes. Each new point is added by computing the contribution to the current IMSE from each box, finding the box with the largest contribution, and adding a point *in that box* that most reduces the contribution *in that box*. The example of the next section exercises this algorithm.

6. CIRCUIT-SIMULATOR EXAMPLE

To illustrate what is already possible, we take a circuit-simulator code similar to that considered by Welch, Yu, Kang and Sacks (1988) and mentioned in Section 2, but differing in the circuit topology. Again, the response is a clock asynchronization or "skew," and we consider six transistor widths as inputs. To avoid getting sidetracked by issues specific to quality control, we do not consider the noise factors here (they are kept fixed at average levels), nor do we perform any circuit-design optimization. We only consider the problem of predicting the clock skew as a function of the six input widths.

The experimental region of interest for the six widths is rectangular, which we transform to the unit cube $[-\frac{1}{2}, \frac{1}{2}]^6$. We assume the model

$$Y(x) = \beta + Z(x), \tag{14}$$

where $Z(\cdot)$ has a correlation function given by (9). This model is selected for various reasons. The regression component includes only the constant β partly because our previous experience in other examples has indicated that this simplification does not affect predictive performance. Moreover, engineering experience does not suggest strong trend over the region of interest. The circuit-simulator clock skew is believed to behave smoothly as a function of the transistor widths; by putting $p = 2$ in (9), a smooth correlation function for $Z(\cdot)$ is obtained. (This initial major assumption of smoothness is revised later by estimating p.) A similar model also gives good predictions when applied to the data in Welch, Yu, Kang and Sacks (1988).

Partly based on our experience with the earlier problem, we allow a total of 32 runs of the simulator for the experimental design. Choosing a single-stage design would mean specifying $\theta_1, \cdots, \theta_6$ and carrying out a 192-variable (6×32) optimization of the design-point coordinates. To reduce the computational burden and to allow adjustment of the model in midstream, we select a first-stage design of 16 points by setting $\theta_1 = \cdots = \theta_6 = 2$ for efficiency-robustness in the sense of Sacks, Schiller and Welch (1989) (described further in Section 7). Optimizing the IMSE over $6 \times 16 = 96$ coordinates using a quasi-Newton library routine takes about 11 minutes on a Cray X-MP. The design, given in the first 16 rows of Table 1, is probably only locally optimal. The

TABLE 1
Experimental design and clock skews for the circuit-simulator example

Run	Experimental design						Skew
1	0.21	−0.26	0.23	−0.21	−0.17	−0.27	−0.972
2	−0.19	0.18	0.22	0.21	0.25	0.28	−0.620
3	−0.19	−0.08	−0.28	−0.28	−0.25	−0.18	−0.711
4	0.19	−0.25	0.28	0.28	−0.06	0.19	−1.040
5	−0.28	0.25	−0.22	−0.21	0.17	0.19	−0.532
6	−0.22	0.21	0.17	0.16	−0.22	−0.22	−0.799
7	−0.22	−0.12	0.27	−0.25	0.23	−0.11	−0.940
8	0.11	0.23	−0.27	0.24	−0.13	0.22	−0.416
9	−0.19	−0.19	−0.19	0.24	0.22	−0.17	−0.500
10	0.17	0.21	0.19	−0.24	−0.20	0.19	−1.293
11	−0.26	−0.24	0.01	0.01	−0.24	0.26	−1.152
12	0.18	0.25	−0.21	−0.21	0.16	−0.28	−0.161
13	0.28	0.18	0.21	0.20	0.25	−0.18	−0.496
14	0.27	−0.18	−0.23	0.21	−0.26	−0.20	−0.612
15	−0.01	0.00	0.00	0.00	0.00	0.01	−0.604
16	0.22	−0.22	−0.17	−0.16	0.21	0.22	−0.897
17	0.10	−0.30	−0.32	−0.38	0.33	−0.30	−0.342
18	0.01	0.31	0.35	0.45	−0.36	0.41	−1.199
19	−0.32	0.45	−0.47	0.44	0.36	−0.28	−0.083
20	−0.27	0.37	0.33	−0.33	0.37	0.30	−1.048
21	−0.41	0.38	−0.32	−0.29	−0.47	0.37	−1.088
22	0.14	0.38	0.36	−0.40	−0.46	−0.49	−0.804
23	−0.15	−0.30	−0.28	0.28	0.29	0.26	−0.444
24	−0.24	−0.36	0.38	0.30	0.35	−0.37	−0.799
25	−0.46	−0.39	0.29	−0.37	−0.46	0.34	−1.918
26	0.17	0.36	−0.26	0.29	−0.41	−0.40	−0.535
27	0.23	−0.20	0.26	0.34	−0.45	−0.27	−1.242
28	0.31	−0.32	−0.25	−0.31	−0.19	0.29	−1.129
29	−0.01	−0.33	0.34	−0.43	0.47	0.37	−1.214
30	0.20	−0.37	−0.36	0.46	−0.45	0.39	−1.049
31	0.21	0.31	0.32	−0.20	0.45	−0.46	−0.135
32	−0.21	0.29	−0.27	0.20	0.40	0.41	−0.256

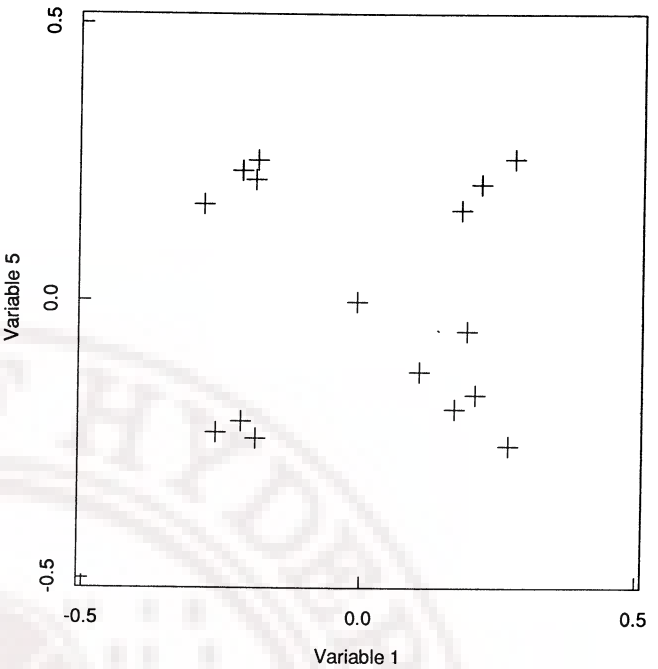


FIG. 1. *Projection of the experimental design onto the coordinates of two input variables.*

projection onto two of the six input coordinates in Figure 1 shows that the design is well away from the boundary, very likely a feature of the IMSE criterion with the constant regression model.

With the data from running the simulator at these 16 points, the MLE of p is 2 (the upper constraint) and those of $\theta_1, \cdots, \theta_6$ are .00, .39, .42, .53, 1.97 and .46. These values are now used in the generation of the second-stage design by the sequential strategy outlined in Section 5. The experimental region is broken into 32 boxes by dividing each of the last five input ranges in half. The first variable is not used to define these boxes as $\hat{\theta}_1 \approx 0$, suggesting that the response is fairly constant (highly correlated) over this factor, though it is still included in the second-stage design. The second set of 16 points, generated one at a time, is given in the second half of Table 1. These points are less concentrated in the center of the design region than the first-stage design, about which we have some misgivings. The MLE of p recomputed from all 32 observations is 1.54, indicating a less-smooth surface than initially thought. The MLEs of $\theta_1, \cdots, \theta_6$ are .00, .06, .19, .34, .14 and .32, again suggesting that the first factor is irrelevant.

To investigate the effectiveness of the BLUP based on this design, we can compare the true responses from the simulator at 100 random points $r_1, \cdots, r_{100}$ in the experimental region with predictions from the BLUP. (We chose a computationally cheap circuit-simulator code to allow this evaluation.) One summary statistic is the empirical integrated squared error

$$\frac{1}{100} \sum [\hat{y}(r_i) - y(r_i)]^2,$$

which equals $(.122)^2$ (relative to a data range of about 2). The maximum absolute discrepancy between the true clock skew and the BLUP over these 100 points is .458. For comparison, a quadratic response surface with 28 unknown coefficients fitted by least squares to the data from our design gives an empirical integrated squared error of $(.674)^2$ and a maximum absolute error of 1.71. This illustrates the potential danger in extrapolating polynomial models, but part of the poor performance may be due to our design, which is not intended for this sort of analysis.

It is also interesting to see whether the MSE (8) of the BLUP is a meaningful indicator of uncertainty in prediction. From the MSEs at the 100 random points (again based on the 32-point MLEs), one can compute standardized residuals $[y(r_i) - \hat{y}(r_i)]/\{\text{MSE}[\hat{y}(r_i)]\}^{1/2}$. The Q–Q plot in Figure 2 shows that these standard-

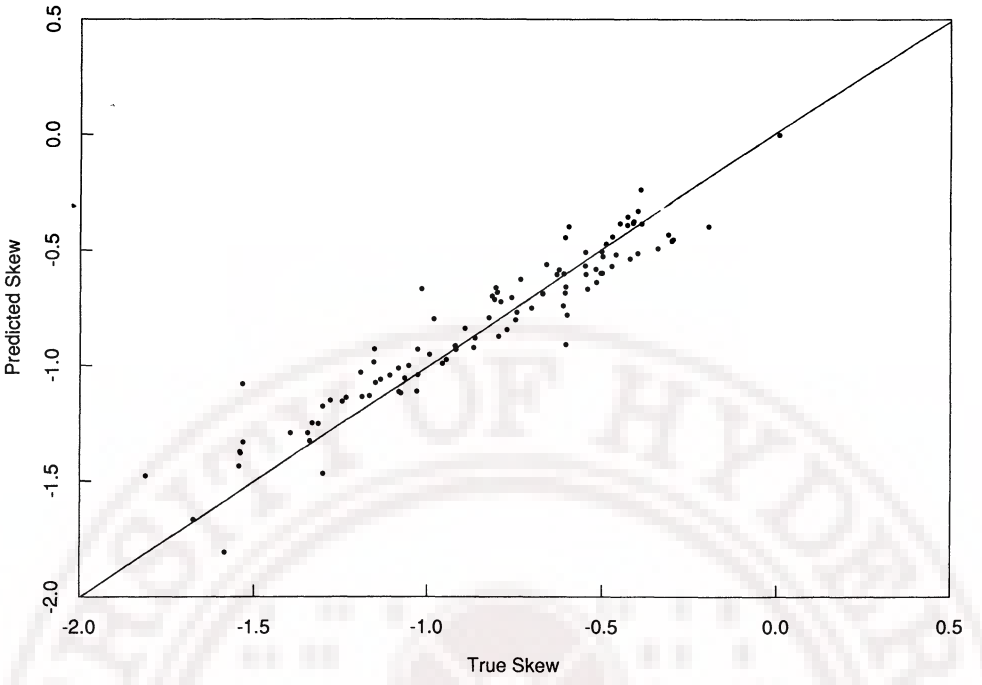


FIG. 2. *Q − Q plot of the standardized residuals against standard normal quantiles.*

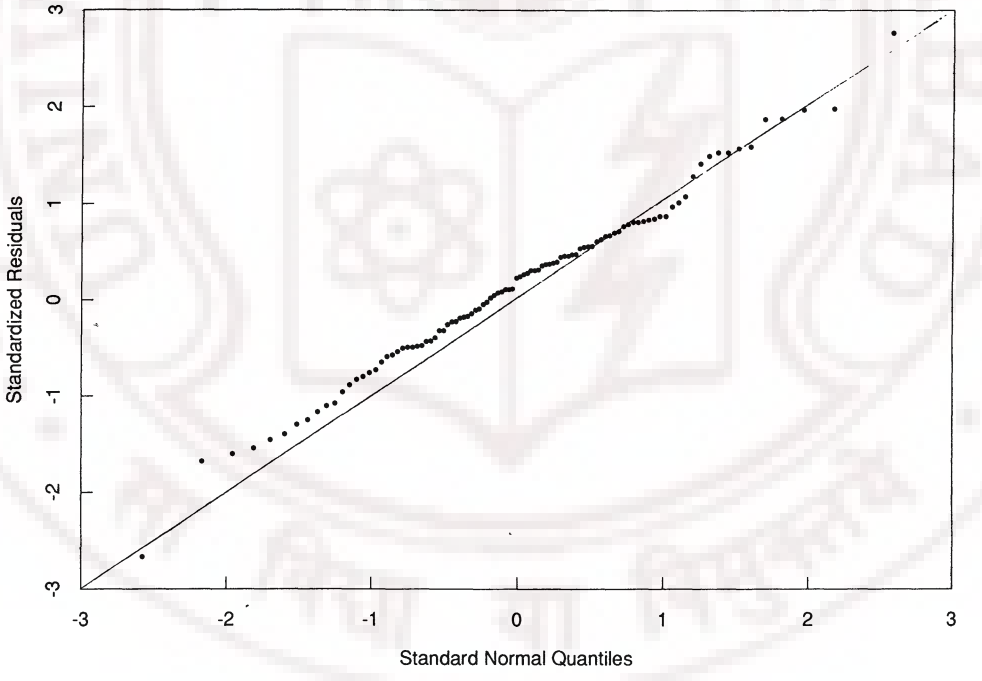


FIG. 3. *Predicted clock skew plotted against true clock skew at* 100 *random sites.*

ized residuals are approximately standard normal, suggesting a central-limit-theorem effect. Also, the slope of the plot is fairly close to 1, indicating that the MSEs do, indeed, provide a valid estimate of error in this example. The plot of $\hat{y}(r_i)$ against $y(r_i)$ in Figure 3 also shows that the poorest predictions tend to be where there are large negative skews. Possibly, the computer code is erratic at such extreme clock skews and harder to predict.

For insight into the relative effects of the six inputs, the response can be decomposed into an average, main effects for each input, two-input interactions and

higher-order interactions. Define the average of $y(x)$ over the experimental region by

$$\mu_0 = \int y(x) \prod_{h=1}^{6} dx_h,$$

the main effect of input x_i (averaged over the other inputs) by

$$\mu_i(x_i) = \int y(x) \prod_{h \neq i} dx_h - \mu_0,$$

the interaction effect of x_i and x_j by

$$\mu_{ij}(x_i, x_j) = \int y(x) \prod_{h \neq i,j} dx_h - \mu_i(x_i) - \mu_j(x_j) - \mu_0,$$

and so on for higher-order interactions. These effects are estimated by replacing $y(x)$ by $\hat{y}(x)$. In the current example, visual inspection of the estimated effects up to two-input interactions suggests that the average, the main effects for factors 2–6, and the interaction of x_4 and x_6 are the important effects. The predictor

$$\hat{\mu}_0 + \hat{\mu}_2(x_2) + \cdots + \hat{\mu}_6(x_6) + \hat{\mu}_{46}(x_4, x_6)$$

gives an empirical squared error of $(.128)^2$, supporting this interpretation.

Using a different design criterion (entropy), algorithm (adaptation of DETMAX), and correlation function [(9) with $p = 1$ at the first stage and (11) at the second stage], Currin, Mitchell, Morris and Ylvisaker (1988) arrived at a design concentrated on the boundary of the experimental region. When used to predict at the same 100 random points, they reported an empirical integrated squared error of $(.163)^2$ and maximum absolute error of .369 over the same 100 random points. Thus, the predictions from this alternative approach are worse on average than the design produced by the IMSE criterion, but the maximum error is better.

7. DISCUSSION

We now summarize a number of open statistical problems that we have discussed only briefly so far and some alternative approaches.

7.1. Simulator Complexity

Almost all of the simulation codes we have worked with are differential-equations solvers. Many of the numerical and other difficulties we have encountered with these codes have implications for the statistical design and analysis.

- A single run of the code may be computationally expensive, for example the 20-minute run time for the flame code (see Section 2), obviously calling for efficient design and analysis.
- The coarse solution to the TWOLAYER code (see Section 2) is a step-like function that may not mirror important features of the accurate solution. An accurate solution is expensive.
- The mathematical model itself may be a poor approximation to reality. For example, the simple, deterministic function used by Taguchi (1986, Chapter 6) for parameter design of a Wheatstone bridge generates negative electrical resistances over part of the region of experimentation. Such aberrant data are misleading and can degrade the analysis. In complex settings, computer-model deficiencies are not so easy to identify. In this article we have largely ignored the problem of validating codes against reality. Rather we have focused on prediction of the computer code itself. Of course, a predicted response that is surprising may help to identify defects in the code.
- The inputs may be of high dimension. This interacts with the first difficulty. If the data are expensive, scientists and statisticians are fully aware of the difficulty in obtaining adequate information about many factors, and screening to reduce dimension is necessary. Thus, expensive data (few runs) and low dimension go together. Cheap data, however, allow many runs, so many factors can be investigated and often are.

7.2. Estimation of Model Parameters

Because the correlation matrix of the data, R, is $n \times n$, the maximum-likelihood computations outlined in Section 4 can be formidable. Vecchia (1988) approximated the likelihood by writing it as a product of conditional densities and conditioning on only a small number of nearest sites. The approximation is cheaper to compute but may retain most of the information.

Properties of the MLE are not well understood and are under study. Mardia and Marshall's (1984) asymptotic results on consistency are not applicable if the region for x is bounded. Their Monte Carlo studies of small-sample behavior indicated substantial variability in the estimates. The validities of the BLUP and measures of uncertainty calculated by substituting MLEs of the correlation parameters therefore appear questionable, but our experience is that even crude MLEs can lead to useful predictions and quantification of uncertainty. Stein (1988) showed that under special circumstances the BLUP can be not only consistent but asymptotically efficient even when the correlation function is misspecified, provided the misspecification leads to a "compatible" Gaussian measure.

7.3. Design Algorithms

All algorithms we have tried for single-stage design are impeded by a number of computational obstacles.

- The optimization is over $n \times d$ design-site coordinates. Though symmetries in the optimal designs are sometimes present, we have not found ways to exploit them to reduce the dimension of the optimization. Since there can be numerous local optima, several tries are necessary.
- Evaluating a "trial" design at each iteration of an optimization algorithm typically involves the solution of a set of at least n linear equations, for example (13) to compute the IMSE. (The vectorizing architectures of computers like the Cray X-MP we have used are ideal for this type of linear algebra, however.)
- The correlation matrix R in (13) (and in other criteria) can be poorly conditioned, and naive rules for cheaply updating the solution from one iteration to the next may lead to numerical errors. For a given correlation function, the conditioning of R becomes worse as n increases.

Thus, the design criteria of Section 5.2 require particularly careful numerical analysis. The computation of D-optimal or other efficient designs for experiments with independent errors shares some of these difficulties, but to a far lesser degree.

As discussed in Section 5.3, sequential design is computationally cheaper and allows adaption to the data. Simple (myopic) sequential strategies of adding the next point to minimize the value of the new design criterion do not work well, however, at least for the IMSE and MMSE criteria. There is a tendency for design sites to eventually "pile up." This may seem counter-intuitive but consider the following example. With the MMSE criterion, take $d = 1$ and $\mathscr{X} = [-\frac{1}{2}, \frac{1}{2}]$. Suppose model (1) has no regression component, and let $Z(\cdot)$ have correlation function $\exp[-(w - x)^2]$. Let the first site, s_1, be placed at zero. If the second site, s_2, is to the left of zero, a straightforward calculation of $\mathrm{MSE}[\hat{y}(x)]$ from (8) shows that the maximum MSE $[\hat{y}(x)]$ occurs at $x = \frac{1}{2}$, and the maximum decreases as s_2 tends to zero. Exact replication does not occur—the limiting design enables $y(0)$ and $y'(0)$ to be evaluated—but this is inefficient relative to the best two-site design. In several dimensions, we have observed that the first few design sites do not pile up in this way, but the same phenomenon eventually occurs. This is not a problem for the entropy criterion, because it places each new design site where the current $\mathrm{MSE}[\hat{y}(x)]$ is maximized, thereby avoiding the neighborhoods of existing design sites.

We described in Section 5.3 a modified sequential algorithm for the IMSE criterion which overcomes this problem by dividing the experimental region. To test the efficiency and running time of this algorithm, we constructed various designs with $9 \le n \le 25$, $p = 1.6$ or 2 in correlation (9), $d = 2, 3$, or 4 dimensions, and constant (β) or first order ($\beta_0 + \sum x_j \beta_j$) regressions. The sequential algorithm required only about 20–30% of the CPU time of a full optimization of all n design sites. Further computational gains would be possible by updating, rather than recomputing, the IMSE as each new site is introduced. Clearly, any sequential scheme without adaption to the data has to be less efficient than an optimal one-stage scheme. Nonetheless, some comparisons show that the efficiency of the designs constructed by the sequential algorithm just described ranges from 40–90%. The lower efficiencies tend to arise when small IMSEs are compared; that is, when n is large, d is small and the regression has just the constant term. Adapting the correlation structure to the data (e.g., by MLE) could lead to sequential methods which outperform one-stage algorithms, especially if the data indicate that some inputs are more important than others.

7.4. Efficiency-Robustness of Designs

Assumptions have to be made about the model for $Y(\cdot)$ and the design criterion. It is natural to ask a number of questions about the efficiency of a design if assumptions change.

- *How sensitive are optimal designs to the choice of correlation structure?*
- *What effect does the regression part of the model have on design?*
- *How do designs chosen by one criterion perform with respect to other criteria?*
- *Are there sub-optimal designs which are robust to choice of criterion?*
- *How important is optimality in this setting?*
- *Are there cheap-to-construct alternatives that perform reasonably well?*

Answers to these questions are limited to a large extent because of the difficulty in computing optimal designs; at the moment we can only refer to some fragmentary, anecdotal results.

Sacks, Schiller and Welch (1989) investigated the effect of the correlation function on the efficiency of the design and predictor. Their study was limited to the effect of the correlation parameter θ within the family (9) with $p = 2$. They computed IMSE-optimal designs for various values of θ. For a given "true" θ, the efficiency of one of these designs, S, relative to the optimal design S_θ was defined to be $\mathrm{IMSE}(S_\theta)/\mathrm{IMSE}(S)$, and there will be some worst-case value of

θ, which minimizes this efficiency. The design that maximizes the worst-case efficiency was deemed to be robust to θ. A further complication is that when evaluating IMSE(S), the BLUP can be based on the true θ or that assumed when generating the design. If the data will be extensive enough to estimate the correlation structure, the true θ may be appropriate, otherwise the assumed θ is retained at the prediction stage. Sacks, Schiller and Welch (1989) considered both cases. Typically, designs for "moderately small" θ resulted. This approach requires computing a number of optimal designs and is limited to problems with $n \times d < 100$, say. For larger problems these efficiency-robust designs can be used, however, to start a sequential scheme.

Currin, Mitchell, Morris and Ylvisaker (1988) implicitly considered robustness of efficiency of the entropy criterion to the correlation structure, although they made no study. In several examples, they designed using (9) with $p = 1$ and θ very large. The intuition was that this prior represents hard-to-predict (low correlation) functions, whereas any reasonable design would deal adequately with easier functions. [There is a connection between designs produced by the entropy criterion as correlations become smaller and those from maximizing the minimum distance between the design sites (Johnson, Moore and Ylvisaker, 1988).] A measure of efficiency based on *differences* in MSEs would lead to a choice of a low-correlation prior, whereas the contrary findings of Sacks, Schiller and Welch (1989) were based on *relative* efficiency.

Sacks and Schiller (1988) investigated the effect of qualitatively different correlation functions—(9) with $p = 2$ versus (12)—on robustness of efficiency. They used MMSE as the criterion, had no regression model and designed on a grid. The θ's of the two correlation functions were chosen to match correlations between Z's at nearest neighbor grid points. This study showed that designs optimal by the MMSE criterion for one correlation were over 80% efficiency for the other (the entries in their Table 3.1 need to be re-ordered). In contrast, we have found that, in predicting two-dimensional integrals, good designs for correlation (9) with $p = 1$ behave poorly in terms of relative efficiency when $p = 2$.

Whether or not a design has robustness of efficiency with respect to alternative correlation functions, the properties of the BLUP will be seriously affected. In particular, higher correlations dramatically increase the apparent precision of prediction. Fortunately, using the data to estimate correlation parameters may lead to effective prediction and reliable estimates of uncertainty (as in the example of Section 6).

The role of the regression model is not yet clear, but it seems to be less important than in design for traditional models with "white noise" errors. Systematic departure from the regression model just becomes part of $Z(\cdot)$, and the BLUP is always an interpolator. In the circuit-simulator experiment, for example, our regression model included only a constant term, yet the predictor appears to follow the true surface, which is clearly not constant, reasonably well. In Example 2 of Sacks, Schiller and Welch (1989) a special class of designs was employed for a methane-combustion code, and it was noted that the effect of the regression model was negligible at the prediction stage. The BLUP was able to adapt to the absence or presence of regression terms: a smaller regression model is compensated for by a covariance function with larger estimated correlations. This phenomenon has some theoretical justification in ongoing work with Y. B. Lim and W. J. Studden on the asymptotic behavior of designs and predictors as the correlation gets large in (9) with $p = 2$.

Sacks and Schiller (1988) found that the entropy and MMSE criteria produce very different designs. The example of Section 6 indicates strong differences between designs from the entropy and IMSE criteria. The entropy criterion tends to push the design sites away from one another, so for small n the optimal design lies on the boundary of the experimental region. As n increases, some interior sites appear—the higher the dimension, the larger n has to be for this to occur. Attraction to the boundary seems not to be a feature of the IMSE and MMSE criteria. In fact, the first 16 runs in Table 1, chosen nonsequentially by IMSE, are well in from the boundary. These remarks are concerned only with the appearance of the designs; we know of no comprehensive investigations of efficiency robustness with respect to the entropy, IMSE, and MMSE criteria. It may turn out that new criteria are necessary, possibly incorporating robustness explicitly.

7.5. Some Alternative Approaches

There are some close connections between the experimental designs produced by the IMSE criterion and previous approaches aimed at minimizing the impact of systematic error in physical experiments. The primary design criterion of Box and Draper (1959, 1963) is also an integrated mean squared error, including components from squared bias and error variance. The variance component turned out to be unimportant for design in the sense that "all-bias" designs that minimize the bias component do fairly well even when the variance component is substantial. Despite modeling the systematic departures by higher-order polynomials rather than a stochastic process, these all-bias designs are qualitatively similar to those from our use of the IMSE criterion, with design points

away from the boundaries of the region of interest. It is plausible that they may be competitive for computer experiments, but the numerical burdens are again extensive.

We have some doubts about transferring least-squares fitting of response surfaces to computer experiments, however. Comparisons can be made by computing the root average squared error or maximum absolute error from test data. In the circuit-simulator example of Section 6, the least-squares quadratic fit is only about 18% and 27% efficient by these criteria relative to the fit from model (14). In this comparison the design constructed for the IMSE criterion was used for both fits. Sacks, Schiller and Welch (1989) reported an example where the least-squares fit to data from a standard factorial design with design points at the boundary of the region of interest had similarly low efficiency.

Our methods are interpolation schemes and could be compared to methods in the numerical analysis literature. The correlation functions (10) and (11) lead to linear and cubic splines. In one dimension, the correlation (9) with $p = 2$ is related to Lagrangian interpolation when θ is small. There is little information in the literature about the construction of good designs for higher-dimensional interpolation.

In the presence of systematic rather than random error, a good experimental design tends to fill out the design space rather than being concentrated on the boundary. Low-discrepancy sequences such as Halton (1960) sequences for numerical integration of non-smooth functions have this "space filling" property (as do Latin hypercube designs). Also, the use made of discrepancy criteria and error bounds based on maximum or average bias are closer in spirit to the approach of this paper than to the randomization bounds of classical Monte Carlo (see Niederreiter, 1978). The efficiencies of these easy-to-generate designs for the objective of prediction should be investigated, especially for very large experimental designs, where criterion optimization may be infeasible.

7.6. Kriging and Spatial Design

In the kriging and spatial statistics literature, the random process $Z(\cdot)$ is often modeled using the variogram $E[Z(w) - Z(x)]^2$ rather than the covariance function. Analogous computational formulas for the BLUP, etc. follow. The variogram permits a wider class of processes, but we are not certain that the added flexibility is needed in our applications. Estimation of the variogram has been studied by several authors; see Cressie (1988) for a recent review.

The data to which spatial methods are applied usually have a two- or three-dimensional x space. They sometimes appear to have measurement error or may be more erratic than responses from computer codes. Geostatistical models used often incorporate a so-called "nugget effect" for erratic local behavior. While we have not addressed such models, it is worth noting that correlation functions of the form (9) with $0 < p \leq 1$ may be useful for modeling such erratic data.

It is not obvious that methods of estimating the variogram extend well from low-dimensional spatial coordinates to the typically high-dimensional inputs of computer experiments. Similarly, results like those in Yfantis, Flatman and Behar (1987) on the properties of regular-grid designs, while interesting for two-dimensional x, are not apparently relevant for computer experiments.

Though we have stressed that deterministic observations are the unique feature of computer experiments, the methodology can be extended to settings where systematic and random error are both important. The covariance function can be adapted so that $\mathrm{Var}[Y(x)] = \sigma^2 + \sigma_\varepsilon^2$, where σ_ε^2 is the variance of the measurement error. (In kriging applications, this can be difficult to distinguish from the nugget effect.) Thus, these approaches should also be useful for physical experiments.

8. CONCLUSIONS

Many scientists feel that statistics is irrelevant to their problems, even for physical experimentation. Their experiments, they claim, have little random variation but are plagued by possibly large systematic biases. These criticisms are not unfounded. There is little easily implemented methodology that addresses systematic error, and the reality might appear even starker for computer experiments with no measurement error. Predictions are nonetheless made with uncertainty, a statistical problem. The stochastic models we have applied to computer experiments quantify uncertainty about the response where it is unobserved and provide a framework for efficient design and analysis, which has been useful in a number of applications.

ACKNOWLEGMENTS

This research program grew out of NSF-funded Workshops on Efficient Data Collection in 1984 and 1985. The authors were supported by AFOSR, NSF Grants DMS-86-09819 and DMS-87-03802, Office of Naval Research Contract N00014-85-K-0357, and the Applied Mathematical Sciences Program of the U.S. Department of Energy through Contract DE-AC05-84OR21400 with Martin Marietta Energy Systems, Inc. Extensive use was made of the Cray X-MP at the National Center for Supercomputing Applications, University of Illinois, partially supported by Cray

Research, Inc. We also acknowledge the contributions of Chona Bernardo, Robert Buck, Carla Currin, Max Morris, Donald Ylvisaker and Tat Kwan Yu and the improvements suggested by the referees.

REFERENCES

BLIGHT, B. J. N. and OTT, L. (1975). A Bayesian approach to model inadequacy for polynomial regression. *Biometrika* **62** 79–88.

BOX, G. E. P. and DRAPER, N. R. (1959). A basis for the selection of a response surface design. *J. Amer. Statist. Assoc.* **54** 622–654.

BOX, G. E. P. and DRAPER, N. R. (1963). The choice of a second order rotatable design. *Biometrika* **50** 335–352.

BOX, G. E. P. and DRAPER, N. R. (1987). *Empirical Model-Building and Response Surfaces.* Wiley, New York.

BOX, G. E. P., HUNTER, W. G. and HUNTER, J. S. (1978). *Statistics for Experimenters.* Wiley, New York.

CRESSIE, N. (1988). Variogram. *Encyclopedia of Statistical Sciences* **9** 489–491. Wiley, New York.

CURRIN, C., MITCHELL, T., MORRIS, M. and YLVISAKER, D. (1988). A Bayesian approach to the design and analysis of computer experiments. ORNL Technical Report 6498, available from the National Technical Information Service, Springfield, Va. 22161.

DAVIS, P. J. and RABINOWITZ, P. (1984). *Methods of Numerical Integration*, 2nd ed. Academic, Orlando, Fla.

DIACONIS, P. (1988). Bayesian numerical analysis. In *Statistical Decision Theory and Related Topics IV* (S. S. Gupta and J. O. Berger, eds.) **1** 163–175. Springer, New York.

FEDOROV, V. V. (1972). *Theory of Optimal Experiments.* Academic, New York.

FISHER, R. A. (1935). *The Design of Experiments.* Oliver and Boyd, Edinburgh.

HALTON, J. H. (1960). On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numer. Math.* **2** 84–90.

IMAN, R. L. and HELTON, J. C. (1988). An investigation of uncertainty and sensitivity analysis techniques for computer models. *Risk Analysis* **8** 71–90.

JOHNSON, M., MOORE, L. and YLVISAKER, D. (1988). Minimax and maximin distance designs. Technical Report, UCLA Statistics Series #13.

KEE, R. J., GRCAR, J. F., SMOOKE, M. D. and MILLER, J. A. (1985). A FORTRAN program for modeling steady laminar one-dimensional premixed flames. Sandia Report SAND85-8240, available from the National Technical Information Service, Springfield, Va. 22161.

KIEFER, J. C. (1985). *Jack Carl Kiefer Collected Papers* **3**: *Design of Experiments* (L. D. Brown, I. Olkin, J. Sacks, and H. P. Wynn, eds.). Springer, New York.

KIMELDORF, G. S. and WAHBA, G. (1970). A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. *Ann. Math. Statist.* **41** 495–502.

KLEIJNEN, J. P. C. (1987). *Statistical Tools for Simulation Practitioners.* Dekker, New York.

LINDLEY, D. V. (1956). On a measure of the information provided by an experiment. *Ann. Math. Statist.* **27** 986–1005.

MARDIA, K. V. and MARSHALL, R. J. (1984). Maximum likelihood estimation of models for residual covariance in spatial regression. *Biometrika* **71** 135–146.

MATHERON, G. (1963). Principles of geostatistics. *Economic Geology* **58** 1246–1266.

MCKAY, M. D., CONOVER, W. J. and BECKMAN, R. J. (1979). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics* **21** 239–245.

MITCHELL, T., MORRIS, M. and YLVISAKER, D. (1988). Existence of smoothed stationary processes on an interval. Unpublished manuscript.

MITCHELL, T. J. (1974). An algorithm for the construction of "*D*-optimal" experimental designs. *Technometrics* **16** 203–210.

NASSIF, S. R., STROJWAS, A. J. and DIRECTOR, S. W. (1984). FABRICS II: A statistically based IC fabrication process simulator. *IEEE Trans. Computer-Aided Design* **CAD-3** 40–46.

NIEDERREITER, H. (1978). Quasi-Monte Carlo methods and pseudo-random numbers. *Bull. Amer. Math. Soc.* **84** 957–1041.

O'HAGAN, A. (1978). Curve fitting and optimal design for prediction (with discussion). *J. Roy. Statist. Soc. Ser. B* **40** 1–42.

PARZEN, E. (1963). A new approach to the synthesis of optimal smoothing and prediction systems. In *Mathematical Optimization Techniques* (R. Bellman, ed.) 75–108. Univ. California Press, Berkeley.

SACKS, J. and SCHILLER, S. (1988). Spatial designs. In *Statistical Decision Theory and Related Topics IV* (S. S. Gupta and J. O. Berger, eds.) **2** 385–399. Springer, New York.

SACKS, J., SCHILLER, S. B. and WELCH, W. J. (1989). Designs for computer experiments. *Technometrics* **31** 41–47.

SACKS, J. and YLVISAKER, D. (1970). Statistical designs and integral approximation. In *Proc. 12th Bien. Sem. Canad. Math. Congress* (R. Pyke, ed.) 115–136. Canadian Mathematical Congress, Montreal.

SACKS, J. and YLVISAKER, D. (1984). Some model robust designs in regression. *Ann. Statist.* **12** 1324–1348.

SACKS, J. and YLVISAKER, D. (1985). Model robust design in regression: Bayes theory. In *Proc. of the Berkeley Conference in Honor of Jerzy Neyman and Jack Kiefer* (L. M. Le Cam and R. A. Olshen, eds.) **2** 667–679. Wadsworth, Monterey, Calif.

SHEWRY, M. C. and WYNN, H. P. (1987). Maximum entropy sampling. *J. Appl. Statist.* **14** 165–170.

SHEWRY, M. C. and WYNN, H. P. (1988). Maximum entropy sampling and simulation codes. *Proc. 12th World Congress on Scientific Computation, IMAC88* **2** 517–519.

SINGHAL, K. and PINEL, J. F. (1981). Statistical design centering and tolerancing using parametric sampling. *IEEE Trans. Circuits and Systems* **CAS-28** 692-702.

SMALE, S. (1985). On the efficiency of algorithms of analysis. *Bull. Amer. Math. Soc.* (*N.S.*) **13** 87–121.

STEIN, M. L. (1988). Asymptotically efficient prediction of a random field with a misspecified covariance function. *Ann. Statist.* **16** 55–63.

STEINBERG, D. M. (1985). Model robust response surface designs: Scaling two-level factorials. *Biometrika* **72** 513–526.

SULDIN, A. V. (1959). Wiener measure and its applications to approximation methods. I. *Izv. Vyssh. Uchebn. Zaved. Mat.* **6**(13) 145–158. (In Russian.)

SULDIN, A. V. (1960). Wiener measure and its applications to approximation methods. II. *Izv. Vyssh. Uchebn. Zaved. Mat.* **5**(18) 165–179. (In Russian.)

TAGUCHI, G. (1986). *Introduction to Quality Engineering.* Asian Productivity Organization, Tokyo.

VECCHIA, A. V. (1988). Estimation and model identification for continuous spatial process. *J. Roy. Statist. Soc. Ser. B* **50** 297–312.

WELCH, W. J. (1983). A mean squared error criterion for the design of experiments. *Biometrika* **70** 205–213.

WELCH, W. J. (1985). ACED: Algorithms for the construction of experimental designs. *Amer. Statist.* **39** 146.

WELCH, W. J., YU, T. K., KANG, S. M. and SACKS, J. (1988). Computer experiments for quality control by parameter design. Technical Report No. 4, Dept. Statistics, Univ. Illinois.

WYNN, H. P. (1970). The sequential generation of D-optimum experimental designs. *Ann. Math. Statist.* **41** 1655–1664.

YFANTIS, E. A., FLATMAN, G. T. and BEHAR, J. V. (1987). Efficiency of kriging estimation for square, triangular and hexagonal grids. *Math. Geol.* **19** 183–205.

YLVISAKER, D. (1975). Designs on random fields. In *A Survey of Statistical Design and Linear Models* (J. N. Srivastava, ed.) 593–607. North-Holland, Amsterdam.

YLVISAKER, D. (1987). Prediction and design. *Ann. Statist.* **15** 1–19.

YOUNG, A. S. (1977). A Bayesian approach to prediction using polynomials. *Biometrika* **64** 309–317.

Comment

Max D. Morris

The authors have provided an interesting and readable account of a statistical approach to the problem of approximating an unknown, deterministic computer model. The approximation of unknown functions, of at least a few arguments, has received considerable attention in other specialty areas of mathematics, but is relatively new to statistics. A statistical approach brings a unique potential for dealing with uncertainty in the problem. In particular, it can lead to measures of quality for each prediction, and a structure on which to base the design of efficient experiments. Techniques which are relevant for approximating computer models are particularly timely, because the scientific and technical professions are quickly becoming reliant upon these as research tools, and this manuscript reports some of the first serious efforts to make statistics relevant to these activities.

THE CLASSICAL APPROACH

At the end of Section 3, the authors give their basic argument for treating this problem statistically: "Modeling a computer code as if it were a realization of a stochastic process . . . gives a basis for the quantification of uncertainty . . ." Following this, Section 4 outlines their strategy which seems clearly classical (as opposed to Bayesian) in form; it is what a classical statistician would do if the computer model actually had been generated as a realization of the stochastic process. While this strategy does provide a mathematical structure for dealing with uncertainty, classical statisticians who like to motivate their analyses with fictional accounts of random sampling and hypothetical replays of an experiment may find this an uncomfortable setting. After all, unless one randomizes the experimental design, there will not be a credible frequentist probability structure in this problem. (My own usual preference for classical procedures is heavily dependent on credible frequentist models. In this problem, the Bayesian approach seems somewhat more direct to me.)

A classical statistician, in order to proceed, will need to be more pragmatic, by saying that a credible frequentist model is unnecessary so long as the method works. The first test of whether the method works is whether it produces good approximations to computer models. These authors, and others they have referenced, have assembled a body of evidence that indicates that this and similar methods have the potential to produce good approximations. The second test, which should be of particular concern to statisticians, is whether it produces good (useful, dependable, meaningful?) measures of uncertainty. Passing this second test will be important if we are to take seriously any claims of quantified prediction uncertainty or design optimality. It is encouraging that the mean square errors of prediction calculated in the example of Section 6 seem to behave as we would hope.

CHOICE OF CORRELATION FUNCTION

As the authors point out in Section 4, the hopes of the pragmatic classical statistician will be pinned on the supposition that the computational model "though deterministic, may resemble a sample path of a (suitably chosen) stochastic process . . ." So, choosing a suitable stochastic process, presumably one for which y would be a "typical" realization, becomes an issue. This is particularly true for preliminary design purposes (before data are taken from which a correlation structure can be estimated). Some guidelines for this selection process are well-known; the authors note that $p = 2$ processes produce smoother realizations than $p = 1$ processes. Also, a tentative value of θ must be chosen for preliminary design purposes; the authors use $\theta = 2$ in the example of Section 6.

When selecting a process in several dimensions, some attention should probably be paid to the degree of interaction among inputs for typical realizations.

Max D. Morris is a Research Staff Member, Mathematical Sciences Section, Oak Ridge National Laboratory, P.O. Box 2009, Oak Ridge, Tennessee 37831-8083.

IMAGE REPRESENTATION: DACE APPROACH

Authors: D.Florance, P.Shalini, Prof. C.Raghavendra Rao, Dr. Rajeev Wankar

ABSTRACT:

An Image is defined as a two-dimensional function which consists of spatial coordinates, and the amplitude of that function at any pair of coordinates is called intensity of the image at that point. Efficient representation and flexible retrieval are two major needs in image processing applications such as image compression, image indexing, face recognition and computer vision. Image representation is a crucial factor in fulfilling these requirements. Image representation and retrieval is a deterministic computer experiment as multiple evaluations produce the same value of the image for fixed inputs.

Design and Analysis of Computer Experiment abbreviated as DACE is used to construct a kriging approximation model based on data from a computer experiment, and to use this approximation model as a surrogate for the computer model. This paper represents image through a feature set which are the corresponding parameter values of a DACE model. In this we are normalizing the image and then sampling the image so that the size of the image gets compressed. Using these sample points we are fitting a model for the image using polynomial regression upto fifth degree. The model gives parameters which are used to predict the image at unknown points. The representation of the image through the parameters of the fitted DACE model and its efficiency are presented in this paper.

1. INTRODUCTION:

An important research issue in the field of multimedia data analysis is that of choosing the right representations for the data where representation can be done by images, sounds, video, etc. In this paper representation is done using Image. With the current huge scientific and non-scientific image databases present, it is naive to expect any human to be able to analyze, understand and extract knowledge from it. The process consists of some computational parts that may be automated and actually better executed by computers than by humans. We will focus our research on image features extraction, typically by the use of DACE approach. Next section describes about DACE model, modeling and prediction using the DACE model.

1.1 DACE MODEL:

This paper it describes the use of the software package DACE (Design and Analysis of Computer Experiments), which is a Matlab toolbox for working with kriging approximations to computer models. The typical use of this software is to construct a kriging approximation model based on data from a computer experiment, and to use this approximation model as a surrogate for the computer model. Here, a computer experiment is a collection of pairs of input and responses from runs of a computer model. Both the input and the response from the computer model are likely to be high dimensional. The computer models we address are deterministic, and thus a response from a model lacks random error, i.e., repeated runs for the same input parameters gives the same response from the model. Often the approximation models are needed as a part of a design problem, in which the best set of parameters for running the computer model is determined. This is for example problems where a computer model is fitted to physical data. This design problem is related to the more general problem of predicting output from a computer model at untried inputs.

The statistical approach for computer experiments involves two parts:

1. Design: Find a set of n points, i.e., a design matrix, denoted by D_n in the input space T so that an approximate model can be 'best' constructed by modeling techniques based on the data set that is formed by D_n and the output on D_n. Putting points of D_n uniformly scattered on T is called a space-filling design, or uniform design.

2. Modeling: Fitting highly adaptive models by various modeling techniques. Because of the space-filling nature of the experimental design, an adaptive model which can represent non-linearity as well as provide good prediction capability at untried points is very important.

1.2 MODELLING AND PREDICTION USING DACE MODEL:

Given a set of m design sites $S = [s_1 \ldots s_m]^T$ with s_i belongs to R^n and responses $Y = [y_{1\ldots} y_m]^T$ with y_i belongs to R^q .The data is assumed to satisfy the normalization condition.

$$\mu[S_{:,j}] = 0,\ V[S_{:,j}\ ,\ S_{:,j}] = 1,\ j = 1,\ldots,n,$$

$$\mu[Y_{:,j}] = 0,\ V[Y_{:,}\ ,Y_{:,j}] = 1,\ j = 1,..,q.$$

Where $X_{:,j}$ is the vector given by the jth column in matrix X, and μ [.] V denote respectively the mean and the covariance.

We adopt a model $\hat{y}$ that expresses the deterministic response $y(x)$, as a realization of a regression model F and a random function (stochastic process)

$$\hat{y}_l(x) = F(\beta_{:,l},x) + z_l(x),\ l = 1\ldots q\ .$$

A regression model is a linear combination of p chosen functions. In the existing matlab toolbox provides regression models with polynomials of orders 0, 1 and 2. We have extended degree of the polynomial till five.

The random process z is assumed to have mean zero and covariance

$$E[z_l(w)\ z_l(x)] = \sigma^2_l\, R(\theta,w,x),\ l = 1,\ldots q$$

Between $z(w)$ and $z(x)$, where σ^2_l is the process variance for the lth component of the response and $R(\theta,w,x)$ is the correlation model with parameter θ . In the toolbox correlations are of the form

$R(\theta,w,x) = \pi_{j=1}^{n} R_j(\theta, w_j - x_j)$, with 7 choices .We focused our experiment to exponential correlation function. Then we perform kriging predictor for y.

The aim of kriging is to estimate the value of an unknown real-valued function f, at a point, x*, given the values of a function at some other points, x_1 ,…., x_n. A kriging estimator is said to be linear because the predicted value $\hat{f}(x^*)$ is a linear combination that may be written as

$$\hat{f}(x^*) = \sum \lambda_i\, y(x_i)$$

λ_i depend on the distance of the test point x from observed points (spatial interpolation)

2. ALGORITHM:

1. Read an image which can be in any one format like jpeg, png, tif, etc. For example we initially have taken inbuilt color image named peppers.png which is of size [M N 3].

2. Crop the image inorder to get an efficient and desired outputs incase of large images. Size of the image after cropping is [N1 N2 3].

3. Convert the RGB image to Gray image now image becomes 2 dimensional and size of image is [N1 N2].

4. Sample the gray image with step size of four. The size of image gets compressed to [mx nx]

5. Pass the sampled image along with regression polynomial, correlation, Theta0, lower bound and upper bound values into dacefit function.

6. The dacefit function gives non-linear least-squares fit of a given correlation model to the provided data set and regression model. The fitted model can be used to predict the image at unknown points.

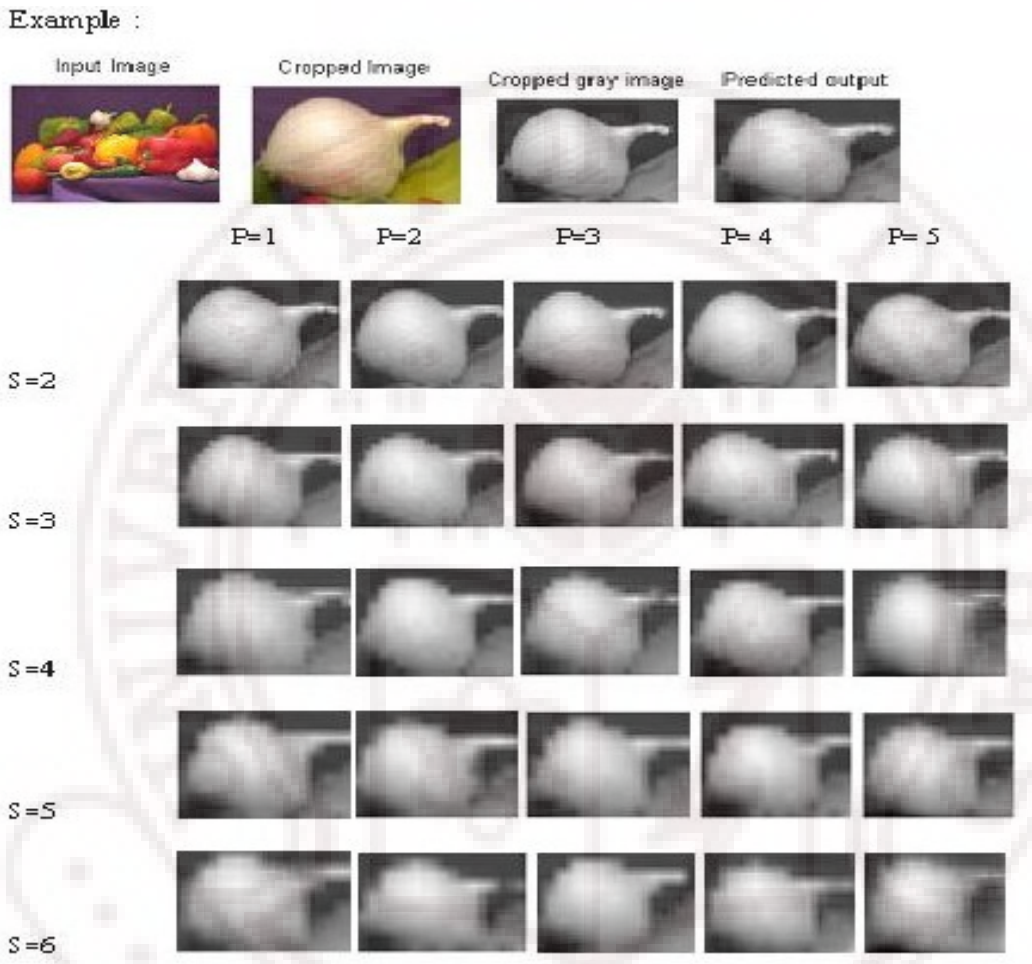
7. Now whole image is passed into predictor function along with the dace model which is output of the dacefit function.

8. Predictor function predicts the response at unknown points based on the dace model and also estimated mean square error of the predictor.

3. CASE STUDY:

3.1 Fixed Input Image and Varying Parameters:

The Input image given to the dace model is an inbuilt image named peppers.png and from the image a small part is been cropped and is been predicted by varying the sampling size of the image with S=2 to S=6 and also varying the regression polynomial with P=1 to P=5

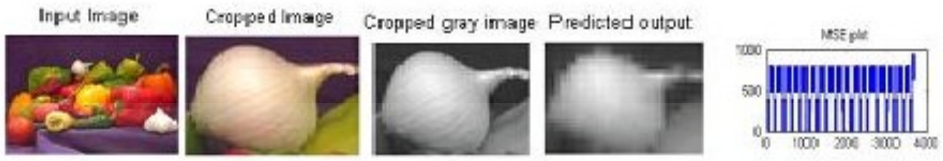


Where 'S' represents the sample size and 'P' represents the regression polynimial used in the dace model program

3.2 Fixed Parameters and Different Input Images

Taking 5 images like pepper.png and 4 other jpeg images as input and with fixed sample size of four and regression polynomial degree five we got the following results.

Example 1:



Example 2:

Example 3:

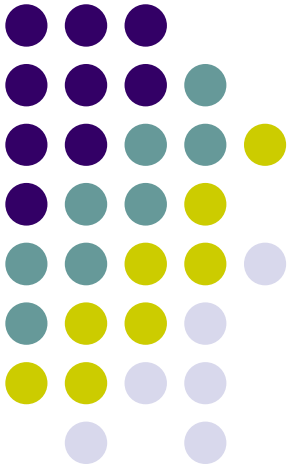
Example 4:

Example 5:

CONCLUSION:

In this paper we have given a naval representation methodology for images using DACE approach. Image is represented through a feature set which are corresponding parameters values of a DACE model. The size of the image gets compressed while sampling and by constructing the dace model the image can be represented with approximately 0.3kb whereas representing with jpeg or any other representation format takes nearly 30kb of size hence representing with dace model gives less storage for image. With the sample size of four and polynomial degree of five we are able to get efficient output. Increasing the regression polynomial degree from three to

ORSI CONFERENCE-2008



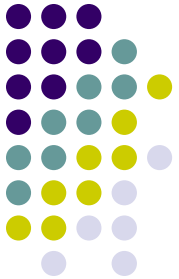


Image Representation: DACE Approach

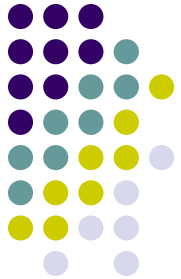
Presented by,
D.Florance (M-Tech)

D.Florance, P.Shalini, Dr. Rajeev Wankar, Prof. C.Raghavendra Rao
University of Hyderabad
Hyderabad-500046

Content

- Introduction
- DACE Model
- Surrogate model
- Kriging Approximation
- Modeling , Estimation and Prediction
- Algorithm
- Case Study
- Conclusion
- Reference

Introduction



- Image representation is a crucial factor in fulfilling the requirements for image processing applications such as image compression, image indexing, face recognition.
- Representing involves two choices:
  1. External characteristics (its boundary)
  2. Internal characteristics (pixels comprising the region)
- Image Representation is a deterministic computer experiment as multiple evaluations produce same pixel value of the image for fixed inputs.

DACE Model

- DACE: Design and Analysis of Computer Experiments
- It is a Matlab toolbox for working with kriging approximations to computer models [1].
- DACE is used to construct a kriging approximation model based on data from a computer experiment, and to use this approximation model as a surrogate for computer model.
- The objective of DACE is to find the design sites and a suitable surrogate model.

[1] Soren N.Lophaven, Hans Bruun Nielsen and Jacob Sondergaard, "DACE A MATLAB Kriging Toolbox", IMM- Technical University of Denmark, August 1, 2002, version 2.0.

Surrogate Model

- Surrogate models are approximation models which replace the behavior of original high fidelity code.
- The objective of surrogate model is to predict the values of deterministic function when its values are known only at a limited finite number of sites.

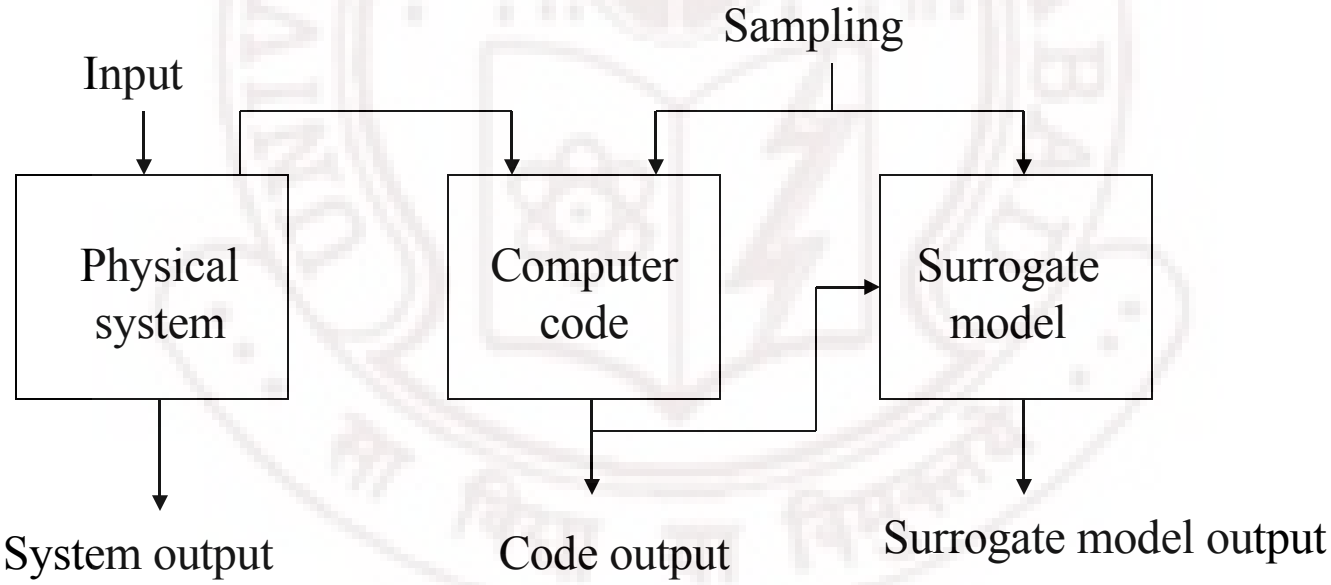
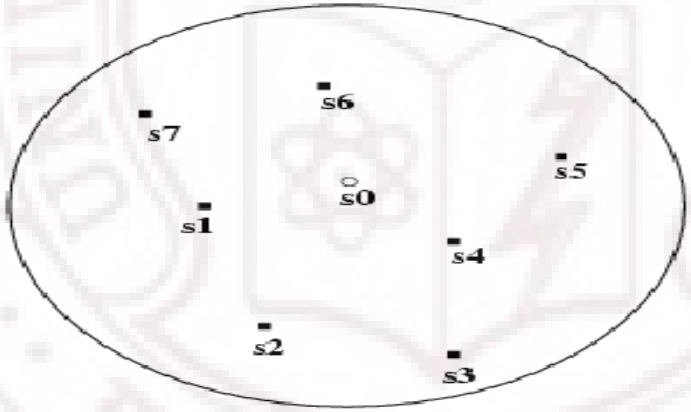


Figure shows the entire philosophy of surrogate modeling in flowchart

Kriging Approximation

- Kriging is a spatial interpolation technique developed by D.G. Krige, a South African mining engineer.
- Kriging gives an approximation method that can give predictions at unknown points of random function.
- Below figure shows the values at points s1, s2, . . . are known and value at point s0 is predicted.



- The entire framework of kriging provides a "model of a model" called as Design and Analysis of Computer Experiments

[7] Amitay Isaacs, "Direct-search methods and DACE", Department of Aerospace Engineering , Indian Institute of Technology Bombay, May 2003.

Modeling

- The Modeling of deterministic computer response y(x) as a realization of a stochastic process Y

$$\mathbf{Y(x) = f^T(x)\,\beta + Z(x)} \quad \text{--------- Eq 1}$$

with: $f(x) = [\, f_1(x), f_2(x), \ldots\ldots f_k(x)]^T$

$$\beta = [\beta_1, \beta_2, \ldots\ldots \beta_k\,]^T$$

- Random Process 'Z' is assumed to have zero mean and constant covariance

$$\text{cov}\,(w,x) = \sigma^2\, R(w,x) \quad \text{----------} \text{ Eq 2}$$

[2] Etman, L.F.P, "Design and analysis of computer experiments: The method of Sacks et al", Engineering Mechanics report WFW 94.098, Eindhoven University of Technology, 1994.

Estimation

- Given $s = \{ s_1, s_2, \ldots\ldots s_N \}$ and response data $y_s = \{ y(s_1), y(s_2), \ldots\ldots y(s_N) \}$

$$\mathbf{Y(x) = f^T(x)\,\beta + \sigma^2\ R(w,x)}$$

- From these computer responses the unknown parameters β and σ^2 can be estimated

$$\beta = (F^T R^{-1} F)^{-1} F^T R^{-1} y_s \text{ ---------- Eq 3}$$

$$\sigma^2 = 1/N\,(y_s - F\beta)^T R^{-1} (y_s - F\beta) \text{ ------ Eq 4}$$

with $F = [f(s_1), f(s_2), \ldots\ldots f(s_N)]^T$

$R = [R(s_i , s_j)]_{ij}$ $1 \le i,j \le N$

[2] Etman, L.F.P, "Design and analysis of computer experiments: The method of Sacks et al", Engineering Mechanics report WFW 94.098, Eindhoven University of Technology, 1994.

Prediction

- The Best linear prediction of the response is

$$\hat{y} = f^T(x)\,\beta + r^T(x)\,\alpha \qquad \text{---------- Eq 4}$$

where $\alpha = R^{-1}(y_s - F\beta)$ -------- Eq 5

and with correlations 'r'

$r = [R(s_1, x), R(s_2, x), \ldots\ldots R(s_N, x)]^T$ ----- Eq 6

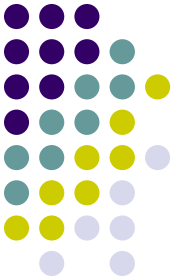
- Calculate mean square error using

$$MSE = \sigma^2 \left(1 + u(x)^T (F^T R^{-1} F)^{-1} u(x) - r(x)^T R^{-1} r(x)\right)$$

where $u(x) = F^T R^{-1} r(x) - f(x)$

[2] Etman, L.F.P, "Design and analysis of computer experiments: The method of Sacks et al", Engineering Mechanics report WFW 94.098, Eindhoven University of Technology, 1994.

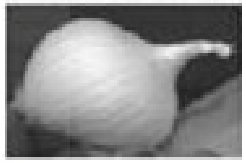
Algorithm



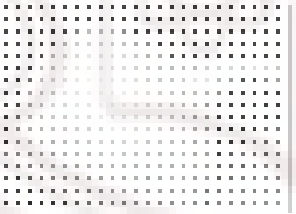
Step 1: Read an input image

Step 2: Crop the desired image

Step 3: Convert RGB image to Gray image



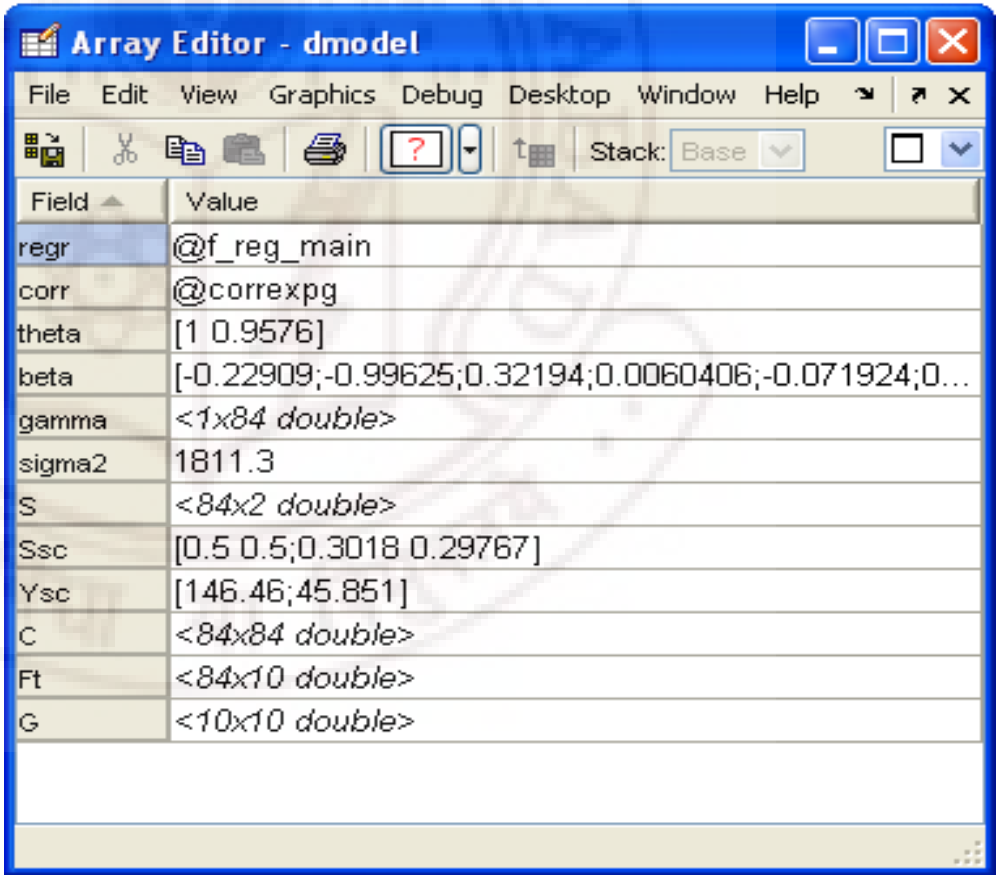
Step 4: Sample the gray image with desired step size



Step 5: Pass the sampled image along with regression polynomial, correlation, Theta0, lower bound and upper bound values into dacefit function.

Contd….

Step 6: The dacefit function gives non-linear least-squares fit of a given correlation model to the provided data set and regression model. The fitted model can be used to predict the image at unknown points.



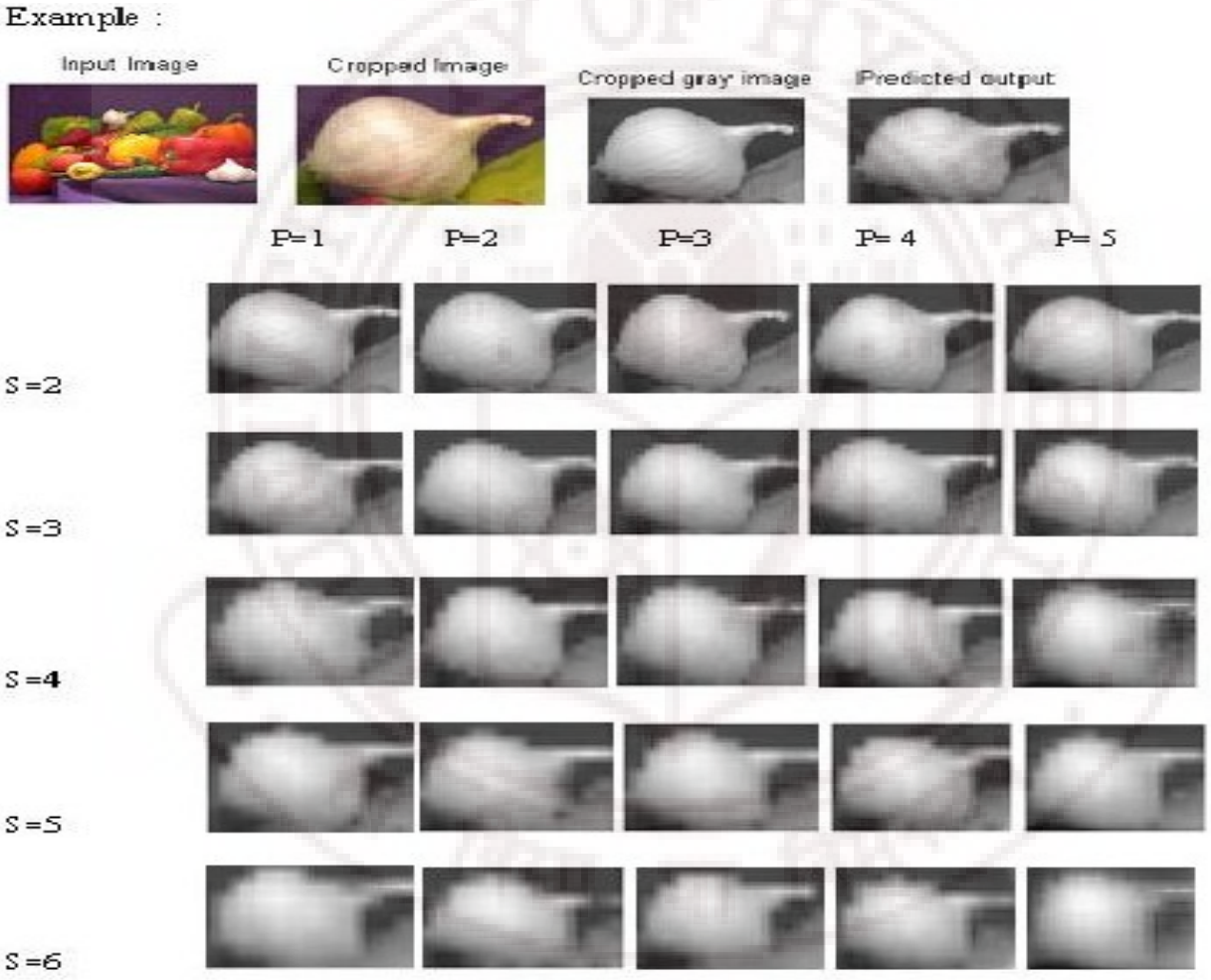
Contd….

Step 7: In predictor function pass the points to be predicted along with the dace model which is output of the dacefit function.

Step 8:Predictor function predicts the response at unknown points based on the dace model and also estimated mean square error of the predictor.

Case Study

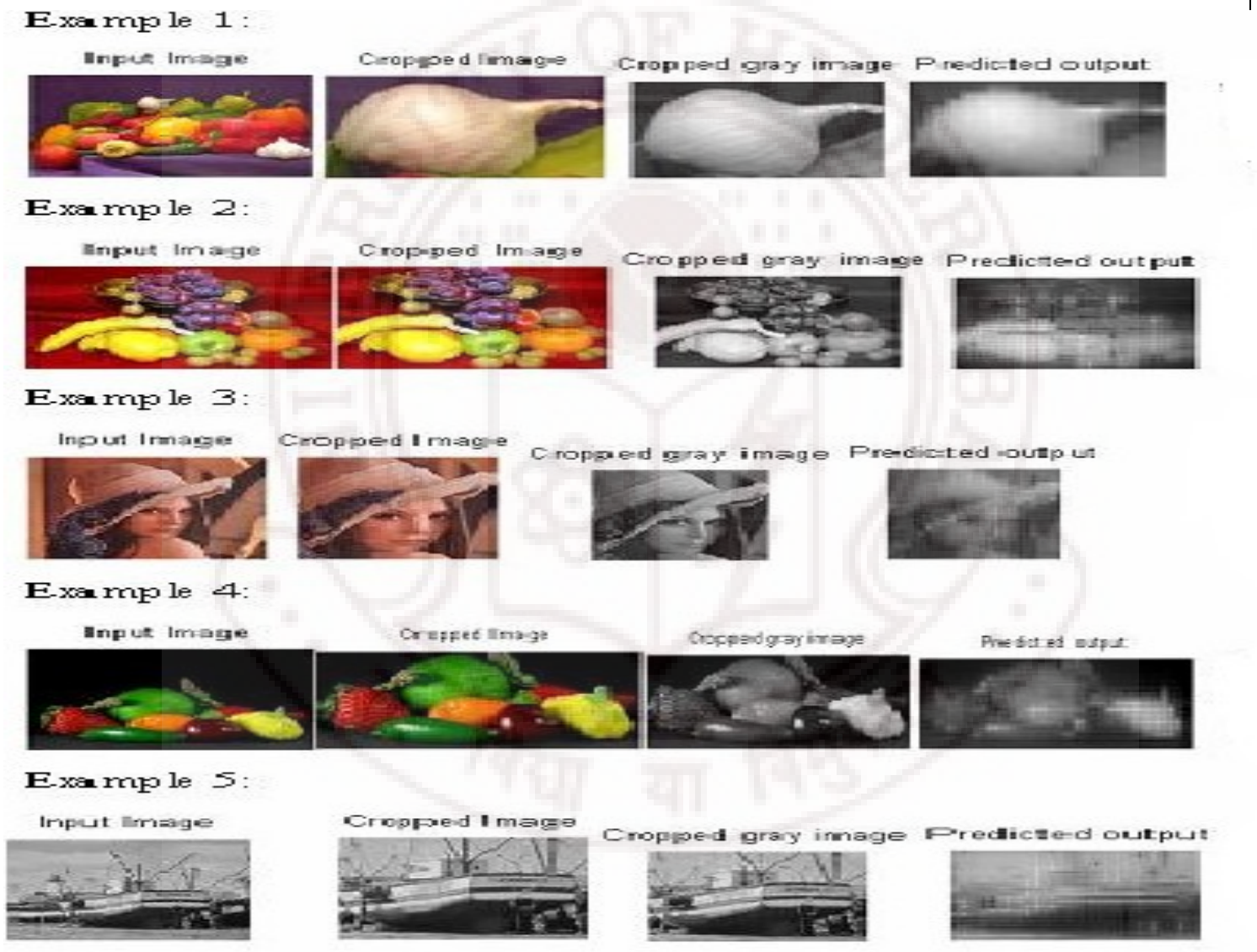
1. Fixed Input Image and Varying parameters



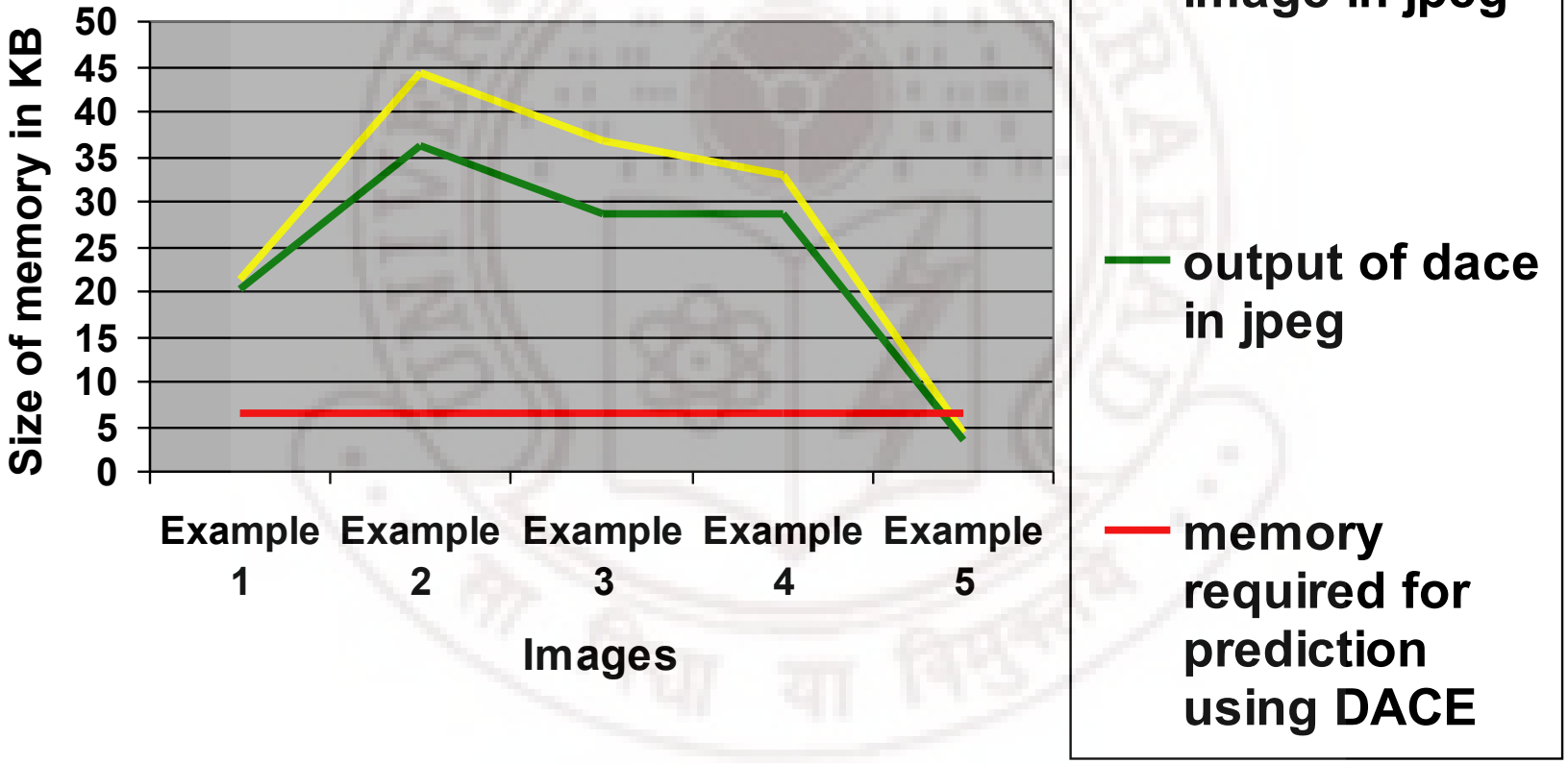
Where ‘S’ is Sample size and ‘P’ represents regression polynomial

Case Study

1. Fixed Parameters and Different Input Images



Comparison of Memory Size



Conclusion

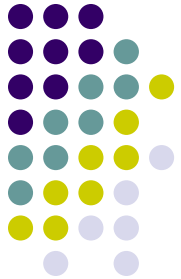
The size of the image gets compressed while sampling and by constructing the dace model the image can be represented with approximately 6.5kb whereas representing with jpeg or any other representation format takes nearly 50kb of size thus representing with dace model gives less storage of memory for image. Image is been compressed for nearly 87% and the output of the predicted image resembles the input image. Further image enhancement techniques can be applied to improve the quality of the result.

References

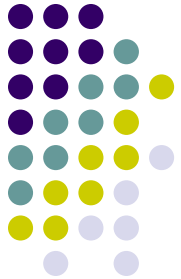
1. Soren N.Lophaven, Hans Bruun Nielsen and Jacob Sondergaard, “DACE A MATLAB Kriging Toolbox”, IMM-Technical University of Denmark, August 1, 2002, version 2.0.
2. Etman, L.F.P, “Design and analysis of computer experiments: The method of Sacks et al”, Engineering Mechanics report WFW 94.098, Eindhoven University of Technology, 1994.
3. Thomas J. Santner, Brian J. Williams, William I. Notz, “The Design and Analysis of Computer Experiments”, Springer series in statistics, 2003.
4. Kai-Tai Fang, Runze Li, and Agus Sudjianto, “Design and Modeling for Computer Experiments”, Computer Science and Data Analysis Series, Published by CRC Press, 2006.

References

- Rafael C. Gonzalez, Richard E. Woods, “Digital Image Processing” Published by Prentice Hall, second edition, 2002.
- J. Sacks, W.J. Welch, T.J. Mitchell, H.P. Wynn, “Design and Analysis of Computer Experiments”, Statistical Science, vol. 4, no. 4, pp. 409-435, 1989.
- Amitay Isaacs, “Direct-search methods and DACE”, Department of Aerospace Engineering , Indian Institute of Technology Bombay, May 2003.



Queries?



Thank You

five has reduced Mean Square Error values of the predicted image get reduced gradually. Increasing the polynomial degree above five has not shown much improvement in the output. Also, reducing the sample size less than four requires more memory to get desired result. Further image enhancement techniques can be applied to improve the quality of the result.

REFERENCES:

[1] Soren N.Lophaven, Hans Bruun Nielsen and Jacob Sondergaard, "DACE A MATLAB Kriging Toolbox", IMM- Technical University of Denmark, August 1, 2002, version 2.0.

[2] Etman, L.F.P, "Design and analysis of computer experiments: The method of Sacks et al", Engineering Mechanics report WFW 94.098, Eindhoven University of Technology, 1994.

[3] Thomas J. Santner, Brian J. Williams, William I. Notz, "The Design and Analysis of Computer Experiments", Springer series in statistics, 2003.

[4] Kai-Tai Fang, Runze Li, and Agus Sudjianto, "Design and Modeling for Computer Experiments", Computer Science and Data Analysis Series, Published by CRC Press, 2006.

[5] Rafael C. Gonzalez, Richard E. Woods, "Digital Image Processing" Published by Prentice Hall, second edition, 2002.

[6] J. Sacks, W.J. Welch, T.J. Mitchell, H.P. Wynn, "Design and Analysis of Computer Experiments", Statistical Science, vol. 4, no. 4, pp. 409-435, 1989.

[7]Timothy W. Simpson, Dennis K.J. Lin, Wei Chen, "Sampling strategies for Computer Experiments: Design and Analysis", International Journal of Reliability and Applications, August, 2001.