Designs for Computer Experiments: Random versus Rational

by

Emelie Anna Gustafsson

B.Sc., The University of British Columbia, 2009

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in

The College of Graduate Studies

(Interdisciplinary Studies)

THE UNIVERSITY OF BRITISH COLUMBIA

(Okanagan)

April 2012

Abstract

Computer experiments facilitate, through mathematical modelling, various experiments that would otherwise be very difficult to perform, or even impossible. Computer experiments are employed to emulate situations in areas such as weather modelling, astrophysics, economics and many more.

In conducting a computer experiment, we are required to select a design for the initial values $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$ of a process, and then emulate the output based on the process and the initial values. The quality of the emulator therefore depends partly on the process and partly on the choice of initial values. We will only consider the Gaussian Process in this thesis, and the focus of our analysis will be on the selection of initial values. We consider the effects of selecting a random versus a rational design for the initial values of computer experiments.

We aim to study a broad range of possible computer models that are likely to arise in practice. Therefore, we present the analysis of eight different design choices, each of which is either random or rational, for the initial values. These initial values are applied to five different test functions that we consider to be prevalent in practice. We observe the effect of each of the designs on each of the test functions at three different sample sizes.

The goal of this thesis is to present a comprehensive study of various standard design choices and make recommendations to a practitioner on a robust design for the initial values of a computer experiment on a given function. As well as ultimately recommend a universally robust design for the initial values of any computer experiment.

Table of Contents

List of Tables

List of Figures

Acknowledgements

I would like to thank all the faculty and staff in the Math, Statistics and Physics department at UBC Okanagan for all the help and support I have received throughout my time here. Special thanks to Dr. Jason Loeppky for his enthusiastic encouragement, guidance and mentorship throughout my Undergraduate and Graduate degrees. I have learned so much over the past years, and I can't express how grateful I am. Many thanks to Dr. Sylvia Esterby and Dr. Paramjit Gill for everything you have taught me as well. I also would like to thank the institution as a whole. UBC Okanagan has given me the opportunity to learn and develop as an individual and as an academic.

University has given me the opportunity to meet so many wonderful people that it would be impossible to mention you all. I have thoroughly enjoyed my time at UBC Okanagan these past few years and I will cherish all the fond memories. Thank you to all my students for the rewarding experience of being your Teaching Assistant.

Lastly and definitely not least I would like to thank my family for their dedication and unwavering support. To my parents, Björn and Marie Gustafsson, thank you for your steadfast support and encouragement throughout my undergraduate and graduate studies. To my sisters, Frida and Pauline, thank you for always being there, it means so much to me. My whole family has been so incredibly supportive and for that I am forever grateful.

Chapter 1

Introduction

Traditional problems in many areas of science and engineering have relied on physical design of experiments to study the effect of various factors on a response. Historically, once physical experiments were not possible due to cost or environmental limits, the experiment would not be completed. With improved computing power we are now able to design computer models that simulate the output of a complex physical system. Computer experiments have applications in most areas of research, as it allows the experimenter to simulate reactions when physical experiments are impossible, such as in astrophysics, economics, biochemistry, modelling weather systems and many other disciplines.

Although computer models allow us to run experiments that are otherwise infeasible, computer models also present some challenges. First, computer experiments are deterministic, meaning that running the code again at the same input $\boldsymbol{x}$ will result in the same output. This is where computer experiments and physical experiments differ significantly. The error in physical experiments is generated by the different outputs from the same input, which is not the case with the deterministic computer experiment. Deterministic computer experiments eliminate concepts such as randomization and replication, and the predictive accuracy of the model is affected solely by the bias in the specification of the computer model. In addition, computer models are often extremely slow to run, meaning that simple Monte-Carlo sampling of the code is not possible for purposes of optimization or sensitivity analysis.

Since the deterministic computer models are usually slow to run, we typically prefer to use a statistical surrogate that can be used to emulate the code output. Approximating the code via a Gaussian Process (GP) is now standard practice (Sacks et al., 1989a,b; Currin et al., 1991) and has proven effective in a large variety of situations. Building a Gaussian Process emulator requires specifying a set of input locations $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$ at which to evaluate the code, where responses $y\left(\boldsymbol{x}^{(1)}\right), \ldots, y\left(\boldsymbol{x}^{(n)}\right)$ are observed. These responses are then used to estimate the unknown parameters in the GP model.

Designing computer experiments requires that we select a design for the initial values $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$ where we wish to run the code. Where $\boldsymbol{x}^{(i)} = (x_1, \ldots, x_d)$ is a d-dimensional vector representing a point in the unit hypercube. The exact specification of the input locations can play an important role in the quality of the estimated model. We consider the effects of selecting a random versus a rational design for computer experiments. A random design is simply taking a random sample from $[0, 1]^d$ to fill the sample space, while the rational designs that we consider are a Sobol sequence as well as six variations of a Latin Hypercube Sample (LHS). Rational designs have been formulated to satisfy different constraints, each with the goal of representing the sample space better than a random sample.

To determine an appropriate sample size for our simulations we use the guideline of $10d$ but also consider smaller and larger values to get a sense of the robustness of a particular design choice. We consider five different test functions, generated by permutations of a fixed design or by a random design. Then we construct a GP model to emulate the output of the function. We run the GP model 50 times for each design at three different sample sizes. In order to assess the performance of the designs we generate the root mean square error and the absolute max error for each run of the GP model. Once we have run all the code, we are able to analyze the prediction errors to select a robust design choice.

Physical experiments and computer experiments are parallel in many aspects; Firstly, the question of experimental design remains. In computer experiments the design problem we face is not to reduce random error from sampling, but rather the selection of initial values for efficient analysis of the data. Secondly, we are still able to apply statistical principles to the analysis of simulated data. Thirdly, although deterministic computer models eliminate the random error that we typically wish to quantify in the analysis of a physical experiment, the statistical problem remains as there is uncertainty associated with prediction from fitted models. Lastly, in modelling the computer code as a Gaussian Process, and in general any stochastic process, we are able to quantify the uncertainty of our predictions, as well as it allows for applications of classical design and analysis (Sacks et al., 1989b).

The goal of this thesis is to present a comprehensive study of various standard design choices and make recommendations to a practitioner on good designs for computer experiments. In order to study a wide range of scenarios we will study various test functions that are believed to represent computer models that are likely to arise in practice.

This thesis is outlined as follows. In Chapter 2 we introduce the Gaussian Process and its applications to computer experiments. In Chapter 4 we outline the procedure for running a Computer Experiment and evaluating the resulting emulations. In Chapter 3 we discuss the various designs we will be considering and the differences between them. Chapters 5 and 6 will outline the results of our simulations, and discuss the differences we find from using the different designs. Chapter 7 will provide conclusions to the results we have found as well as outline areas of possible future research.

Chapter 2

Computer Experiments and the Gaussian Process

Given several input values and one or more (possibly functional) output variables, we can program a complex computer code that mathematically describes the relationship between them. Usually, the computer model of interest is computationally demanding, and scientific objectives like optimization would require too many evaluations if the code is used directly. In favour of efficiency, it has become common practice to adopt computationally efficient statistical approximations (emulations) of the code, as illustrated by Sacks et al. (1989a,b); Currin et al. (1991); O'Hagan (1992).

A homogeneous Gaussian process prior is placed on the possible output functions, which leads to an approximator of the code output given by the posterior mean conditional on the data from the computer experiment. Although the output from the computer model is often multivariate, we will restrict our attention to scalar output. The computer code output is denoted by $y(\boldsymbol{x})$, where the code's vector-valued input, $\boldsymbol{x} = (x_1, \ldots, x_d)$ is assumed to be a point in a d-dimensional unit hypercube $[0,1]^d$.

The GP model places a prior on the class of possible output functions from the computer code. That is, if we let $y(\mathbf{x})$ be the output of the code for a given vector valued input $\boldsymbol{x} = (x_1, \ldots, x_d)$ and let $Y(\boldsymbol{x})$ denote a random function model for $y(\boldsymbol{x})$, the GP model specifies

$$Y(\mathbf{x}) = \mathbf{f}^T(\mathbf{x})\boldsymbol{\beta} + Z(\mathbf{x}). \tag{2.1}$$

Here, $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x}))^T$ contains k known regression functions, $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_k)^T$ is a vector of parameters with unknown values, and $Z(\mathbf{x})$ is Gaussian with mean zero and unknown variance σ^2.

In many cases the regression terms can be replaced by a constant mean, as the Gaussian process will absorb any linear trends in the data (Lim et al., 2002; Steinberg and Bursztyn, 2004). Similarly, Linkletter et al. (2006) and Higdon et al. (2008) have modelled the centred data, $y(\mathbf{x}) - \bar{y}$, with $\bar{y}$ the sample mean of the data for model fitting, as a zero mean process with much success. In what follows we will assume a constant mean $\mathbf{f} = \mu\mathbf{1}$.

The correlation structure of $Z(\cdot)$ is critical to this approach. Let $\mathbf{x}$ and $\mathbf{x}'$ denote the values of two configurations of the input vector, and define the correlation between $Z(\mathbf{x})$ and $Z(\mathbf{x}')$ as

$$\operatorname{Corr}(Z(\mathbf{x}), Z(\mathbf{x}')) = R(\mathbf{x}, \mathbf{x}'). \tag{2.2}$$

The correlation function, $R(\cdot, \cdot)$, depends on a vector of parameters, $\boldsymbol{\theta}$, with unknown values. For a deterministic computer code, $R(\mathbf{x}, \mathbf{x}) = 1$.

Although there are many choices for the correlation function in what follows we consider the most common, which is the Power-Exponential correlation function given as

$$R(\mathbf{x}, \mathbf{x}') = \prod_{j=1}^{d} \exp\left(-\theta_j |x_j - x'_j|^{p_j}\right), \tag{2.3}$$

where $\theta_j \geq 0$ controls the sensitivity of the output function with respect to x_j and $1 \leq p_j \leq 2$ controls the smoothness of the function. When $p_j = 2$ we have the so called Gaussian correlation function which leads to sample paths that are infinitely differentiable and is appropriate for a smooth (analytic) response surface.

From running the computer model n times at input vectors $\mathbf{x}^{(1)}, \ldots,$ $\mathbf{x}^{(n)}$, we have data $\mathbf{y}(\mathbf{x}) = (y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(n)}))^T$ for predicting y. Let $\mathbf{x}^*$ be a new configuration of the input variables, and we want to predict $y(\mathbf{x}^*)$. Conditional on the values of $\boldsymbol{\beta}$, σ^2, and $\boldsymbol{\theta}$, the posterior predictive distribution of $y(\mathbf{x}^*)$ is Gaussian:

$$N(m(\mathbf{x}^*), v(\mathbf{x}^*)), \tag{2.4}$$

where

$$m(\mathbf{x}^*) = \mathbf{f}^T(\mathbf{x}^*)\boldsymbol{\beta} + \mathbf{r}(\mathbf{x}^*)^T \mathbf{R}^{-1}(\mathbf{y} - \mathbf{F}\boldsymbol{\beta}),$$

and

$$v(\mathbf{x}^*) = \sigma^2 \left(1 - \mathbf{r}(\mathbf{x}^*)^T \mathbf{R}^{-1} \mathbf{r}(\mathbf{x}^*)\right).$$

Here, $\mathbf{F}$ is the $n \times k$ matrix with row i containing $\mathbf{f}^T(\mathbf{x}^{(i)})$, the $n \times 1$ vector $\mathbf{r}(\mathbf{x}^*)$ is generated from the correlation function (2.2) with element i given by $R(\mathbf{x}^*, \mathbf{x}^{(i)})$, and the $n \times n$ matrix $\mathbf{R}$ has element i, j from $R(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$.

The full Bayesian formulation requires the specification of prior distributions for the unknown parameter vector $(\boldsymbol{\beta}, \sigma^2, \boldsymbol{\theta})$. For computer experiments, Handcock and Stein (1993) and O'Hagan (1994) assume an improper prior of the form $\pi(\boldsymbol{\beta}, \sigma^2) \propto 1/\sigma^2$ and marginalize $\boldsymbol{\beta}$ and σ^2. The correlation parameters $\boldsymbol{\theta}$ are sampled from the resulting marginal posterior distribution via Markov chain Monte Carlo (MCMC; Gelman et al. 2004).

In what follows we consider fitting the GP to 10,000 data sets in order to assess various different design choices on the estimation of the process. In this situation the MCMC algorithm would be computationally burdensome. As such, we must consider a more efficient method. Following Currin et al. (1991) we adopt an empirical Bayesian approach and estimate the parameters by maximizing the likelihood of the data.

According to the GP model in (2.1), $\mathbf{y}$ is multivariate normal and has the density

$$L(\mathbf{y}\,|\,\boldsymbol{\beta},\sigma^2,\boldsymbol{\theta}) = \frac{1}{(2\pi\sigma^2)^{n/2}\det^{1/2}(\mathbf{R})}\exp\left(-\frac{1}{2\sigma^2}(\mathbf{y}-\mathbf{F}\boldsymbol{\beta})^T\mathbf{R}^{-1}(\mathbf{y}-\mathbf{F}\boldsymbol{\beta})\right). \tag{2.5}$$

Regarding this as a likelihood function with respect to the parameters $\boldsymbol{\beta}$, σ^2, and $\boldsymbol{\theta}$, the MLEs of the GP parameters are straightforward to compute (Sacks et al., 1989b). For any fixed value of $\boldsymbol{\theta}$, the MLEs of $\boldsymbol{\beta}$ and σ^2 are

$$\hat{\boldsymbol{\beta}} = (\mathbf{F}^T\mathbf{R}^{-1}\mathbf{F})^{-1}\mathbf{F}^T\mathbf{R}^{-1}\mathbf{y}$$

and

$$\widehat{\sigma}^2 = \frac{1}{n}(\mathbf{y}-\mathbf{F}\hat{\boldsymbol{\beta}})^T\mathbf{R}^{-1}(\mathbf{y}-\mathbf{F}\hat{\boldsymbol{\beta}}).$$

Substituting these MLEs into the likelihood function (2.5) yields the profile likelihood with respect to $\boldsymbol{\theta}$ only,

$$\frac{1}{(2\pi\widehat{\sigma}^2)^{n/2}\det^{1/2}(\mathbf{R})}\exp(-n/2),$$

which has to be optimized numerically with respect to $\boldsymbol{\theta}$ to give the MLE, $\hat{\boldsymbol{\theta}}$ (e.g., Welch et al., 1992).

In a plug-in predictor, the model parameters are replaced by their MLEs. With $\hat{\boldsymbol{\beta}}$ instead of $\boldsymbol{\beta}$ the conditional mean in (2.4) becomes

$$\hat{m}(\mathbf{x}^*) = \mathbf{f}^T(\mathbf{x}^*)\hat{\boldsymbol{\beta}} + \mathbf{r}(\mathbf{x}^*)^T\mathbf{R}^{-1}(\mathbf{y}-\mathbf{F}\hat{\boldsymbol{\beta}}). \tag{2.6}$$

Extra prediction uncertainty is introduced by using $\hat{\boldsymbol{\beta}}$, and the predictive variance in (2.4) becomes

$$\begin{aligned} v(\mathbf{x}^*) \quad = \quad & \sigma^2(1-\mathbf{r}(\mathbf{x}^*)^T\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*) \qquad (2.7)\\ & +(\mathbf{f}^*-\mathbf{F}^T\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*))^T(\mathbf{F}^T\mathbf{R}^{-1}\mathbf{F})^{-1}(\mathbf{f}^*-\mathbf{F}^T\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}^*))) \end{aligned}$$

where $\mathbf{f}^* = \mathbf{f}(\mathbf{x}^*)$. The extra uncertainty from using $\hat{\boldsymbol{\theta}}$ for $\boldsymbol{\theta}$ to give the plug-in predictor, $\hat{m}(\mathbf{x}^*)$, is not so easily quantified. Similarly, the extra uncertainty in estimating (2.8) by using $\hat{\boldsymbol{\theta}}$ and replacing σ^2 by $\widehat{\sigma}^2$ is often ignored.

The plug-in prediction limits for confidence $1 - \alpha$ are

$$\hat{m}(\mathbf{x}^*) \pm z_{1-\alpha/2}\sqrt{\hat{v}(\mathbf{x}^*)}, \tag{2.8}$$

where $z_{1-\alpha/2}$ is the $1 - \alpha/2$ quantile of the standard normal density.

All that remains is to decide on the actual $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$ locations at which to evaluate the code. This involves specifying a value for n and the exact locations $\boldsymbol{x}^{(i)}$. Loeppky et al. (2009) provide evidence that choices of $n \approx 10d$ are typically sufficient to estimate any reasonably well-behaved function. In our analysis we use the guideline of $10d$ for choice of sample size but also consider smaller and larger values to get a sense of the robustness of a particular design choice. In the following chapter we discuss the various design choices suggested in the literature.

Chapter 3

Designs

Design choices for computer experiments can be broadly characterized into two categories, random or rational. Random designs in this context would consist of a simple random sample of n points uniformly from $[0,1]^d$. Alternatively, rational designs would be based on a more systematic selection of points. Rational designs can be mainly broken down into two categories. First are those that are based on a quasi-random sample of points in $[0,1]^d$. Design choices here involve Sobol sequences (Sobol, 1967) or designs constructed from a Latin hypercube sample (LHS) (McKay et al., 1979) both of which have been shown to be particularly effective for Monte-Carlo estimation of high-dimensional integrals. Second are designs based on statistical criterion that could be better suited for estimation of response surfaces. In this chapter we review the various design choices available for computer experiments. Recall that the computer code is assumed to be deterministic, so that properties such as randomization, blocking and especially replication have no place in modern design for computer experiments. The overall concern for computer experiments is finding designs that adequately sample the unit hypercube. In general we wish to find a design $\mathcal{D}_n$ which is a collection of n points in $\mathcal{F} \equiv [0,1]^d$

3.1 Uniform Designs

Uniform designs (Fang et al., 2000) have been employed with some success in computer experiments. Consider d factors over some region $\mathcal{C}^d$ and we wish to find a design $\mathcal{D}_n \subset \mathcal{C}^d$ such that the points are uniformly scattered on $\mathcal{C}^d$. Finding a good design is typically related to minimizing the discrepancy between the uniform distribution $F(\boldsymbol{x})$ on $\mathcal{C}^d$ and the empirical CDF of the design $\mathcal{D}_n$ which is denoted by $F_n(\boldsymbol{x})$ typically the L_p discrepancy

$$D_p(\mathcal{D}_n) = \left[\int_{\mathcal{D}_n} |F_n(\boldsymbol{x}) - F(\boldsymbol{x})|^p d\boldsymbol{x} \right]^{1/p}$$

is used. One popular choice is the L_∞ norm which is simply

$$D_p(\mathcal{D}_n) = \sup_{\boldsymbol{x} \in \mathcal{D}_n} |F_n(\boldsymbol{x}) - F(\boldsymbol{x})|.$$

In such cases the designs are typically constructed using number theoretic properties or via an algorithm based on a candidate set of points. Since design construction can be extremely time consuming or difficult these designs have not gained significant popularity. There are some catalogues of designs on the unit hypercube, however, these catalogues are very small. For the purposes of the comparisons in Chapters 5 and 6 the existing catalogues do not provide run sizes that are large enough to use these designs for comparisons. Since the goal is to assess designs for the practitioner to use, we do not consider this a serious limitation since these designs would be hard to construct, so practitioners would not be able to readily construct uniform designs in practice.

3.2 Distance Based Designs

Shewry and Wynn (1987) considered choosing designs that maximize the entropy. The criterion of maximum entropy characterizes the amount of information in an experiment introduced by Blackwell (1951) and further expanded by Lindley (1956). In the context of Gaussian processes, Shewry and Wynn (1987) showed that maximum entropy is equivalent to maximizing the determinant of the correlation matrix $\mathbf{R}$ when there is no regression term in the model. Since the correlation parameters are unknown a-priori, the selection of designs using this criterion is not feasible in practice.

Alternatively, Johnson et al. (1990) show that minimizing $1 - \det(\mathbf{R})$ results in maximizing the minimum weighted distance. Specifically, design points are selected using the Maxi-Min weighted distance criterion (Johnson et al., 1990),

$$\max_{\mathcal{D}_n \subset \mathcal{F}} \min_{\mathbf{x}, \mathbf{x}' \in \mathcal{D}_n} \sqrt{\sum_{j=1}^{d} \theta_j (x_j - x_j')^2}\,.$$

Again, since the correlation parameters are unknown this leads to a Maxi-Min distance criterion which chooses a design $\mathcal{D}_n$ such that

$$\max_{\mathcal{D}_n \subset \mathcal{F}} \min_{\mathbf{x}, \mathbf{x}' \in \mathcal{D}_n} \sqrt{\sum_{j=1}^{d} (x_j - x_j')^2}\,.$$

There are a number of appealing features of this criterion. First, it is based on both the maximum entropy criterion and the maximum mean square error criterion. Secondly, from a practical standpoint, it is much more computationally tractable than the other design criteria. Although, Maxi-Min distance designs are intuitively appealing, in practice they tend to push design points to the boundaries of the design space $\mathcal{F}$ which can often result in poor fits to the underlying response surface. In a later section we review a modification of the Latin hypercube based on this distance criterion.

3.3 Latin Hypercube Sampling

Routinely, initial sets of code runs for computer experiments are selected using a latin hypercube sample (LHS)(McKay et al., 1979). These are by far the most common design choice and are extensively used in practice. Latin hypercube samples can be constructed by assuming the input domain for the computer code is $[0, 1]^d$ and assuming a probability distribution function $\mathcal{P}(\boldsymbol{x}) \equiv \prod_{i=1}^{d} \mathcal{P}_i(x_i)$ defined on the hypercube $[0, 1]^d$ which is constructed as the product of independent marginal distributions. A Latin Hypercube sample (McKay et al., 1979) X is an $n \times d$ array in $[0, 1]^d$ where n is the number of runs and d is number of factors. For any column of X, $[0, 1]$ is divided into n non-overlapping stratum with the same marginal probability and one sample is selected at random from each stratum. This is done for every column of X and a random LHS is found by taking permutations of each column. If the input distribution is assumed to be a product of one dimensional uniforms this results in a sampling design in $[0, 1]^d$ where any element x_{ij} of X is of the form $\frac{i+\epsilon_{ij}}{n}, i = 0, \ldots, (n - 1)$ where, $\epsilon_{ij} \sim UNIF[0, 1)$ for all i, j. This results in a random design that achieves an even sampling of points in any one dimension and obtains some degree of space filling in higher dimensions. When the input domain is a hyper-rectangle and the prior distribution on the input is the product of one dimensional marginal distributions the LHS is an extremely straightforward way of obtaining a sampling design.

The plot in Figure 3.1 shows two different random LHS designs in $[0, 1]^2$ with $n = 6$ points. Clearly the design on the left is undesirable, whereas the design on the right is significantly more appealing since it provides a much better coverage of the unit square. In what follows we will discuss various modifications of the LHS designs aimed at improving the space filling nature of the designs.

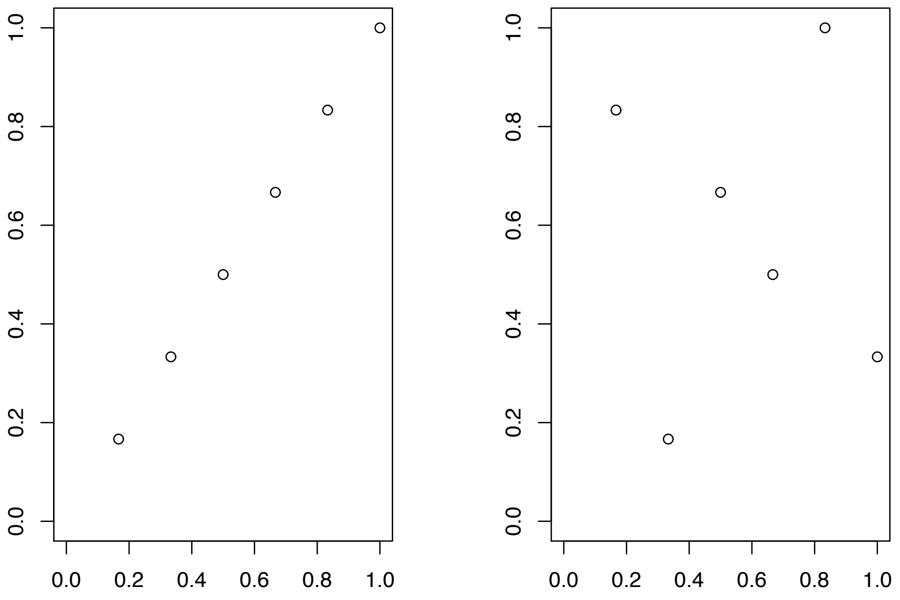


Figure 3.1: Examples of an undesirable random design and a good random design.

3.4 Orthogonal Array Based LHS

Tang (1993) and Owen (1992) describe a modification of a random LHS that is based on using a stratified sampling approach to achieve uniformity in one-dimensional and higher-order dimensional projections. The designs are easily constructed by using an Orthogonal array to provide a method of imposing a stratified sample. These designs are shown to be more efficient than the random LHS and have been shown to be effective at estimating computer models. However, since the designs are based on orthogonal arrays the run sizes are limited to powers of primes or multiples of 4 if a two-level orthogonal array is used. As such, these design choices are not particularly useful for our purposes.

3.5 MaxiMin LHS

Morris and Mitchell (1995) proposed a MaxiMin LHS which imposes a distance based criterion on top of the random LHS. Specifically, the goal is to find an LHS that maximizes the minimum distance between any pair of runs. Imposing the distance measure on top of the LHS prevents design points going to the boundary of the design space. In this way these designs should be ideal for computer experiments since they combine the intuitively appealing statistical aspects of Maxi-Min designs while avoiding the problem of forcing the design points to the boundary of the design space. These designs are easy to construct by using a simulated annealing algorithm or a column pairwise exchange procedure.

3.6 Orthogonal LHS

Iman and Conover (1982) proposed a so-called Orthogonal LHS as a means of improving the space filling nature of a random LHS. Not to be confused with the Orthogonal Array based LHS, a perfectly orthogonal LHS is nothing more than an orthogonal array with a very large number of levels. When an orthogonal array is not available a near orthogonal array can be used in its place. In terms of an LHS sample a nearly orthogonal array can be constructed by finding an LHS where the pairwise correlation among all the columns in the design is zero or near zero. These designs are easy to construct numerically and represent a very easily implemented solution for the practitioner.

3.7 Cosine Transformed LHS

The final variation of the LHS considers a cosine based transformation of the design points. Dette and Pepelyshev (2010) proposed a method of constructing space filling designs that place more points on the boundary of the design space. In particular they consider a transformation of each univariate projection by an arc-sine transformation. They show that given an LHS $\mathcal{D}$ with design points x_{ij}, the resulting arc-sine transformed design Z can be obtained by defining new design points as

$$z_{ij} = \frac{1 - \cos(x_{ij}\pi)}{2}.$$

As seen in Figure 3.2 this results in forcing more points near the boundary of the design space which the authors suggest will result in better estimation of the underlying response surface.

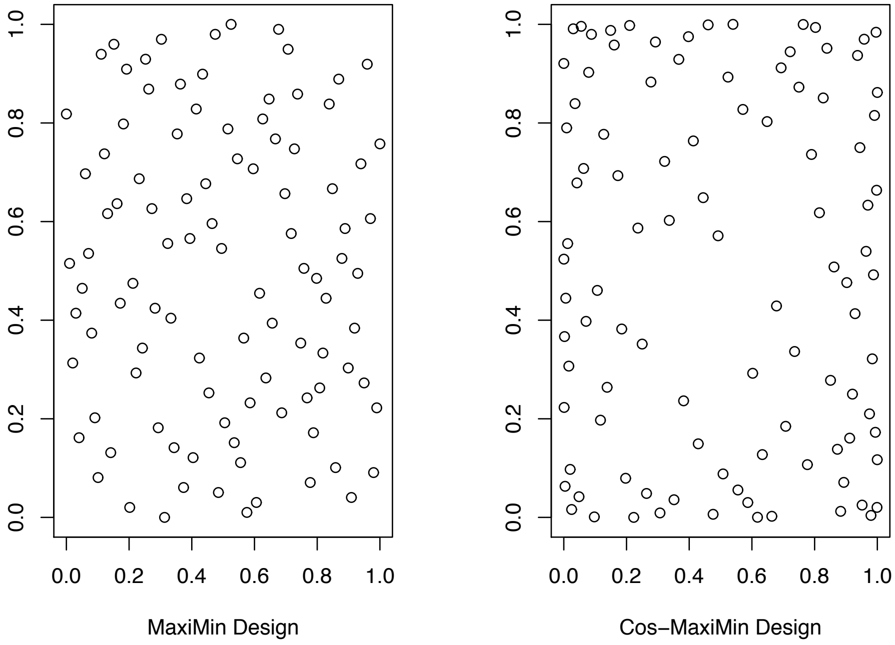


Figure 3.2: Comparing the designs generated under the MaxiMin criterion with the design of the Cos-transform of the same points.

In the comparisons discussed in the next two chapters we consider using the cosine transformation for the Random, MaxiMin and Orthogonal LHS designs.

3.8 Sobol Sampling

Sobol based sampling (Sobol, 1967) is specifically designed for Monte-Carlo integration of high dimensional functions. Specifically, a Sobol sequence is a sequence of n vectors $\boldsymbol{x} = (x_1, \ldots, x_d)$ that fill the d-dimensional unit hypercube $[0, 1]^d$ more uniformly than uncorrelated pseudo-random numbers. Sobol sequences have proven effective for integration and provide space-filling designs in the unit hypercube. Although not designed for computer experiments they have shown some utility for estimation of response surfaces. One of the major advantages of such Sobol sequences is that points can be generated sequentially. If the initial run size proves ineffective at estimating the function it is easy to generate new points. However, the new points are not selected in a data adaptive fashion and as such may not be particularly effective in improving the fitted model.

In the following chapter we will outline a typical approach to running a computer experiment and outline in detail the simulation studies we conduct in Chapters 5 and 6.

Chapter 4

Running a Computer Experiment

In order to present a systematic comparison of the above design choices it is important to clearly outline the process we wish to simulate. A typical computer experiment follows the same general format which we outline below:

1. Specify a sample size n and construct an appropriate design obtaining input locations $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}$ at which to evaluate the code.

2. Run the computer model $f(\boldsymbol{x})$ and obtain the responses $y\left(\boldsymbol{x}^{(1)}\right), \ldots, y\left(\boldsymbol{x}^{(n)}\right)$.

3. Emulate the computer model output using a GP

   (a) Estimate the parameters of the GP using Maximum Likelihood or running the MCMC.

   (b) Specify the posterior predictive process as specified in (2.6) and (2.8)

4. Assess the quality of the emulator using leave-one-out cross validation (Currin et al., 1991).

Since the computer model $f(\boldsymbol{x})$ is assumed deterministic the above model-fitting procedure results in a fixed outcome for any specified set of data. That is, if we want to compare designs the above procedure only results in a single response which will be insufficient to compare different designs. If a comprehensive design study is to be performed it is necessary to replicate the above process and obtain different results at each run. In addition, it would be beneficial to have an external measure of quality of fit.

4.1 External Validation

Cross validation is one method of comparing the quality of the emulator. However, when comparing multiple emulators of the same process it is beneficial to have an external estimate of emulator quality. Assuming the computer model is quick to run, this is possible by evaluating the emulator using out-of-sample predictions. In particular we look at the designs ability to predict a set of m test points. That is, given a collection of m random locations in $[0,1]^d$, observed function values y_i and predicted values $\hat{y}_i$ we compute the Root Mean Squared prediction Error (RMSE) given by

$$RMSE = \sqrt{\frac{\sum_{i=1}^{m}(y_i - \hat{y}_i)^2}{m}} \tag{4.1}$$

as a measure of average performance. Alternatively, a worst case measure of performance is given by the Absolute Max Error (AME)

$$AME = \max_{i=1,\dots,m} |y_i - \hat{y}_i|. \tag{4.2}$$

When comparing different functions or comparing within functions it is necessary to know what constitutes an acceptable error. One potential way of judging if the error is acceptable would be to compare the prediction with a prediction method that uses $\bar{y}$, the mean of the test points, as a predictor. That is, one might consider

$$\frac{RMSE}{Deviation} = 100\sqrt{\frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}}$$

and judge a fit as acceptable when this value is less than 10 and preferably closer to 5.

The analogous measure for the maximum error would be

$$\frac{AME}{MAD} = 100\sqrt{\frac{\max_{i=1,\dots,m}|y_i - \hat{y}_i|}{\max_{i=1,\dots,m}|y_i - \bar{y}|}}$$

where MAD is the maximum absolute deviation about the mean. In this case acceptable errors would yield a value less than 15.

Using out of sample predictions provides a convenient method to compare different emulators of the same process. However, for each chosen design choice we only get one emulator of the process. In order to address this problem we consider two different methods for “replicating” the emulator.

4.1.1 Design Permutations

Given a fixed design $D \subset [0,1]^d$ running the code at D produces a set of n responses. Running the code again at the same D produces the same set of responses. However, given d unique values $x_1, \ldots, x_d$ evaluating the code at $x_1, \ldots, x_d$ and $x_2, x_1, x_3, x_4 \ldots, x_d$ for example would result in two different outputs. That is, the computer model is not invariant to permutations of the inputs. However, each of the designs discussed in Chapter 5 are invariant to column permutations. Thus, given a design D of a particular type and D^* in which the columns of D are permuted then D^* is still a design of the specified type, but evaluating the code at D^* will result in a different set of outputs. Thus permuting the columns of D is one way to "replicate" the experimental process.

4.1.2 Random Functions

In chapter 6 we consider computer models of the form $f(\boldsymbol{x}, \boldsymbol{c})$ where $\boldsymbol{c} = (c_1, \ldots, c_d)$ are a specified set of values. Taking

$$c_i = \frac{c_0 k_i}{\sum_{i=1}^{d} k_i} \quad \text{and} \quad k_i \sim UNIF(0,1)$$

will also result in a form of replication. In this instance the c's are changing which will result in certain dimensions being more or less active in the function, which takes the place of column permutations above. In addition changes of this nature also result in small changes to the function output resulting in a series of different examples all with similar properties.

4.2 Simulation Study

In order to run a full simulation study with replication we use the following procedure

1. Specify a test function
2. Obtain a set of $m = 10,000$ points uniformly in $[0,1]^d$ to use for out-of-sample prediction
3. Run the model to obtain the sample points $y_1, \ldots, y_m$.
4. For each design choice D_i, $i = 1, \ldots, 8$. We consider the eight following designs:

- Latin Hypercube Sampling (LHS)
- MaxiMin LHS
- Orthogonal LHS
- Cos-Transformed LHS
- Cos-Transformed MaxiMin
- Cos-Transformed Orthogonal
- Sobol Sequences
- Random Sample

5. Finally, repeat the following procedure 50 times:

   (a) Run the computer model at D_i^* to obtain data for fitting

   (b) Build the GP emulator (as outlined earlier in the chapter)

   (c) Using the emulator, predict the m points specified in step 2.

   (d) Compute the RMSE and AME resulting in values $RMSE_j$ and AME_j

In Chapter 5 we take D_i^* in step 5a to be a permutation of the columns of design D_i specified in Section 4.1.1. In Chapter 6 we take D_i^* in step 5a to be the design D_i specified in Section 4.1.2. Since the computer model is random there is no need to consider design permutations.

Chapter 5

Design Permutations

The objective of our analysis is to present a comprehensive study of various standard design choices and make recommendations to a practitioner on good designs for computer experiments. In order to study a wide range of scenarios we have chosen five different test functions in $d = 10$ that are believed to represent computer models that are likely to arise in practice.

We apply eight different design choices for the initial values to each of the test functions. We are interested in investigating the differences in the design choices of the initial values of the simulations from each test function. Simulations were made at sample sizes of 70, 100, and 200. We consider eight different design types, the first six are variations of Latin Hypercube Sampling, where the first will be the Random LHS (LH), the second is constructed under the MaxiMin criterion (MM), and the third under the Zero-Correlation (Orthogonal) criterion (OD). The fourth, fifth and sixth will be the same designs again but under a cosine transformation (CL, CM, CO respectively). The last two designs are not based off of Latin Hypercube Sampling; the seventh design is a Sobol Sequence (SS) and finally the eighth design is the basic Random Design (RD).

Each function requires a constant c_i, in this section we define

$$c_i = \frac{c_0 k_i}{\sum_{i=1}^{d} k_i}$$

where $k_i = \frac{i-0.5}{d}, i = 1, \ldots, 10$ and $d = 10$, we select different values of c_0 depending on the test function, we introduce replication in the form of permutation. By permuting the columns of the design as discussed in Section 4.1.1 we are able to generate varying designs with the same properties. In this chapter we introduce and discuss five different functions. For each function we ran simulations 50 times for each design. The output gave the RMSE and AME from each simulation, our analysis will focus on a discussion of the various external validation techniques discussed in Chapter 4.

5.1 Continuous Function

The first test function we will consider is the Continuous function.

$$f_1(x) = \exp\left(-\sum_{i=1}^{d} c_i|x_i - w_i|\right)$$

In our simulations we use $c_0 = 5$ for the Continuous function, and we let $w_i = 0$.

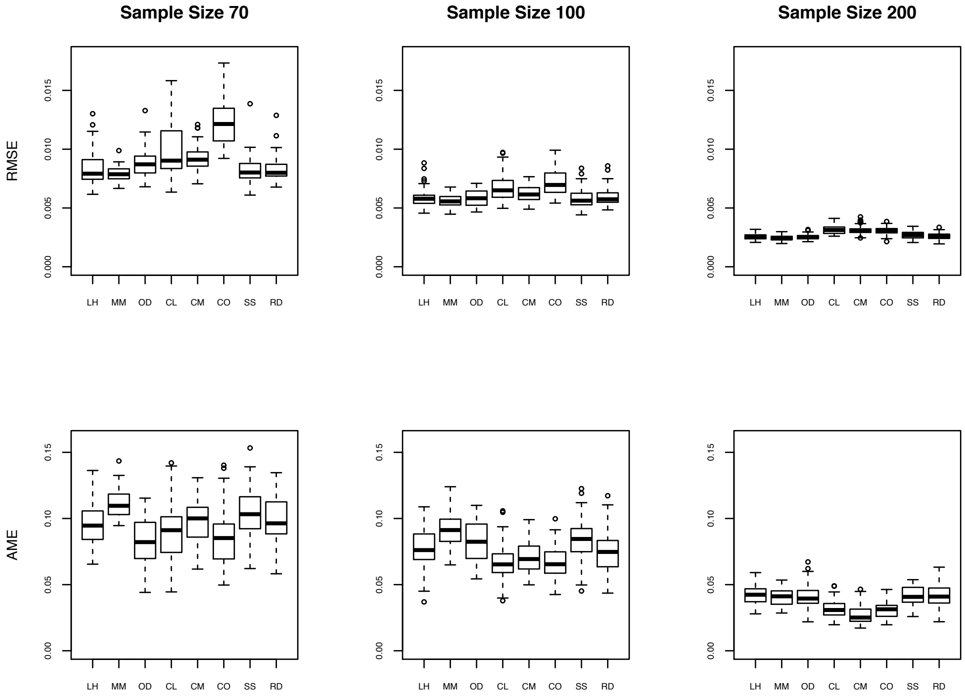


Figure 5.1: Comparing the RMSE and AME generated by the Continuous test function over three different sample sizes using a constant $c_0 = 5$.

In our discussion of the performance of the designs on the Continuous test function, we will first consider the RMSE. The first thing to notice in the first three plots of Figure 5.1 is the variability around the Cos-Transformed designs. Coupled with the largest median values of RMSE across all three sample sizes that we have considered, we can not recommend a Cos-Transformed

Table 5.1: Mean ratio of RMSE/DEV for each design of the Continuous function.

	Sample Size 70	Sample Size 100	Sample Size 200
LHS	16.19	11.28	4.91
MaxiMin	15.23	10.82	4.68
Orthogonal	17.19	11.25	4.86
Cos-LHS	19.19	12.96	6.08
Cos-MaxiMin	17.73	11.96	6.03
Cos-Orthogonal	23.90	13.75	5.90
Sobol	15.66	11.12	5.12
Random	15.90	11.49	5.02

Table 5.2: Mean ratio of AME/MAD for each design of the Continuous function.

	Sample Size 70	Sample Size 100	Sample Size 200
LHS	25.36	20.40	11.15
MaxiMin	29.24	24.01	10.63
Orthogonal	21.84	21.74	10.88
Cos-LHS	23.76	17.57	8.37
Cos-MaxiMin	25.85	18.39	7.07
Cos-Orthogonal	22.47	17.50	8.24
Sobol	27.67	22.12	10.91
Random	25.84	19.55	10.97

design as a good design for this test function. The one design that stands out with low variability and small values of RMSE across all three sample sizes is the MaxiMin design. Also note that with the exception of one outlier, the Sobol sequence performs well on the Continuous function as well. As we increase the sample size all the designs yield smaller values of RMSE as well as the variability around those values decreases dramatically. Even at a sample size of 200 we still notice heightened levels of RMSE as well as variability in the cos-transformed designs.

Similarly, in the lower three plots of Figure 5.1 the AME shows an overall decrease in magnitude as well as variability as the sample size increases. One difference that we notice in the AME versus the RMSE is that the MaxiMin design has the highest magnitude of AME at sample sizes of 70 and 100, and is in the higher range at the sample size of 200. However, the MaxiMin design still has the smallest variability across the sample sizes. A distinct difference that we see between the RMSE plots and the AME plots is the dip in AME for the Cos-Transformed designs at the sample size of 200, which contradicts the “spike” in RMSE that we see at the same sample size.

When faced with discrepancies between RMSE and AME it is important to remember that RMSE is the average error, while the AME is the worst case error. We desire a design with small RMSE and ideally small AME as well, but when the RMSE is small and the AME is larger than we would prefer, we use RMSE as the main criterion for judging a good predictor. If the discrepancies are considerably large, further criterion such as percentiles might be useful in deciding whether the AME is large due to large errors overall, or perhaps just due to one or two outliers. We might consider looking at the number of points above some set limit or the 95^{th} percentile of the observed errors.

Our goal is to find a design that performs well in most situations. To find a design that has a desirable RMSE and find that it does very poorly in the AME is not a reliable design. To further investigate the reliability of the designs, we refer to Table 5.1. A design that yields a ratio for RMSE/Deviation that is less than 10 and preferably closer to 5 will be considered a good design. It is obvious that none of the designs perform well on the Continuous function at a sample size of 70. At a sample size of 100 all of the designs are close to 10 and there is no design that is clearly better than the others. Similarly, at a sample size of 200, all designs appear to perform well, but none are distinctly different from the others. If we were to select a design based on RMSE/Deviation we would select the MaxiMin design as it has the lowest ratios at all sample sizes.

Referring to Table 5.2 consider the ratio of AME/MAD. In this case ratios less than 15 will be considered a good fit. Evidently, no design yields a desirable fit at a sample size of 70. The cos-transformed designs perform marginally better than the others at a sample size of 100. It is not until we reach a sample size of 200 that the designs begin to perform well. Once again we face the issue of no design that is decidedly better than the others. At a sample size of 200 we would recommend the Cos-transformed MaxiMin design solely based on the AME/MAD ratio. However, when the sample size is that large any of the designs will be approximately equivalent.

The question of selecting a robust design remains. Due to the inconsistency in the Continuous function it is difficult to recommend a robust design. Although the MaxiMin design appears to do poorly on the AME and AME/MAD, we would still prefer it since the maximum values on the

Continuous function do not behave well as we discussed earlier, so we are more inclined to rely on the RMSE as an indication of how well the designs perform in this case. Thus, we would recommend the MaxiMin design for the Continuous function.

5.2 Corner Peak Function

The second function we consider is the Corner Peak function.

$$f_2(x) = \left(1 + \sum_{i=1}^{d} c_i x_i\right)^{-(d+1)}$$

In our simulations we investigate the effect of two different constants $c_0 = 0.25$ and $c_0 = 1.85$ for the Corner Peak function.

Table 5.3: Mean ratio of RMSE/DEV for each design of the Corner Peak function with a constant of $c_0 = 0.25$.

	Sample Size 70	Sample Size 100	Sample Size 200
LHS	6.08	3.77	1.28
MaxiMin	5.66	3.66	1.24
Orthogonal	6.06	3.88	1.27
Cos-LHS	7.11	4.34	1.62
Cos-MaxiMin	6.90	4.19	1.56
Cos-Orthogonal	8.07	4.57	1.51
Sobol	5.85	3.70	1.36
Random	5.91	3.88	1.32

We evaluate the Corner Peak function at two different constants in order to better investigate the effect of the constant on the performance of the designs. In Figure 5.2 we see the same general cascading trend in both the RMSE and the AME for the Corner Peak function with a constant of 0.25 as we did in the Continuous function. The difference is in the scales; the scale of RMSE for the Corner Peak function with a constant of 0.25 is less than 0.009 going down to around 0.002 at the largest sample size. While the scale of RMSE for the Continuous function goes from 0.015 to around 0.005 when it is at its lowest at the sample size of 200. Similarly, we see a decrease in the scale of AME, the scale for the Corner Peak goes from 0.07 to around 0.02 at a sample size of 200. While the scale for the AME of the

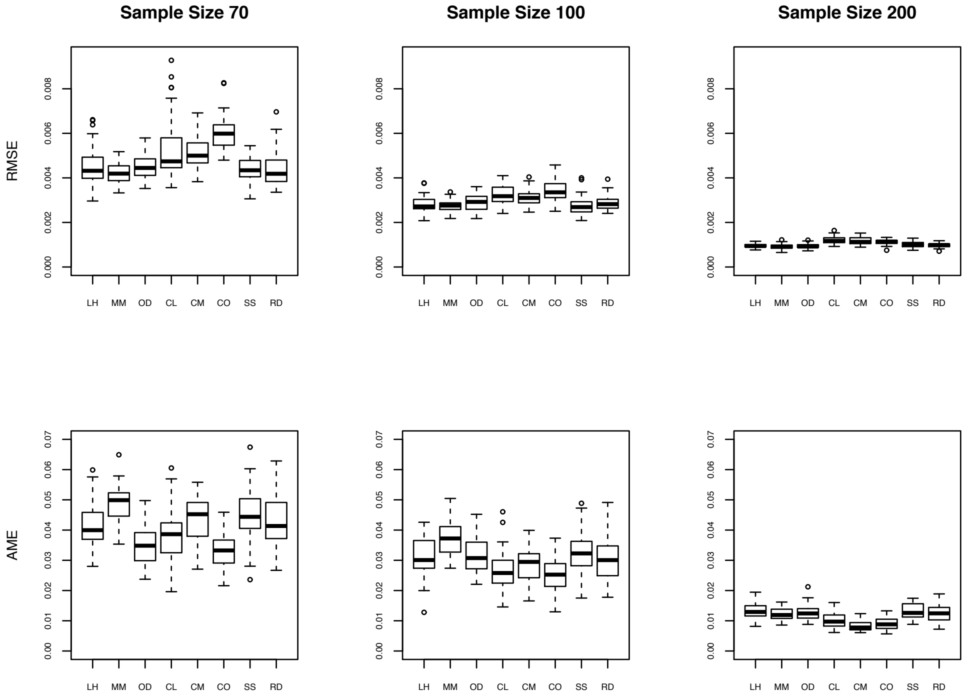


Figure 5.2: Comparing the RMSE and AME generated by the Corner Peak test function over three different sample sizes using a constant of $c_0 = 0.25$.

Continuous function was as high as 0.15 going down to around 0.05. The variability in the RMSE decreases dramatically as we increase the sample size. We see the same trend that we saw in the Continuous function for the cos-transformed designs, high values with large variability. The design that performs the best on the Corner Peak function with a constant of 0.25 in terms of RMSE across all sample sizes is the MaxiMin design. The MaxiMin design yields consistently low RMSEs with small variability. Now consider the AME of the same function, there is not the same dramatic decrease in variability as we saw in the RMSE, however, the variability is very small for all designs at a sample size of 200.

Based on the AME we would prefer the cos-transformed Orthogonal design, as it yields overall lower values of AME. However, the variability of the cos-transformed Orthogonal design is not smaller than any of the other

Table 5.4: Mean ratio of AME/MAD for each design of the Corner Peak function with a constant of $c_0 = 0.25$.

	Sample Size 70	Sample Size 100	Sample Size 200
LHS	10.74	8.06	3.41
MaxiMin	12.73	9.63	3.17
Orthogonal	9.10	8.23	3.27
Cos-LHS	9.96	6.91	2.59
Cos-MaxiMin	11.49	7.51	2.16
Cos-Orthogonal	8.57	6.61	2.36
Sobol	11.80	8.55	3.43
Random	11.08	7.86	3.23

Table 5.5: Mean ratio of RMSE/DEV for each design of the Corner Peak function with a constant of $c_0 = 1.85$.

	Sample Size 70	Sample Size 100	Sample Size 200
LHS	77.19	67.80	47.00
MaxiMin	63.50	58.74	44.35
Orthogonal	103.61	69.25	47.63
Cos-LHS	124.17	94.18	84.17
Cos-MaxiMin	78.95	69.49	69.63
Cos-Orthogonal	267.19	107.90	83.04
Sobol	73.52	64.36	50.92
Random	73.39	71.71	51.16

designs. In fact, the variability of each design within each sample size is approximately the same. The AME of the Corner Peak function with a constant of 0.25 shows the same trend in the cos-transformed designs as we saw in the cos-transformed designs of the Continuous function.

Now consider Table 5.3 to investigate the ratio of RMSE/Deviation. Again, designs that yield ratios less than 10 are considered good designs, and ratios less than 5 are preferred. The major difference between the Corner Peak function with a constant of 0.25 and the Continuous function is that even the ratios at a sample size of 70 are considered good, while we had to look at sample sizes of 200 to reach similar ratios in the Continuous case. Again, based on the RMSE/Deviation ratio for the corner peak function with a constant of 0.25 we would select the MaxiMin design as the design

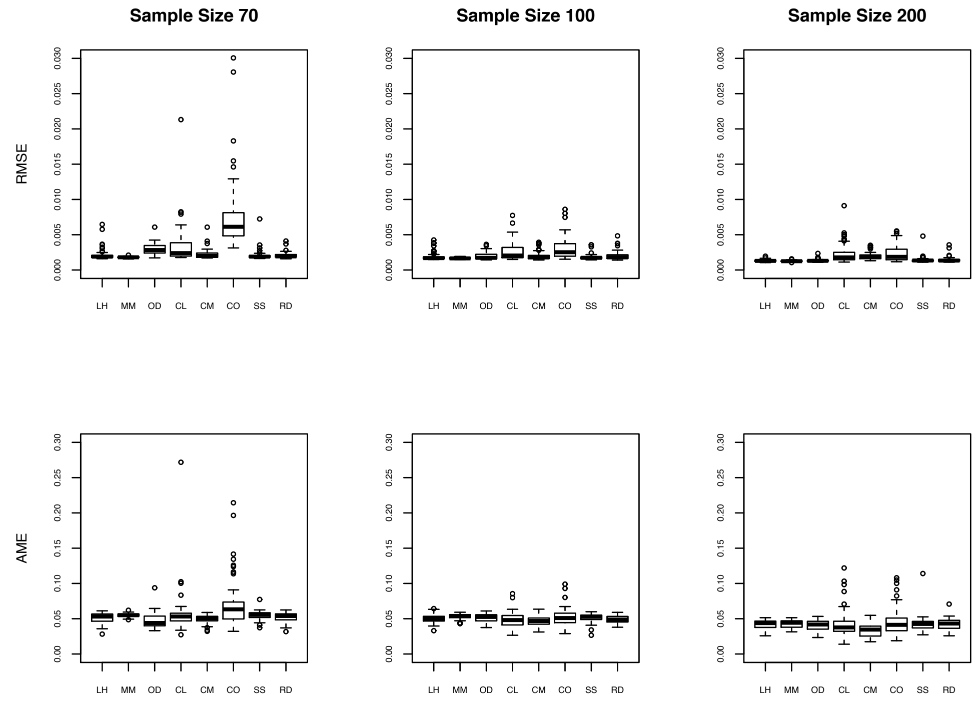


Figure 5.3: Comparing the RMSE and AME generated by the Corner Peak test function over three different sample sizes using a constant of $c_0 = 1.85$.

that performs the best over all sample sizes.

Table 5.4 shows the ratios of AME/MAD for the Corner Peak function with a constant of 0.25. The ratios for all designs start off better at a sample size of 70 than they were for a sample size of 200 for the Continuous function. The ratios steadily decrease as we increase the sample size, however, there is not one design that consistently performs better than the others in the AME/MAD case.

Now we will consider the effect of changing the constant of the Corner Peak function from 0.25 to 1.85. Figure 5.3 shows an obvious difference in the fit when compared to Figure 5.2. The scales for both the RMSE and the AME for the Corner Peak function with a constant of 1.85 are nearly three times as large as they were for the same function with a constant of 0.25. We do not see the same cascading trend in the magnitude of neither

Table 5.6: Mean ratio of AME/MAD for each design of the Corner Peak function with a constant of $c_0 = 1.85$.

	Sample Size 70	Sample Size 100	Sample Size 200
LHS	76.69	74.46	62.68
MaxiMin	82.15	79.61	63.57
Orthogonal	70.09	76.58	60.92
Cos-LHS	86.40	72.24	65.14
Cos-MaxiMin	74.18	68.99	50.28
Cos-Orthogonal	109.86	78.52	68.29
Sobol	82.12	76.74	63.77
Random	77.19	73.52	63.78

the RMSE nor AME that we saw in the Continuous function and the Corner Peak function with a constant of 0.25. Although the variability in the designs decreases as we increase the sample size as it did in the previous two cases, there is still significant variability in all the designs, especially the cos-transformed designs at a sample size of 200. In practice, if we were to see this behaviour we would use Cross Validation to assess how poorly the function is represented.

It becomes rather obvious that using a constant of 1.85 for the Corner Peak function is not advisable once we look at Tables 5.5 and 5.6. Both ratios yield extremely high values, some are even larger than 100, indicating that the mean of the output would be a better predictor than the predictor itself. Even at a sample size of 200 most values are close to 50 if not higher. Of course, in a real-world situation we do not have a choice in the constant, so for our purposes we will extend our recommendation from the constant of 0.25 to all Corner Peak functions, and recommend either the Cos-Transformed Orthogonal design, or the MaxiMin design.

5.3 Gaussian Function

The third function we consider is the Gaussian function.

$$f_3(x) = \exp\left(-\sum_{i=1}^{d} c_i^2 (x_i - w_i)^2\right)$$

In our simulations we use $c_0 = 7$ for the Gaussian function, and we let $w_i = 0$.

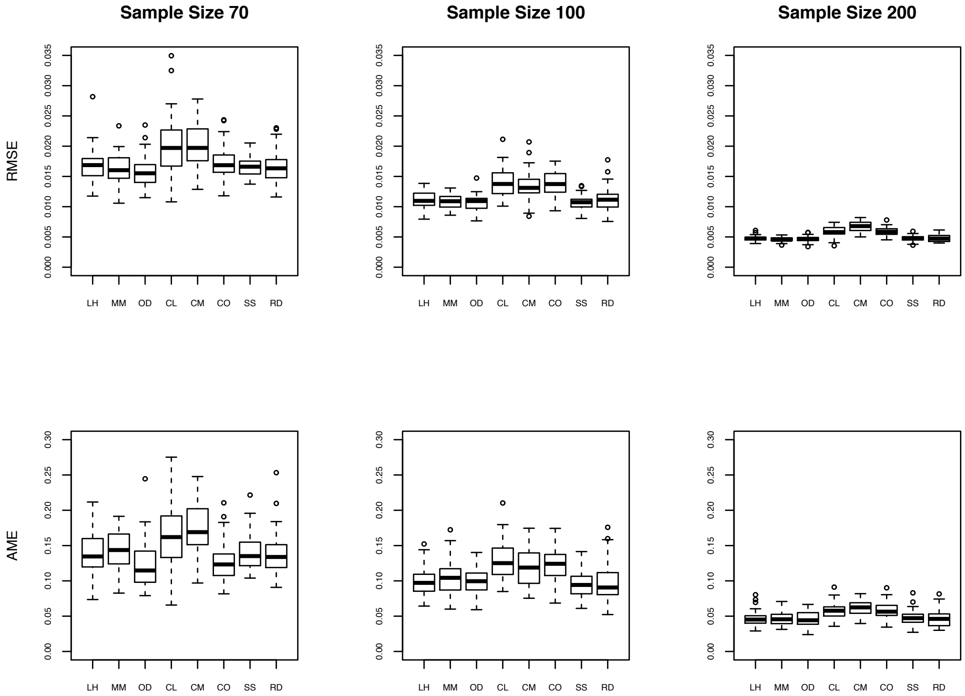


Figure 5.4: Comparing the RMSE and AME generated by the Gaussian test function over three different sample sizes using a constant of $c_0 = 7$.

In Figure 5.4 we see the familiar cascading values across the sample sizes. The difference is in the scales where the RMSE for the Gaussian function ranges from values of 0.035 in the sample size of 70, down to values of 0.005 in the sample size of 200. So it does not perform as well as the Corner Peak function with a constant of 0.25. The scales of the RMSE for the Gaussian function are approximately the same as those for the Corner Peak function with a constant of 1.85. The difference we see in the Gaussian function as opposed to the Corner Peak function with a constant of 1.85 is the defined decrease in values and variability of the RMSE as the sample size increases. Based on the RMSE plots in Figure 5.4 the design that appears to yield consistently good fits is the Sobol sequence along with the MaxiMin design.

Further investigating the Gaussian function we turn our attention to the AME. There is a notable decrease in variability as the sample size increases,

Table 5.7: Mean ratio of RMSE/DEV for each design of the Gaussian function.

	Sample Size 70	Sample Size 100	Sample Size 200
LHS	13.77	9.22	3.91
MaxiMin	13.34	8.93	3.77
Orthogonal	13.10	8.81	3.85
Cos-LHS	16.59	11.63	4.86
Cos-MaxiMin	16.61	11.04	5.55
Cos-Orthogonal	14.34	11.44	4.85
Sobol	13.68	8.78	3.92
Random	13.68	9.28	3.94

Table 5.8: Mean ratio of AME/MAD for each design of the Gaussian function.

	Sample Size 70	Sample Size 100	Sample Size 200
LHS	20.98	14.97	6.98
MaxiMin	21.55	15.61	6.99
Orthogonal	18.12	14.72	6.90
Cos-LHS	24.51	19.38	8.71
Cos-MaxiMin	25.81	17.69	9.16
Cos-Orthogonal	19.14	18.37	8.69
Sobol	20.97	14.24	7.05
Random	20.53	14.68	7.06

as well as a decrease in AME as the sample size increases. One important difference to note is that the behaviour of the AME in the cos-transformed designs mirrors the behaviour of the RMSE in the cos-transformed designs. There are a number of outliers in the AME across all sample sizes, even at a sample size of 200. This suggests that either we should run the code again with a larger sample size, or explore ways of improving our emulator.

Table 5.7 shows the ratios of the RMSE/Deviation for the Gaussian function. None of the ratios for a sample size of 70 are low enough to be considered good; it is not until we reach sample sizes of 100 that the ratios become reasonable. Once we reach a sample size of 200 we have good ratios for all of the designs. There is not one design that yields ratios that are better than the others at all sample sizes, however, in moving from a sample size of 100 to 200 we prefer the MaxiMin design.

Now consider Table 5.8 for the ratios of AME/MAD. As in Table 5.7 the ratios are not good at a sample size of 70. Once we reach a sample size of 100 we have ratios that are on the verge of acceptable, and finally at a sample size of 200 we have desirable ratios. Based on the ratio of AME/MAD at a sample size of 200 we would prefer the Orthogonal design, but the difference between the Orthogonal design and the LHS or MaxiMin design is minimal.

5.4 Oscillatory Function

The fourth function we consider is the Oscillatory function.

$$f_4(x) = cos\left(2\pi + \sum_{i=1}^{d} c_i x_i\right)$$

In our simulations we use $c_0 = 4.5$ for the Oscillatory function.

Table 5.9: Mean ratio of RMSE/DEV for each design of the Oscillatory function.

	Sample Size 70	Sample Size 100	Sample Size 200
LHS	9.84	6.39	2.36
MaxiMin	9.61	6.04	2.19
Orthogonal	10.14	6.76	2.23
Cos-LHS	11.29	7.12	2.58
Cos-MaxiMin	10.17	6.62	2.51
Cos-Orthogonal	12.24	7.36	2.42
Sobol	9.35	6.32	2.35
Random	9.94	6.77	2.28

In the RMSE plots of Figure 5.5 the same cascading effect is seen across the sample sizes that we have become familiar with in most of the other functions. There is noticeably more variability in the cos-transformed designs in the lower two sample sizes, but at the sample size of 200 all designs have essentially the same variability around approximately the same RMSE. At a sample size of 70, the MaxiMin design, the Orthogonal design, and the Sobol Sequence are approximately equal in terms of variability and median value of RMSE. To select a preferred design we move to the sample size of 100. At a sample size of 100 the obvious choice is the MaxiMin design. Not

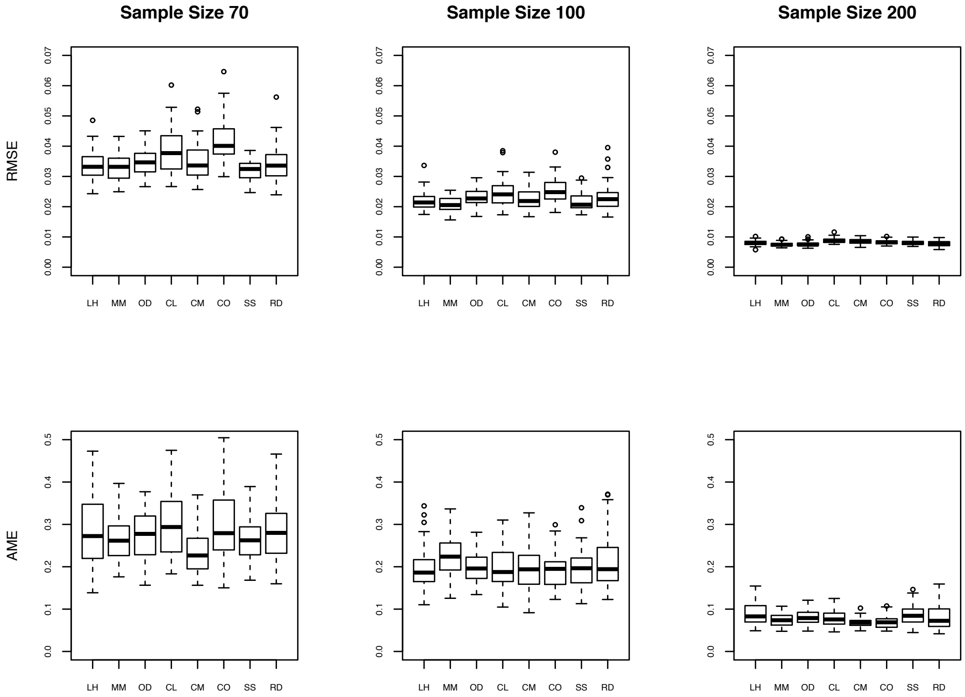


Figure 5.5: Comparing the RMSE and AME generated by the Oscillatory test function over three different sample sizes using a constant of $c_0 = 4.5$.

only does it have the smallest median value of RMSE, but also the smallest variability around it. This remains true at a sample size of 200.

In the AME plot of Figure 5.5 the cos-transformed MaxiMin design is decidedly smaller than the other designs at a sample size of 70. Upon closer inspection however, the range of the cos-transformed MaxiMin reaches almost as high as the maximum values of the MaxiMin design or the Orthogonal design or even the Sobol Sequence. Proceeding to a sample size of 100 there is no clear choice for the best design, but the cos-transformed MaxiMin has the most variability out of all the designs, which suggests that it is not a robust design. The Sobol Sequence is too variable to be considered a good design. At a sample size of 100 the Orthogonal design yields low values of RMSE with small variability. Yet, at a sample size of 200 the Orthogonal design does not perform well in comparison with the MaxiMin design. The

Table 5.10: Mean ratio of AME/MAD for each design of the Oscillatory function.

	Sample Size 70	Sample Size 100	Sample Size 200
LHS	20.68	14.70	6.63
MaxiMin	19.66	16.91	5.53
Orthogonal	20.47	14.66	5.98
Cos-LHS	21.97	14.67	5.82
Cos-MaxiMin	17.38	14.40	5.10
Cos-Orthogonal	21.69	14.21	5.24
Sobol	19.73	14.65	6.39
Random	20.90	15.87	5.98

MaxiMin design is a good choice of design for all the sample sizes we have investigated, so we feel comfortable recommending it as a robust design for the Oscillatory function.

In Table 5.9 the ratios of RMSE/Deviation are satisfactory for all designs at a sample size of 70. The smallest ratio at a sample size of 70 is from the Sobol Sequence, followed closely by the MaxiMin design. At a sample size of 100 the MaxiMin design is preferred followed by the Sobol Sequence. Finally, at a sample size of 200 the MaxiMin is the preferred design, followed by the Orthogonal design. At a sample size of 200 the Sobol Sequence does not perform well in comparison to the other designs.

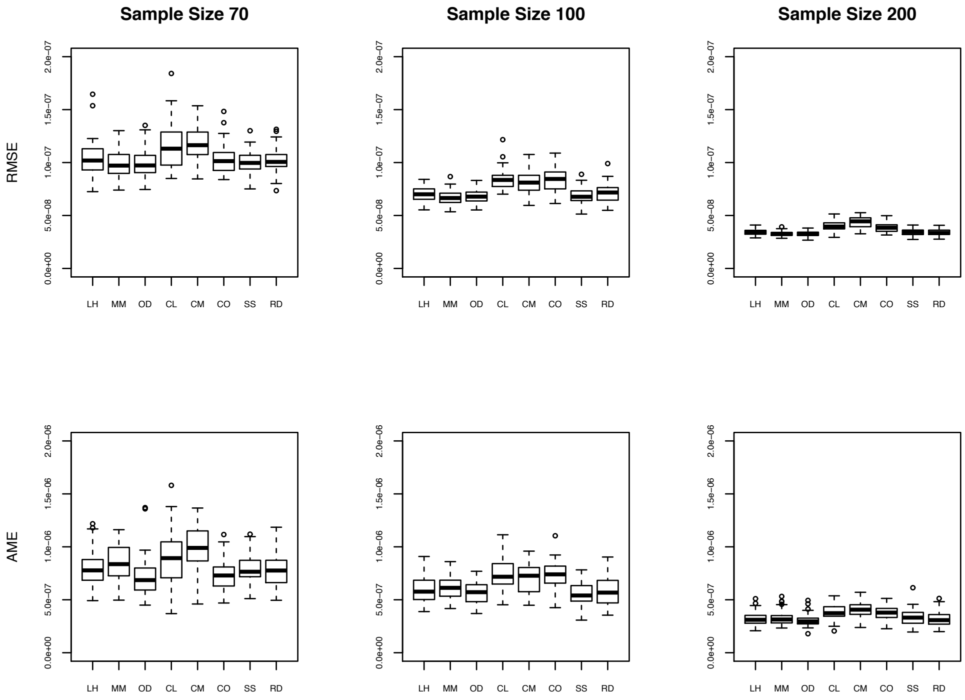
In Table 5.10 none of the designs yield good ratios of AME/MAD at a sample size of 70, and no design performs as well as we would expect at a sample size of 100. It is not until we reach a sample size of 200 that we see acceptable ratios of AME/MAD. At a sample size of 200 based purely on the ratios of AME/MAD we would select the cos-transformed MaxiMin design. However, since the ratio of the MaxiMin design is not drastically different from its cos-transformed counterpart, we prefer the MaxiMin design as it consistently performs well.

5.5 Product Peak Function

The fifth and final function we consider is the Product Peak function.

$$f_5(x) = \prod_{i=1}^{d} \left(c_i^{-2} + (x_i - w_i)^2\right)^{-1}$$

In our simulations we use $c_0 = 7.25$ for the Product Peak function, and we let $w_i = 0$.



Sample Size 70

Sample Size 100

Sample Size 200

Figure 5.6: Comparing the RMSE and AME generated by the Product Peak test function over three different sample sizes using a constant of $c_0 = 7.25$.

At a glance, Figure 5.6 makes the Product Peak function appear to be well-behaved. Upon closer investigation however, we notice that the scale of RMSE goes from around 2×10^{-7} at a sample size of 70, and then decreases further to around 5×10^{-8} at a sample size of 200. The scale of the AME is between 1.6×10^{-6} at a sample size of 70, down to 5×10^{-7} at a sample size of 200. To better understand why the errors are so small, we take a closer look at the Product Peak function. The Product Peak takes the product of inverses; this means that the output of the Product Peak function is expected to be close to zero. So it is not surprising that the emulator does so well, all it is doing is essentially predicting zero. To be able to predict zero in practice is not very useful. Since the Product Peak function is not useful in practice, we will not include the tables of RMSE/DEV or AME/MAD, and we will not discuss the design choices. The Product Peak function has been excluded from the analysis in Chapter 6.

Chapter 6

Results for Random Functions

In Chapter 5 we specified the constant c_i using $k_i = \frac{i-0.5}{d}, i = 1, \ldots, 10$ with $d = 10$ and c_0 varied depending on the test function. In this chapter we will use the same values of c_0 as in Chapter 5, but we will now find 10 $k_i \sim UNIF(0, 1)$, which will in turn generate 10 different values of c_i. So where the previous test functions remained fixed and the design changed, here the actual function changes which gives a sense of design performance over many different functions and not merely design performance over variations of the same design.

For our discussion of the results in this chapter we have excluded the tables of RMSE/Deviation and AME/MAD. The reason for this is, as we found under the Product Peak function in Chapter 5, the tables can be misleading if we do not have a well behaved function to start with. So we will focus our discussion on the plots of RMSE and AME.

6.1 Continuous Function

In Figure 6.1 an interesting difference between the RMSE in this case and the one we observed in Chapter 5, is in the variability. The variability within each design is much smaller than it was before, but the variability between the different designs is much larger. At a sample size of 70 the Orthogonal design, Sobol Sequence and the Random Design perform equally well. At a sample size of 100 the MaxiMin design performs as well as the Orthogonal design, or the Sobol Sequence, while the Random Design does not perform as well as the others any more. At a sample size of 200 the MaxiMin design or Sobol Sequence design are preferred. The scale of the RMSE ranges from 0.012 at a sample size of 70 down to 0.002 which is a better range than what we saw in Chapter 5 (from 0.015 to 0.005).

The scale of the AME in Figure 6.1 remains the same from what we saw in Chapter 5, from 0.15 to 0.05. The design that performs best on the

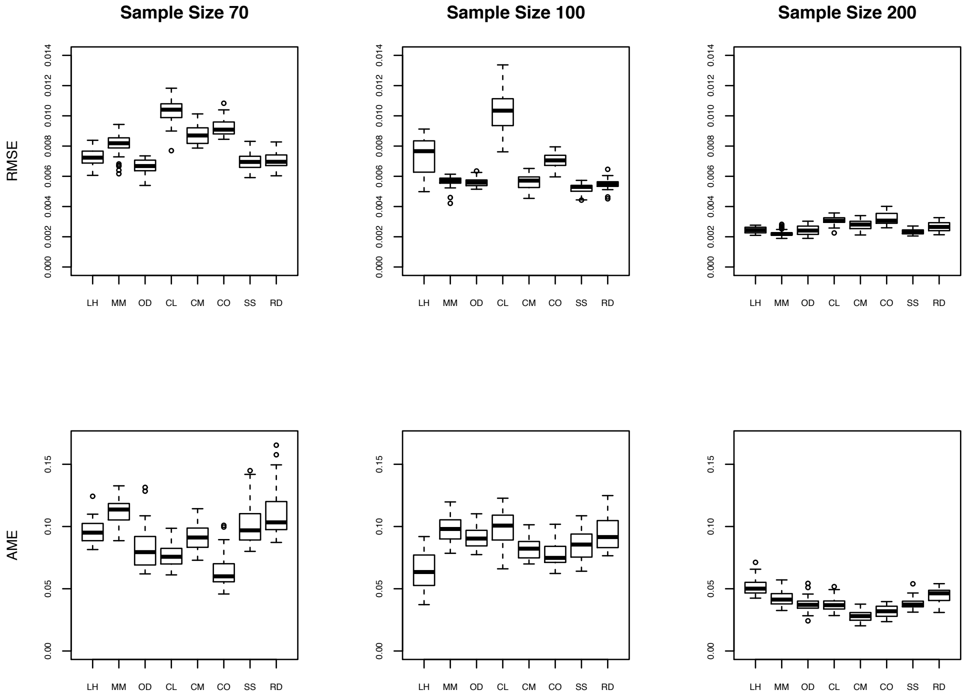
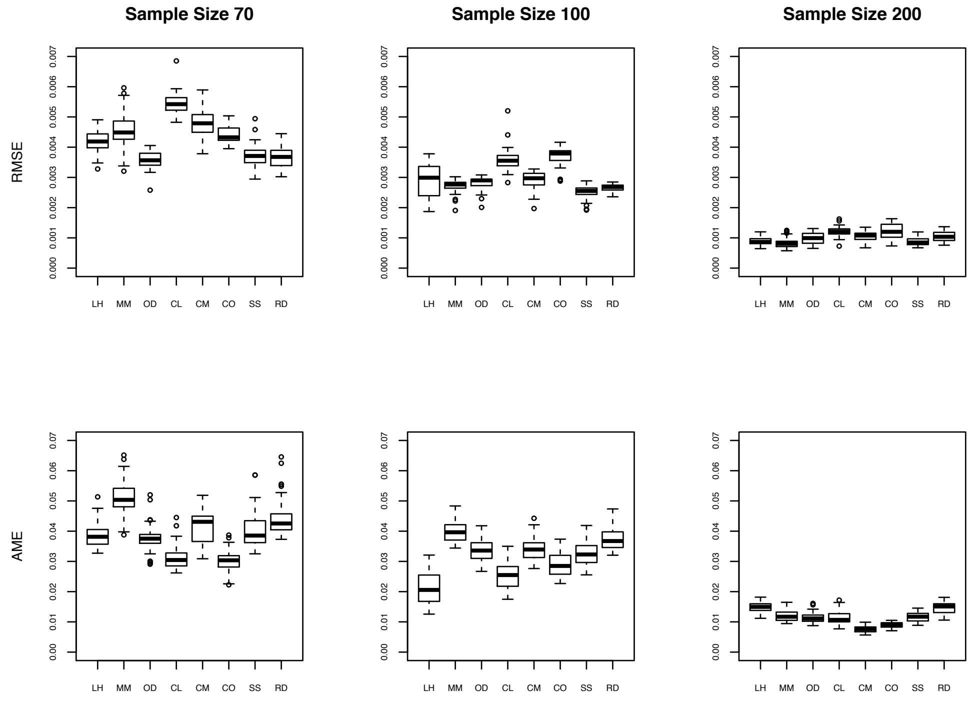


Figure 6.1: Comparing the RMSE and AME generated by the Continuous test function over three different sample sizes using a constant of $c_0 = 5$.

AME at a sample size of 70 is the cos-transformed LHS. The cos-transformed Orthogonal design has the smallest median AME out of all the designs at a sample size of 70, it is not preferred however because of the large variability and what appears to be a cluster of outliers in the upper bounds of the AME. The median AME for the MaxiMin design is the highest at a sample size of 70, but the variability within the design is among the smallest. Continuing to a sample size of 100 we see a drastic decrease in the AME of the LHS design. The variability of the LHS is too large to be considered robust, but the significant drop in AME is worth making note of. The MaxiMin design and Orthogonal design along with all three cos-transformed designs perform well with approximately equal variance at a sample size of 100. To decide on a preferred design we consider a sample size of 200. We are hesitant to select a cos-transformed design as the behaviour of the AME contradicts the behaviour of the RMSE. The Orthogonal design has too many outliers to be considered robust. Thus, we select the MaxiMin design as our design of choice overall for the Continuous function.

6.2 Corner Peak Function


Sample Size 70

Sample Size 100

Sample Size 200

Figure 6.2: Comparing the RMSE and AME generated by the Corner Peak test function over three different sample sizes using a constant of $c_0 = 0.25$.

The RMSE plots of the Corner Peak function with a constant of 0.25 in Figure 6.2 show that the scale has decreased from what we saw in Chapter 5. We are now going from around 0.006 (down from 0.008) at a sample size of 70, and then down to 0.001 (down from 0.002) at a sample size of 200. The cos-transformed designs do not perform well on the Corner Peak function with a constant of 0.25 at any sample size. Initially, we are drawn to the low values of RMSE of the Orthogonal design at a sample size of 70. However,

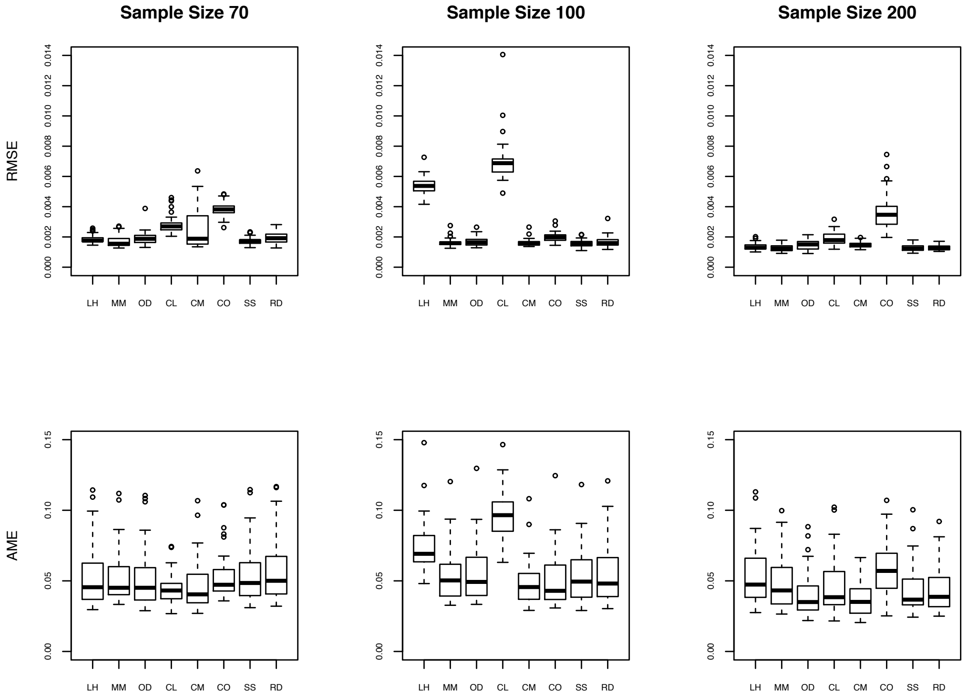


Figure 6.3: Comparing the RMSE and AME generated by the Corner Peak test function over three different sample sizes using a constant of $c_0 = 1.85$.

once we increase the sample size the Orthogonal design looks less promising. At a sample size of 200 the MaxiMin design and the Sobol design appear to be performing equally well. At a sample size of 100 the same two designs perform approximately as well again. As we move down to a sample size of 70 however, the Sobol Sequence actually performs better than the MaxiMin sampling.

The AME plots of the Corner Peak function with a constant of 0.25 in Figure 6.2 reveal the same trend we saw in the RMSEs of the same plot. The cos-transformed designs do not perform well consistently, so we will not consider them to be candidates for a robust overall design. Starting at a sample size of 70 then, we would prefer the Orthogonal design or the LHS design, in terms of both median values and variability. Continuing to a sample size of 100 we prefer the LHS design once more and the Orthogonal design or the Sobol Sequence are close contenders thereafter. Once we move to a sample size of 200, the LHS design does not perform well, and the Orthogonal design has a few too many outliers to be considered robust. The Sobol Sequence however performs well across all sample sizes. Therefore, in terms of RMSE and AME for the Corner Peak function at a constant of 0.25, we would prefer the Sobol Sequence as a robust design choice.

Figure 6.3 confirms what we concluded in Chapter 5, that using a constant of 1.85 for the Corner Peak function does not result in a well behaved function. Thus, selecting a design that performs well on the Corner Peak function with a constant of 1.85 is counterproductive.

6.3 Gaussian Function

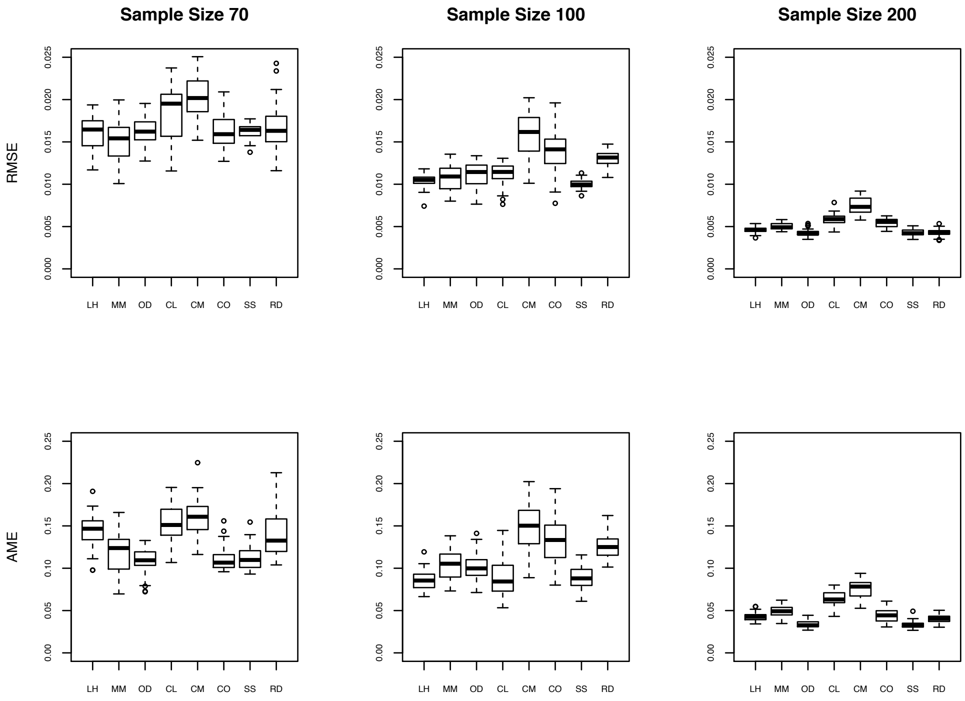


Figure 6.4: Comparing the RMSE and AME generated by the Gaussian test function over three different sample sizes using a constant of $c_0 = 7$.

The RMSE of the Gaussian function in Figure 6.4 is relatively consistent in terms of both variability and the median RMSEs for all designs in each sample size. The scale of the RMSE for the Gaussian function is between 0.025 for a sample size of 70, down to 0.005 for a sample size of 200. This is smaller than the scale (from 0.035 to 0.005) we saw in the Gaussian function of Chapter 5. In general the RMSEs decrease as we increase the sample size, as well as the variability decreasing for all designs. The cos-transformed designs have larger values of RMSEs than we desire for a robust design. At a sample size of 70 the MaxiMin design reaches the lowest values of RMSE, the variability of the MaxiMin design is large at the small sample size, but even at its maximum it does not extend past any other design that can be considered robust except for the Sobol Sequence. At a sample size of 100 the RMSEs of all designs excluding the cos-transformed designs are approximately equal. Again, at a sample size of 200, with the exception of the cos-transformed designs, the RMSE of all the designs is approximately equal. Since the goal of our analysis is to find a design that consistently performs well, we would select the MaxiMin function as it is preferred in the small sample case, and then is equal to the other designs at the larger sample sizes.

The AME of the Gaussian function in Figure 6.4 shows a similar trend as what we saw in the RMSE of the same Figure. The MaxiMin design attains the smallest values of AME at a sample size of 70, and we see more variability in the cos-transformed designs than in the other designs across all sample sizes. The scale of the AME starts at smaller values (0.20) for the sample size of 70 as opposed to 0.3 but at a sample size of 200 the designs do not reach RMSEs lower than 0.05, what we saw in the previous chapter. Although the Orthogonal design appears to be the best design at both a sample size of 70 and 200, it shows some outliers in the higher end at a sample size of 100. While the MaxiMin design attains values as low as the Orthogonal at a sample size of 70 and 100, and is among the best designs at a sample size of 200. This is an indication that the MaxiMin design performs well on all sample sizes we have considered, and we feel confident recommending the MaxiMin design as a robust design for the Gaussian function with a constant of 7.

6.4 Oscillatory Function

In Figure 6.5 we notice that there has been no change in the scales of neither RMSE nor AME of the Oscillatory function from the previous

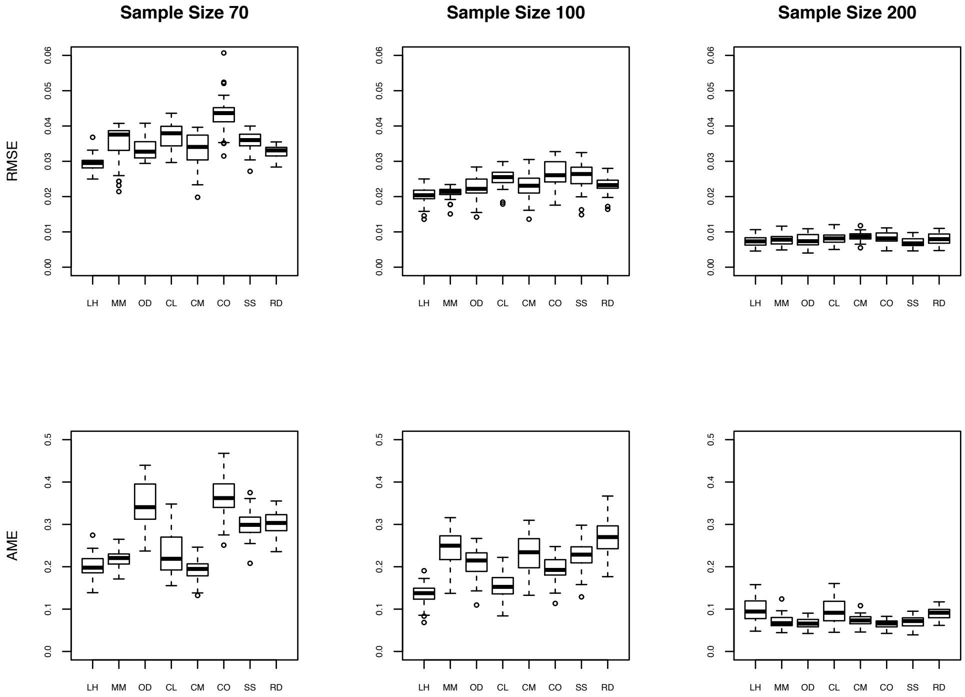


Figure 6.5: Comparing the RMSE and AME generated by the Oscillatory test function over three different sample sizes using a constant of $c_0 = 4.5$.

chapter. Once again, the Cos-Transformed designs do not perform well consistently in the RMSE. Although the MaxiMin design has outliers in the RMSE at a sample size of 70, they are towards the lower values, which we allow. In fact, we would prefer the MaxiMin design at a sample size of 70, because although the Orthogonal design has less variability, the values of the Orthogonal are in general larger than in the MaxiMin design. We continue on to a sample size of 100 and see that most designs have conformed around approximately the same values of RMSE, but the low variability of the MaxiMin design make it favourable at a sample size of 100. At a sample size of 200 the designs are essentially equivalent, so we make the MaxiMin the preferred design based on the RMSE on the Oscillatory function with a constant of 4.5.

The AME plots in Figure 6.5 show inconsistencies in the Cos-Transformed designs once more. At a sample size of 70, we would consider either the LHS design or the MaxiMin design as preferable designs both in terms of variability and RMSE values. At a sample size of 100 the LHS performs better than all the other designs, while the MaxiMin performs approximately as well as the other designs. At a sample size of 200 the LHS design is among the worst designs, which suggests the seemingly superior performance at a sample size of 100 is due to chance rather than good performance. So across all sample sizes we prefer the MaxiMin design as a robust design choice for the Oscillatory function with a constant of 4.5.

Chapter 7

Conclusion

Computer experiments have applications in most areas of research, and have broadened research opportunities by enabling the researcher to emulate a mathematical model of the system and then simulate future output. This allows for experiments that would otherwise be impossible to perform.

In Chapter 2 we introduced the idea of computer experiments and the applications of the Gaussian Process. In Chapter 3 we introduced various designs that one might consider for selecting the initial values of the simulations. In our analysis we considered eight of those designs, namely the LHS, MaxiMin and Orthogonal designs and their cos-transformed counterparts, as well as the Sobol Sequence and the Random Design.

The goal of our analysis was to present a comprehensive study of various standard design choices and make recommendations to a practitioner on good designs for computer experiments. We considered a wide range of scenarios, varying the sample size, varying the test function, as well as the effect of changing the constant for one of the test functions. In Chapters 5 and 6 we compared the effects of evaluating the output at fixed points, versus evaluating the output at random points. The differences in Chapters 5 and 6 were not notable.

One important finding, is that the Cos-Transformed designs yield good emulators for the simpler functions, but as the function increases in complexity, the Cos-Transformed designs do not perform as well as the other designs. The MaxiMin design however, is a robust design across all functions except in Chapter 6 for the Corner Peak function with a constant of 0.25 when evaluated at random points, where the Sobol Sequence is the preferred design choice. Although the Sobol Sequence is preferred in this case, the difference between the Sobol Sequence and the MaxiMin design is minimal. In terms of selecting a design choice that will yield consistently good results for any test function, the MaxiMin would still perform well on this type of function in practice. So we are confident in recommending the MaxiMin design as a robust design choice for the initial values of any of the functions we have considered.

In future work we would consider exploring the effects of different constants on the different functions. Our work was done in 10 dimensions, so it would be interesting to investigate the design choices if we consider other dimensions than 10.

Bibliography

Blackwell, D. (1951), "Comparison of experiment," in *Proceedings of the 2nd Berkeley Symposium*, Berkeley, Ca., University of California Press, pp. 93–102.

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*, 86, 953–963.

Dette, H. and Pepelyshev, A. (2010), "Generalized Latin Hypercube Design for Computer Experiments," *Technometrics*, 52, 421–429.

Fang, K.-T., Lin, D. K. J., Winker, P., and Zhong, Y. (2000), "Uniform Design: Theory and Application," *Technometrics*, 42, 237–248.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004), *Bayesian Data Analysis*, Boca Raton, Florida: Chapman & Hall / CRC, 2nd ed.

Handcock, M. S. and Stein, M. L. (1993), "A Bayesian Analysis of Kriging," *Technometrics*, 35, 403–410.

Higdon, D., Gattiker, J., Williams, B., and Rightley, M. (2008), "Computer Model Calibration Using High-Dimensional Output," *Journal of the American Statistical Association*, 103, 570–583.

Iman, R. L. and Conover, W. J. (1982), "A Distribution-Free Approach to Inducing Rank Correlation Among Input Variables," *Communications in Statistics. Simulation and Computation.*, 11, 311–334.

Johnson, M. E., Moore, L. M., and Ylvisaker, D. (1990), "Minimax and Maximin Distance Designs," *Journal of Statistical Planning and Inference*, 26, 131–148.

Lim, Y. B., Sacks, J., Studden, W. J., and Welch, W. J. (2002), "Design and Analysis of Computer Experiments When the Output is Highly Correlated Over the Input Space," *Canadian Journal of Statistics*, 30, 109–126.

Lindley, D. V. (1956), "On a measure of information provided by an experiment," *Annals of Mathematical Statistics*, 27, 986–1005.

Linkletter, C., Bingham, D., Hengartner, N., Higdon, D., and Ye, K. Q. (2006), "Variable Selection for Gaussian Process Models in Computer Experiments," *Technometrics*, 48, 478–490.

Loeppky, J. L., Sacks, J., and Welch, W. J. (2009), "Choosing the Sample Size of a Computer Experiment: A Practical Guide," *Technometrics*, 51, 366–376.

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 Ouput from a Computer Code," *Technometrics*, 21, 239–245.

Morris, M. D. and Mitchell, T. J. (1995), "Exploratory Designs for Computational Experiments," *Journal of Statistical Planning and Inference*, 43, 381–402.

O'Hagan, A. (1992), "Some Bayesian Numerical Analysis," in *Bayesian Statistics 4*, eds. Bernardo, J. M., Berger, J. O., Dawid, A. P., and Smith, A. F. M., Oxford University Press, pp. 345–363.

— (1994), *Kendall's Advanced Theory of Statistics: Volume 2B, Bayesian Analysis*, London: Edward Arnold.

Owen, A. B. (1992), "Orthogonal Arrays for Computer Experiments, Integration and Visualization," *Statistica Sinica*, 2, 439–452.

Sacks, J., Schiller, S. B., and Welch, W. J. (1989a), "Designs for Computer Experiments," *Technometrics*, 31, 41–47.

Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989b), "Designs and Analysis of Computer Experiments (with Discussion)," *Statistical Science*, 4, 409–435.

Shewry, M. C. and Wynn, H. P. (1987), "Maximum Entropy Sampling," *Journal of Applied Statistics*, 14, 165–170.

Sobol, I. M. (1967), "Distribution of points in a cube and approximate evaluation of integrals," *USSR Computational Mathematics and Mathematical Physics*, 7, 86–112.

Steinberg, D. M. and Bursztyn, D. (2004), "Data Analytic Tools for Understanding Random Field Regression Models," *Technometrics*, 46, 411–420.

Tang, B. (1993), “Orthogonal Array-based Latin Hypercubes,” *Journal of the American Statistical Association*, 88, 1392–1397.

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*, 34, 15–25.