Computer Experiments

J. R. Koehler[1] and A.B. Owen[2]

[1]Department of Mathematics, University of Colorado, Denver
[2]Department of Statistics, Stanford University

1 Introduction

Deterministic computer simulations of physical phenomena are becoming widely used in science and engineering. Computers are used to describe the flow of air over an airplane wing, combustion of gasses in a flame, behavior of a metal structure under stress, safety of a nuclear reactor, and so on.

Some of the most widely used computer models, and the ones that lead us to work in this area, arise in the design of the semiconductors used in the computers themselves. A process simulator starts with a data structure representing an unprocessed piece of silicon and simulates the steps such as oxidation, etching and ion injection that produce a semiconductor device such as a transistor. A device simulator takes a description of such a device and simulates the flow of current through it under varying conditions to determine properties of the device such as its switching speed and the critical voltage at which it switches. A circuit simulator takes a list of devices and the way that they are arranged together and computes properties of the circuit as a whole.

In each of these computer simulations, the user must specify the values of some governing variables. For example in process simulation the user might have to specify the duration and temperature of the oxidation step, and doses and energies for each of the ion implantation steps. These are continuously valued variables. There may also be discrete variables, such as whether to use wet or dry oxidation. Most of this chapter treats the case of continuous variables, but some of it is easily adaptable to discrete variables, especially those taking only two values.

Let $X \in R^p$ denote the vector of input values chosen for the computer program. We will write X as the row vector $(X^1, \ldots, X^p)$ using superscripts to denote components of X. We assume that each component X^j is continuously adjustable between a lower and an upper limit, which after a linear transformation can be taken to be 0 and 1 respectively. (For some results where every input is dichotomous see [42].) The computer program is denoted by f and it computes q output quantities, denoted by $Y \in R^q$.

$$Y = f(X), \quad X \in [0,1]^p. \tag{1}$$

Some important quantities describing a computer model are the number of inputs p, the number of outputs q and the speed with which f can be computed. These vary enormously in applications. In the semiconductor problems we have considered p is usually between 4 and 10. Other computer experiments use scores or even hundreds of input variables. In our motivating applications q is usually larger than 1. For example interest might center on the switching speed of a device and also on its stability as measured by a breakdown voltage. For some problems f takes hours to evaluate on a supercomputer and for others f runs in milliseconds on a personal computer.

Equation (1) differs from the usual X-Y relationship studied by statisticians in that there is no random error term. If the program is run twice with the same X, the same Y is obtained both times. Therefore it is worth discussing why a statistical approach is called for.

These computer programs are written to calculate Y from a known value of X. The way they are often used however, is to search for good values of X according to some goals for Y. Suppose that $X_1 = (X_1^1, \ldots, X_1^p)$ is the initial choice for X. Often X_1 does not give a desirable $Y_1 = f(X_1)$.

The engineer or scientist can often deduce why this is, from the program output, and select a new value, X_2 for which $Y_2 = f(X_2)$ is likely to be an improvement. This improvement process can be repeated until a satisfactory design is found. The disadvantage of this procedure is that it may easily miss some good designs X, because it does not fully explore the design space. It can also be slow, especially when p is large, or when improvements of Y^1 say, tend to appear with worsenings of Y^2 and vice versa.

A commonly used way of exploring the design space around X_1 is to vary each of the X_1^j one at a time. As is well known to statisticians, this approach can be misleading if there are strong interactions among the components of X. Increasing X^1 may be an improvement and increasing X^2 may be an improvement, but increasing them both together might make things worse. This would usually be determined from a confirmation run in which both X^1 and X^2 have been increased. The greater difficulty with interactions stems from missed opportunities: the best combination might be to increase X^1 while decreasing X^2, but one at a time experimentation might never lead the user to try this. Thus techniques from experimental design may be expected to help in exploring the input space.

This chapter presents and compares two statistical approaches to computer experiments. Randomness is required in order to generate probability or confidence intervals. The first approach introduces randomness by modeling the function f as a realization of a Gaussian process. The second approach does so by taking random input points (with some balance properties).

2 Goals in computer experiments

There are many different but related goals that arise in computer experiments. The problem described in the previous section is that of finding a good value for X according to some criterion on Y. Here are some other goals in computer experimentation: finding a simple approximation $\hat{f}$ that is accurate enough over a region A of X values, estimating the size of the error $\hat{f}(X_0) - f(X_0)$ for some $X_0 \in A$, estimating $\int_A f dX$, sensitivity analysis of Y with respect to changes in X, finding which X^j are most important for each response Y^k, finding which competing goals for Y conflict the most, visualizing the function f and uncovering bugs in the implementation of f.

2.1 Optimization

Many engineering design problems take the form of optimizing Y^1 over allowable values of X. The problem may be to find the fastest chip, or the least expensive soda can. There is often, perhaps usually, some additional constraint on another response Y^2. The chip should be stable enough, and the can should be able to withstand a specified internal pressure.

Standard optimization methods, such as quasi-Newton or conjugate gradients (see for example Gill, Murray and Wright [22]) can be unsatisfactory for computer experiments. These methods usually require first and possibly second derivatives of f, and these may be difficult to obtain or expensive to run. The standard methods also depend strongly on having good starting values. Computer experimentation as described below is useful in the early stages of optimization where one is searching for a suitable starting value. It is also useful when searching for several widely separated regions of the predictor space that might all have good Y values. Given a good starting value, the standard methods will be superior if one needs to locate the optimum precisely.

2.2 Visualization

As Diaconis [16] points out, being able to compute a function f at any given value X does not necessarily imply that one "understands" the function. One might not know whether the function is continuous or bounded or unimodal, where its optimum is or whether it has asymptotes.

Computer experimentation can serve as a primitive way to visualize functions. One evaluates f at a well chosen set of points $x_1, \ldots, x_n$ obtaining responses $y_1, \ldots, y_n$. Then data visualization methods may be applied to the $p+q$ dimensional points $(x_i, y_i), i = 1, \ldots, n$. Plotting the responses versus the input variables (there are pq such plots) identifies strong dependencies, and plotting residuals from a fit can show weaker dependencies. Selecting the points with desirable values of Y and then producing histograms and plots of the corresponding X values can be used to identify the most promising subregion of X values. Sharifzadeh, Koehler, Owen and Shott [71] took this approach to find that increasing a certain implant dose helped to make two different threshold voltages near their common targets *and* nearly equal (as they should have been). Similar exploration can identify which input combinations are likely to crash the simulator.

Roosen [66] has used computer experiment designs for the purpose of visualizing functions fit to data.

2.3 Approximation

The original program f may be exceedingly expensive to evaluate. It may however be possible to approximate f by some very simple function $\hat{f}$, the approximation holding adequately in a region of interest, though not necessarily over the whole domain of f. If the function $\hat{f}$ is fast to evaluate, as for instance a polynomial, neural network or a MARS model (see Friedman [20]), then it may be feasible to make millions of $\hat{f}$ evaluations. This makes possible brute force approximations for the other problems. For example, optimization could be approached by finding the best value of $\hat{f}(x)$ over a million random runs x.

Approximation by computer experiments involves choosing where to gather $(x_i, f(x_i))$ pairs, how to construct an approximation based on them and how to assess the accuracy of this approximation.

2.4 Integration

Suppose that X^* is the target value of the input vector, but in the system being modeled the actual value of X will be random with a distribution dF that hopefully is concentrated near X^*. Then one is naturally interested in $\int f(X)dF$, the average value of Y over this distribution. Similarly the variance of Y and the probability that Y exceeds some threshold can be expressed in terms of integrals. This sort of calculation is of interest to researchers studying nuclear safety. McKay [37] surveys this literature.

Integration and optimization goals can appear together in the same problem. In robust design problems [60], one might seek the value X_0 that minimizes the variance of Y as X varies randomly in a neighborhood of X_0.

3 Approaches to computer experiments

There are two main statistical approaches to computer experiments, one based on Bayesian statistics and a frequentist one based on sampling techniques. It seems to be essential to introduce randomness in one or other of these ways, especially for the problem of gauging how much an estimate $\hat{f}(X_0)$ might differ from the true value $f(X_0)$.

In the Bayesian framework, surveyed below in Sections 4 and 5, f is a realization of a random process. One sets a prior distribution on the space of all functions from $[0,1]^p$ to R^q. Given the values $y_i = f(x_i), i = 1, \ldots, n$ one forms a posterior distribution on f or at least on certain aspects of it such as $f(x_0)$. This approach is extremely elegant. The prior distribution is usually taken to be Gaussian so that any finite list of function values has a multivariate normal distribution. Then the posterior distribution, given observed function values is also multivariate normal. The posterior mean interpolates the observed values and the posterior variance may be used to give 95% posterior probability intervals. The method extends naturally to incorporate measurement and prediction of derivatives, partial derivatives and definite integrals of f.

The Bayesian framework is well developed as evidenced by all the work cited below in Sections 4 and 5. But, as is common with Bayesian methods there may be difficulty in finding an appropriate prior distribution. The simulator output might not have as many derivatives as the underlying physical reality, and assuming too much smoothness for the function can lead to Gibbs-effect overshoots. A numerical difficulty also arises: the Bayesian approach requires solving n linear equations in n unknowns when there are n data points. The effort involved grows as n^3 while the effort in computing $f(X_1), \ldots, f(X_n)$ grows proportionally to n. Inevitably this limits the size of problems that can be addressed. For example, suppose that one spends an hour computing $f(x_1), \ldots, f(x_n)$ and then one minute solving the linear equations. If one then finds it necessary to run 24 times as many function evaluations, the time to compute the $f(x_i)$ grows from an hour to a day, while the time to solve the linear equations grows from one minute to over nine and a half days.

These difficulties with the Bayesian approach motivate a search for an alternative. The frequentist approach, surveyed in Sections 6 and 7, introduces randomness by taking function values $x_1, \ldots, x_n$ that are partially determined by pseudo-random number generators. Then this randomness in the x_i is propagated through to randomness in $\hat{f}(x_0)$. This approach allows one to consider f to be deterministic, and in particular to avoid having to specify a distribution for f. The material given there expands on a proposal of Owen [49]. There is still much more to be done.

4 Bayesian prediction and inference

A Bayesian approach to modeling simulator output [68, 69, 87] can be based on a spatial model adapted from the geo-statistical Kriging model [36, 29, 12, 62]. This approach treats the bias, or systematic departure of the response surface from a linear model, as the realization of a stationary random function. This model has exact predictions at the observed responses and predicts with increasing error variance as the prediction point move away from all the design points.

This section introduces the Kriging (or Bayesian) approach to modeling the response surfaces of computer experiments. Several correlation families are discussed as well as their effect on prediction and error analysis. Additionally, extensions to this model are presented that allow the use and the modeling of gradient information.

4.1 The Kriging model

The Kriging approach uses a two component model. The first component consists of a general linear model while the second (or lack of fit) component is treated as the realization of a stationary Gaussian random function. Define $S = [0,1]^p$ to be the design space and let $x \in S$ be a scaled

p-dimensional vector of input values. The Kriging approach models the associated response as

$$Y(x) = \sum_{j=1}^{k} \beta_j h_j(x) + Z(x) \tag{2}$$

where the h_j's are known fixed functions, the β_j's are unknown coefficients to be estimated and $Z(x)$ is a stationary Gaussian random function with $\mathrm{E}[Z(x)] = 0$ and covariance

$$\mathrm{Cov}[Z(x_i), Z(x_j)] = \sigma^2 R(x_j - x_i). \tag{3}$$

For any point $x \in S$, the simulator output $Y(x)$ at that point has the Gaussian distribution with mean $\sum \beta_j h_j(x)$ and variance σ^2. The linear component models the drift in the response, while the systematic lack-of-fit (or bias) is modeled by the second component. The smoothness and other properties of $Z(\cdot)$ are controlled by $R(\cdot)$.

Let design $D = \{x_i, i = 1, \ldots, n\} \subset S$ yield responses $y'_D = \{y(x_1), \ldots, y(x_n)\}$ and consider a linear predictor

$$\hat{y}(x_0) = \lambda'(x_0) y_D$$

of an unobserved point x_0. The Kriging approach of Matheron [36] treats $\hat{y}(x_0)$ as a random variable by substituting Y_D for y_D where

$$Y'_D = (Y(x_1), \ldots, Y(x_n)).$$

The best linear unbiased predictor (BLUP) finds the $\lambda(x_0)$ that minimizes

$$\mathrm{MSE}[\hat{Y}(x_0)] = \mathrm{E}[\lambda' Y_D - Y(x_0)]^2$$

subject to the unbiasedness condition

$$\mathrm{E}[\lambda' Y_D] = \mathrm{E}[Y(x_0)].$$

The BLUP of $Y(x_0)$ is given by

$$\hat{Y}(x_0) = h'(x_0)\hat{\beta} + v'_{x_0} V_D^{-1}(Y_D - H_D\hat{\beta}) \tag{4}$$

where

$$\begin{aligned} h'(x_0) &= (h_1(x_0), \ldots, h_k(x_0)) \\ (H_D)_{ij} &= h_j(x_i) \\ (V_D)_{ij} &= \mathrm{Cov}[Z(x_i), Z(x_j)] \\ v'_{x_0} &= (\mathrm{Cov}[Z(x_0), Z(x_1)], \ldots, \mathrm{Cov}[Z(x_0), Z(x_n)]) \end{aligned}$$

and

$$\hat{\beta} = [H'V^{-1}H]^{-1} H'V^{-1} Y_D$$

is the generalized least squares estimate of β. The mean square error of $\hat{Y}(x_0)$ is

$$\mathrm{MSE}[\hat{Y}(x_0)] = \sigma^2 - (h'(x_0), v'_{x_0}) \begin{pmatrix} 0 & H'_D \\ H_D & V_D \end{pmatrix}^{-1} \begin{pmatrix} h(x_0) \\ v_{x_0} \end{pmatrix}.$$

The first component of Equation (4) is the generalized least squares prediction at point x_0 given the design covariance matrix V_D, while the second component "pulls" the generalized least squares

response surface through the observed data points. The elasticity of the response surface "pull" is solely determined by the correlation function $R(\cdot)$. The predictions at the design points are exactly the corresponding observations, and the mean square error equals zero. As a prediction point x_0 moves away from all of the design points, the second component of Equation (4) goes to zero, yielding the generalized least squares prediction, while the mean square error at that point goes to $\sigma^2 + h'(x_0)\left[H'V_D^{-1}H\right]^{-1} h(x_0)$. In fact, these results are true in the wide sense if the Gaussian assumption is removed.

As an example, consider an experiment where $n = 3$, $p = 1$, $\sigma^2 = .05$, $R(d) = \exp(-20d^2)$ and $D = \{.3, .5, .8\}$. The response of the unknown function at the design is $y_D' = (.7, .3, .5)$. The dashed line of Figure 1 is the generalized least squares prediction surface for $h(\cdot) \equiv 1$ where $\hat{\beta} = .524$. The effect of the second component of Equation (4) is to pull the dashed line through the observed design points as shown by the solid line. The shape of the surface or the amount of elasticity of the "pull" is determined by the vector $v_x' V_D^{-1}$ as a function of x and therefore is completely determined by $R(\cdot)$. The dotted lines are $\pm 2\sqrt{\mathrm{MSE}[\hat{Y}(x)]}$ or 95% pointwise confidence envelopes around the prediction surface. The interpretation of these pointwise confidence envelopes is that for any point x_0, if the unknown function is truly generated by a random function with constant mean and correlation function $R(d) = \exp(-20d^2)$, then approximately 95% of the sample paths that go through the observed design points would be between these dotted lines at x_0. The predictions and confidence intervals can be very different for different σ^2 and $R(\cdot)$. The effect of different correlation functions is discussed in Section 4.3. Clearly, the true function is not "generated" stochastically. The above model is used for prediction and to quantify the uncertainty of the prediction. This naturally leads to a Bayesian interpretation of this methodology.

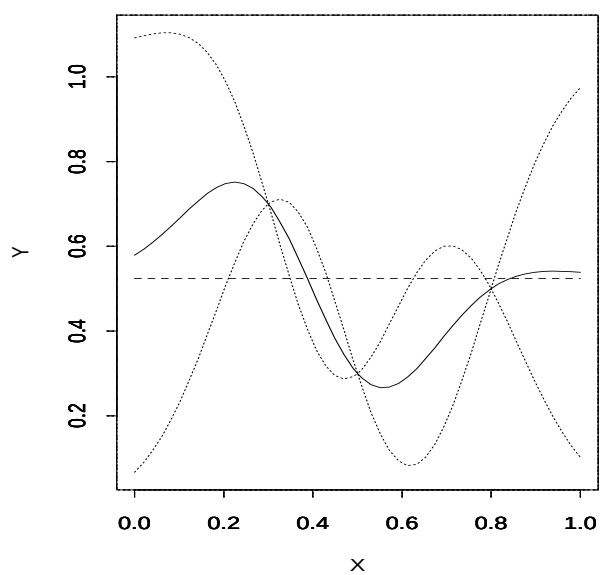

Figure 1: A prediction example with $n = 3$.

4.2 A fully Bayesian interpretation

An alternative to the above interpretation of Equation (2) is the fully Bayesian interpretation which uses the model as a way of quantifying the uncertainty of the unknown function. The Bayesian approach [13, 48] uses the same model but has a different interpretation of the β_j's. Here the β_j's are random variables with prior distribution π_j. The effect of these prior distributions is to quantify the prior belief of the unknown function or to put a prior distribution on a large class of functions $\mathcal{G}$. Hence hopefully the true function $y(\cdot) \in \mathcal{G}$. The mixed convolution of the $\pi_j's$ and $\pi(Z)$ yield the prior distribution $\Pi(G)$ for subsets of functions $G \subset \mathcal{G}$.

Once the data $Y_D = y_D$ has been observed, the posterior distribution $\Pi(G \mid Y_D)$ is calculated. The mean

$$\hat{Y}(x_0) = \int g(x_0)\Pi(g \mid Y_D = y_D)dg$$

and variance

$$\mathrm{Var}(\hat{Y}(x_0) \mid Y_D = y_D) = \int (g(x_0) - \hat{Y}(x_0))^2 \Pi(g \mid Y_D = y_D)dg$$

of the posterior distribution at each input point are then used as the predictor and measure of error, respectively, at that point. In general, the Kriging and Bayesian approaches will lead to different estimators. However, if the prior distribution of $Z(\cdot)$ is Gaussian and if the prior distribution of the β_j's is diffuse, then the two approaches yield identical estimators.

As an example, consider the case where the prior distribution of the vector of β's is

$$\beta \sim N_k(b, \tau^2\Sigma)$$

and the prior distribution of $Z(\cdot)$ is a stationary Gaussian distribution with expected value zero and covariance function given by Equation (3). After the simulator function has been evaluated at the experimental design, the posterior distribution of β is

$$\beta \mid Y_D \sim N_k\left(\tilde{\beta}, \tilde{\Sigma}\right)$$

where

$$\tilde{\beta} = \tilde{\Sigma}\left[H'V_D^{-1}Y_D + \tau^{-2}\Sigma^{-1}b\right]$$

and

$$\tilde{\Sigma} = \left[H'V_D^{-1}H + \tau^{-2}\Sigma^{-1}\right]^{-1}$$

and the posterior distribution of $Y(x_0)$ is

$$Y(x_0) \mid Y_D \sim N\left(v'_{x_0}V_D^{-1}Y_D + c'_{x_0}\tilde{\beta}, \sigma^2 - v'_{x_0}V_D^{-1}v_{x_0} + c'_{x_0}\tilde{\Sigma}c_{x_0}\right)$$

where

$$c'_{x_0} = h' - v'_{x_0}V_D^{-1}H.$$

Hence the posterior distribution is still Gaussian but it is no longer stationary. Now if $\tau^2 \to \infty$ then

$$\begin{aligned} \tilde{\beta} &\to \hat{\beta} \\ \tilde{\Sigma} &\to \left[H'V_D^{-1}H\right]^{-1} \end{aligned}$$

and hence the posterior variance of $Y(x_0)$ is

$$\begin{aligned}
\mathrm{Var}(Y(x_0) \mid Y_D) &= \sigma^2 - v'_{x_0} V_D^{-1} v_{x_0} + c'_{x_0} \left[H' V_D^{-1} H \right]^{-1} c_{x_0} \\
&= \sigma^2 - v'_{x_0} V_D^{-1} v_{x_0} + h' \left[H' V_D^{-1} H \right]^{-1} h \\
&\quad -2h' \left[H' V_D^{-1} H \right]^{-1} H' V_D^{-1} v_{x_0} \\
&\quad +v'_{x_0} V_D^{-1} H \left[H' V_D^{-1} H \right]^{-1} H' V_D^{-1} v_{x_0} \\
&= \sigma^2 - \left[-h' \left[H' V_D^{-1} H \right]^{-1} h + 2h' \left[H' V_D^{-1} H \right]^{-1} H' V_D^{-1} v_{x_0} \right] \\
&\quad - \left[v'_{x_0} (V_D^{-1} - V_D^{-1} H \left[H' V_D^{-1} H \right]^{-1} H' V_D^{-1}) v_{x_0} \right] \\
&= \sigma^2 - (h'(x_0), v'_{x_0}) \begin{pmatrix} 0 & H'_D \\ H_D & V_D \end{pmatrix}^{-1} \begin{pmatrix} h(x_0) \\ v_{x_0} \end{pmatrix}
\end{aligned}$$

which is the same variance as the BLUP in the Kriging approach. Therefore, if $Z(\cdot)$ has a Gaussian prior distribution and if the β's have a diffuse prior, the Bayesian and the Kriging approaches yield identical estimators.

Currin, Mitchell, Morris and Ylvisaker [13] provide a more in depth discussion of the Bayesian approach for the model with a fixed mean ($h \equiv 1$). O'Hagan [48] discusses Bayes Linear Estimators (BLE) and their connection to Equations (2) and (4). The Bayesian approach, which uses random functions as a method of quantifying the uncertainty of the unknown simulator function $Y(\cdot)$, is more subjective than the Kriging or frequentist approach. While both approaches require prior knowledge or an objective method of estimating the covariance function, the Bayesian approach additionally requires knowledge of parameters of the prior distribution of β (b and Σ). For this reason, the Kriging results and Bayesian approach with diffuse prior distributions and the Gaussian assumption are widely used in computer experiments.

4.3 Correlation functions

As discussed above, the selection of $R(\cdot)$ plays a crucial role in constructing designs and in the predictive process. Consider the example of Section 4.1 where $n = 3$, $p = 1$, $D = \{.3, .5, .8)$, $y'_d = \{.7, .3, .5\}$, $R(d) = \exp\{\theta d^2\}$ and $\theta = 20$. Figure 2(a) shows the effect on prediction for $\theta = 2$. Now $\hat{\beta} = 1.3$ and the surface elasticity is very low. The predictions outside of the design are actually higher than the observed surface since the convex nature of the observed response indicate that the design range contains a local minimum for the total process. Eventually, the extrapolations would return to the value of $\hat{\beta}$. Additionally, the 95% pointwise confidence intervals are much narrower within the range of the design than in Figure 1. Figure 2(b) displays the prediction when $\theta = 100$. Here $\hat{\beta} = .5$ and the surface elasticity is very high. The prediction line is typically .5 with smooth curves pulling the surface through the design points. The 95% pointwise confidence intervals are wider than before.

This section presents some simplifying restrictions on $R(\cdot)$ and four families of univariate correlation functions used in generating the simplified correlation functions. Examples of realization of these families will be shown to explain the effect on prediction by varying the parameter of these families. Furthermore, the maximum likelihood method for estimating the parameters of a correlation family along with a technique for implementation will be discussed in Section 4.4.

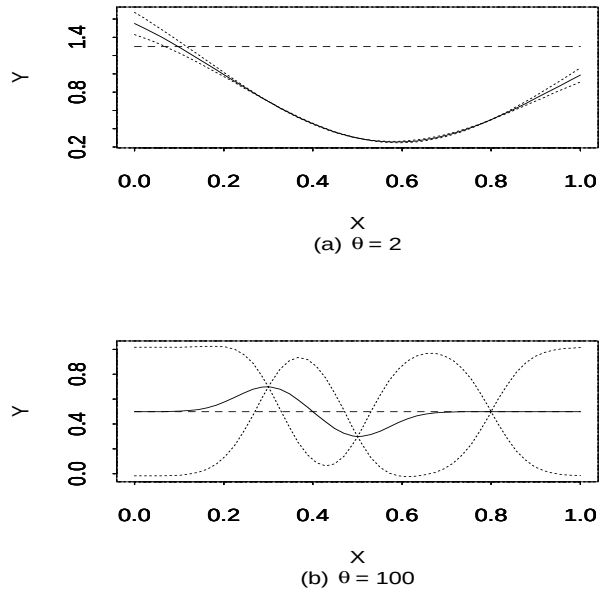

Figure 2: The effects of θ on prediction.

4.3.1 Restrictions on $R(\cdot)$

Any positive definite function R with $R(x, x) = 1$ could be used as a correlation function, but for simplicity, it is common to restrict $R(\cdot)$ such that for any $x_1, x_2 \in S$

$$R(x_1, x_2) = R(x_1 - x_2)$$

so that the process $Z(\cdot)$ is stationary. Some types of nonstationary behavior in the mean function of $Y(\cdot)$ can be modeled by the linear term in Equation (2). A further restriction makes the correlation function depend only on the magnitude of the distance.

$$R(x_1, x_2) = R(|x_1 - x_2|).$$

In higher dimensions ($p \geq 2$) a product correlation function,

$$R(x_1, x_2) = \prod_{j=1}^{p} R_j(|x_{1j} - x_{2j}|)$$

is often used for mathematical convenience. That is, $R(\cdot)$ is a product of univariate correlation functions and hence, only univariate correlation functions are of interest. The product correlation function has been used for prediction in spatial settings [89, 13, 69, 68, 87].

Several choices for the factors in the product correlation function are outlined below.

4.3.2 Cubic

The (univariate) cubic correlation family is parameterized by $\rho \in [0, 1]$ and $\gamma \in [0, 1]$ and is given for $d \in [0, 1]$ by

$$R(d) = 1 - \frac{3(1-\rho)}{2+\gamma} d^2 + \frac{(1-\rho)(1-\gamma)}{2+\gamma} |d|^3$$

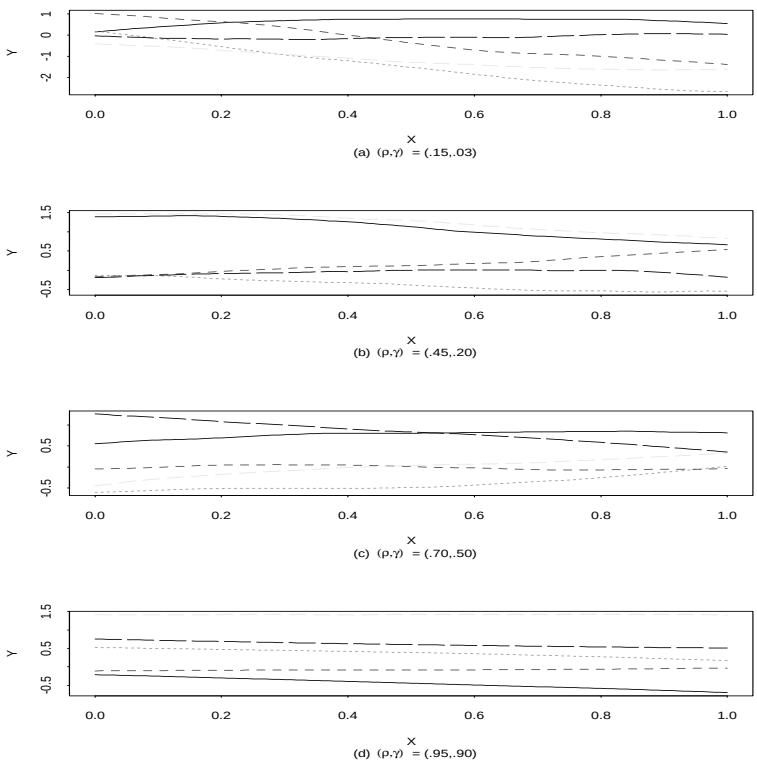

Figure 3: Realizations for the cubic correlation function (ρ, γ) = (a) (.15,.03), (b) (.45,.20), (c) (.70,.50), and (d) (.95,.90).

where ρ and γ are restricted by

$$\rho \geq \frac{5\gamma^2 + 8\gamma - 1}{\gamma^2 + 4\gamma + 7}$$

to ensure that the function is positive definite (See Mitchell, Morris and Ylvisaker [41]). Here $\rho = \text{corr}(Y(0), Y(1))$ is the correlation between endpoint observations and $\gamma = \text{corr}(Y'(0), Y'(1))$ is the correlation between endpoints of the derivative process. The cubic correlation function implies that the derivative process has a linear correlation process with parameter γ.

A prediction model in one dimension for this family is a cubic spline interpolator. In two dimensions, when the correlation is a product of univariate cubic correlation functions the predictions are piece-wise cubic in each variable.

Processes generated with the cubic correlation function are once mean square differentiable. Figure 3 shows several realizations of processes with the cubic correlation function and parameter pairs $(.15, .03)$, $(.45, .20)$, $(.70, .50)$, $(.95, .9)$. Notice that the realizations are quite smooth and almost linear for parameter pair $(.95, .90)$.

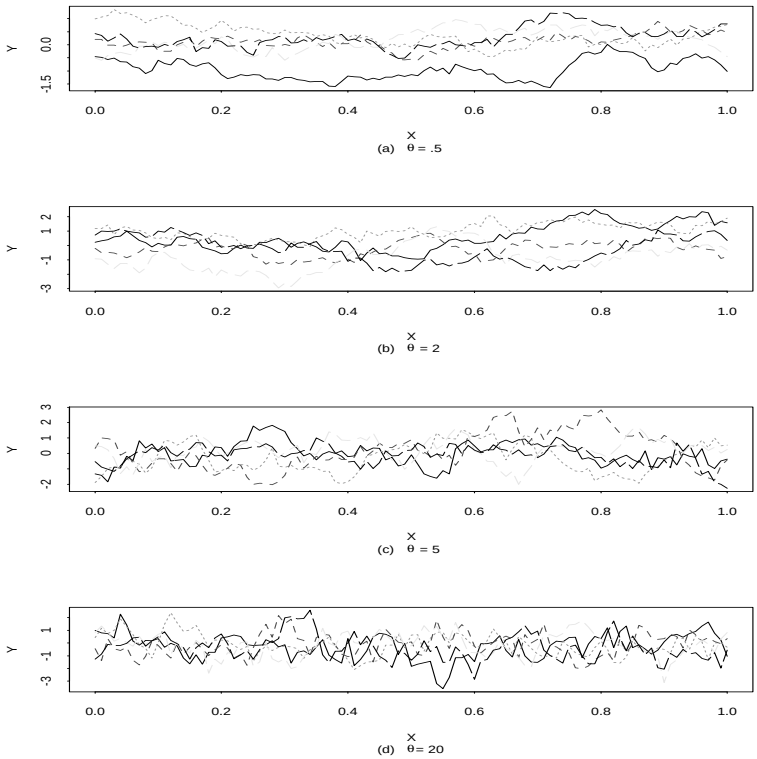

Figure 4: Realizations for the exponential correlation function with $\theta =$ (a) 0.5, (b) 2.0 (c) 5.0, and (d) 20.0.

4.3.3 Exponential

The (univariate) exponential correlation family is parameterized by $\theta \in (0, \infty)$ and is given by

$$R(d) = \exp(-\theta|d|)$$

for $d \in [0, 1]$. Processes with the exponential correlation function are Ornstein-Uhlenbeck processes [58]. The exponential correlation function is not mean square differentiable.

Figure 4 presents several realizations of one dimensional processes with the exponential correlation function and $\theta = 0.5, 2.0, 5.0, 20$. Figure 4a is for $\theta = 0.5$ and these realizations have very small global trends but much local variation. Figure 4d is for $\theta = 20$, and is very jumpy. Mitchell, Morris and Ylvisaker [41] also found necessary and sufficient conditions on the correlation function so that the derivative process has an exponential correlation function. These are called smoothed exponential correlation functions.

4.3.4 Gaussian

Sacks, Welch, Mitchell and Wynn [69] generalized the exponential correlation function by using

$$R(d) = \exp(-\theta|d|^q)$$

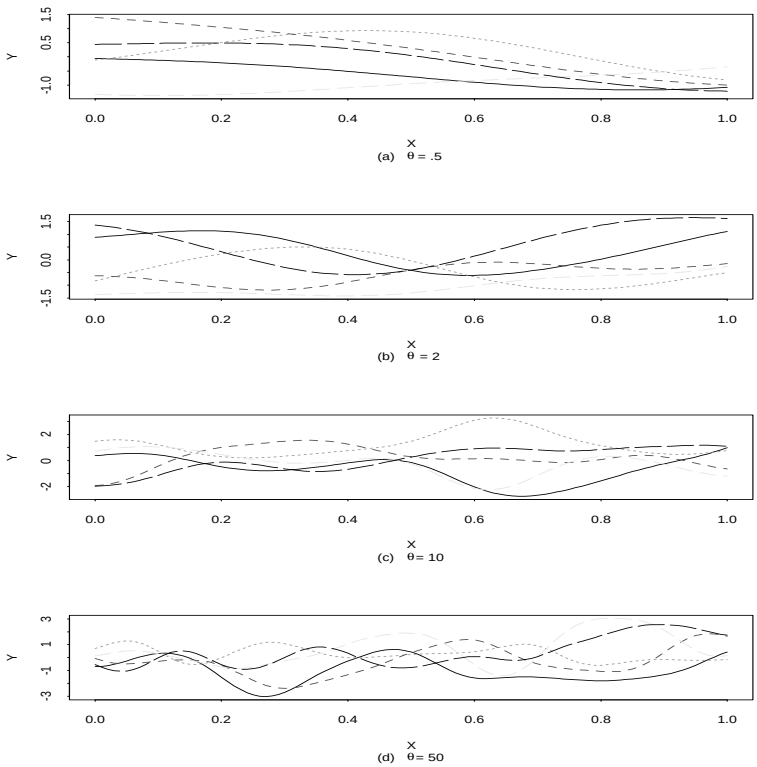

Figure 5: Realizations for the Gaussian correlation function with $\theta =$ (a) 0.5, (b) 2.0, (c) 10.0, and (d) 50.0.

where $0 < q \leq 2$ and $\theta \in (0, \infty)$. Taking $q = 1$ recovers the exponential correlation function. As q increases, this correlation function produces smoother realizations. However, as long as $q < 2$, these processes are not mean square differentiable.

The Gaussian correlation function is the case $q = 2$ and the associated processes are infinitely mean square differentiable. In the Bayesian interpretation, this correlation function puts all of the prior mass on analytic functions [13]. This correlation function is appropriate if the simulator output is known to be analytic. Figure 5 displays several realizations for various θ for the Gaussian correlation function. These realizations are very smooth, even when $\theta = 50$.

4.3.5 Matérn

All of the univariate correlation functions described above are either zero, once or infinitely many times mean square differentiable. Stein [77] recommends a more flexible family of correlation function [35, 88]. The Matérn correlation function is parameterized by $\theta \in (0, \infty)$ and $\nu \in (-1, \infty)$ and is given by

$$R(d) = \frac{(\theta|d|)^{\nu}}{\Gamma(\nu)2^{\nu-1}} K_{\nu}(\theta|d|)$$

where $K_{\nu}(\cdot)$ is a modified Bessel function of order ν. The associated process will be m times differentiable if and only if $\nu > m$. Hence, the amount of differentiability can be controlled by

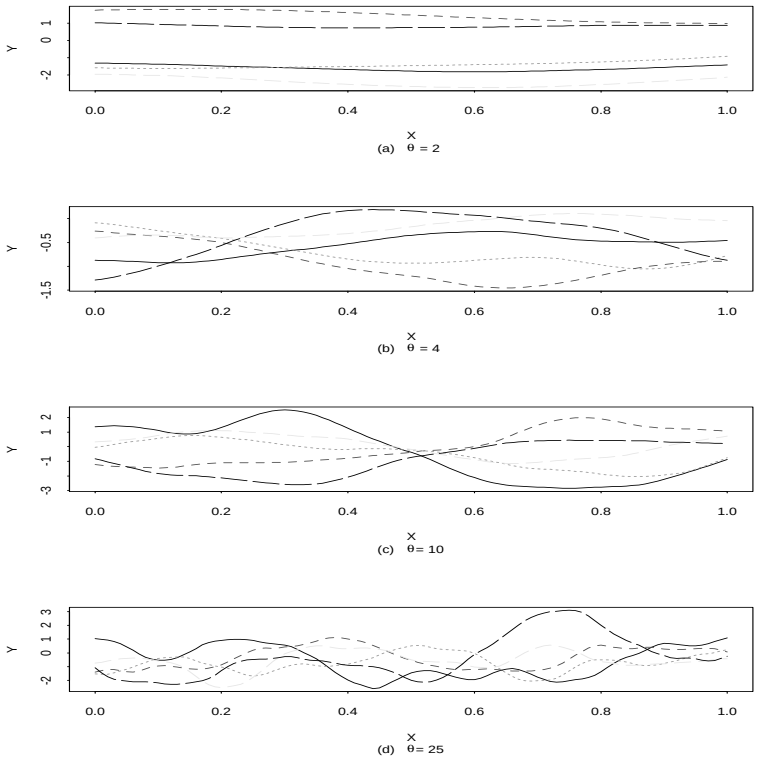

Figure 6: Realizations for the Bessel correlation function with $\nu = 2.5$ and $\theta =$ (a) 2.0, (b) 4.0, (c) 10.0, and (d) 25.0.

ν while θ controls the range of the correlations. This correlation family is more flexible than the other correlation families described above due to the control of the differentiability of the predictive surface.

Figure 6 displays several realizations of processes with the Bessel correlation function with $\nu = 2.5$ and various values of θ. For small values of θ, the realizations are very smooth and flat while the realizations are erratic for large values of θ.

4.3.6 Summary

The correlation functions described above have been applied in computer experiments. Software for predicting with them is described in [30]. The cubic correlation function yields predictions that are cubic splines. The exponential predictions are non-differentiable while the Gaussian predictions are infinitely differentiable. The Matérn correlation function is the most flexible since the degree of differentiability and the smoothness of the predictions can be controlled. In general, enough prior information to fix the parameters of a particular correlation family and σ^2 will not be available. A pure Bayesian approach would place a prior distribution on the parameters of a family and use the posterior distribution of the parameter in the estimation process. Alternatively, an empirical Bayes approach which uses the data to estimate the parameters of a correlation family and σ^2 is often used. The maximum likelihood estimation procedure will be presented and discussed in the

next section.

4.4 Correlation function estimation - maximum likelihood

The previous subsections of this section presented the Kriging model, and families of correlation functions. The families of correlations are all parameterized by one or two parameters which control the range of correlation and the smoothness of the corresponding processes. This model assumes that σ^2, the family and parameters of $R(\cdot)$ are known. In general, these values are not completely known a priori. The appropriate correlation family might be known from the simulator's designers experience regarding the smoothness of the function. Also, ranges for σ^2 and the parameters of $R(\cdot)$ might be known if a similar computer experiment has been performed. A pure Bayesian approach is to quantify this knowledge into a prior distribution on σ^2 and $R(\cdot)$. How to distribute a non-informative prior across the different correlation families and within each family is unclear. Furthermore, the calculation of the posterior distribution is generally intractable.

An alternative and more objective method of estimating these parameters is an empirical Bayes approach which finds the parameters which are most consistent with the observed data. This section presents the maximum likelihood method for estimating β, σ^2 and the parameters of a fixed correlation family when the underlying distribution of $Z(\cdot)$ is Gaussian. The best parameter set from each correlation family can be evaluated to find the overall "best" σ^2 and $R(\cdot)$.

Consider the case where the distribution of $Z(\cdot)$ is Gaussian. Then the distribution for the response at the n design points Y_D is multinormal and the likelihood is given by

$$\begin{aligned} \mathrm{lik}(\beta, \sigma^2, R \mid Y_D) &= (2\pi)^{-n/2}\, \sigma^{-n}\, |R_D|^{-1/2} \\ &\quad \times \exp\left\{ -\frac{1}{2\sigma^2} (Y_D - H\beta)' R_D^{-1} (Y_D - H\beta) \right\} \end{aligned}$$

where R_D is the design correlation matrix. The log likelihood is

$$\begin{aligned} l_{ml}\left(\beta, \sigma^2, R_D \mid Y_D\right) &= -\frac{n}{2} \ln(2\pi) - \frac{n}{2} \ln\left(\sigma^2\right) - \frac{1}{2} \ln(|R_D|) \\ &\quad -\frac{1}{2\sigma^2} (Y_D - H\beta)' R_D^{-1} (Y_D - H\beta). \end{aligned} \tag{5}$$

Hence

$$\frac{\partial l_{ml}(\beta, \sigma^2, R \mid Y_D)}{\partial \beta} = -\frac{1}{\sigma^2} \left(H' R_D^{-1} Y_D - H' R_D^{-1} H \beta \right)$$

which when set to zero yields the maximum likelihood estimate of β which is the same as the generalized least squares estimate,

$$\hat{\beta}_{ml} = \left[H' R_D^{-1} H \right]^{-1} H' R_D^{-1} Y_D. \tag{6}$$

Similarly,

$$\frac{\partial l_{ml}(\beta, \sigma^2, R_D \mid Y_D)}{\partial \sigma^2} = -\frac{n}{2\sigma^2} + \frac{1}{2\sigma^4} (Y_D - H\beta)' R_D^{-1} (Y_D - H\beta)$$

which when set to zero yields the maximum likelihood estimate of σ^2

$$\hat{\sigma}^2_{ml} = \frac{1}{n} \left(Y_D - H\hat{\beta} \right)' R_D^{-1} \left(Y_D - H\hat{\beta} \right). \tag{7}$$

Therefore, if R_D is known, the maximum likelihood estimates of β and σ^2 are easily calculated. However, if $R(\cdot)$ is parameterized by $\theta = (\theta_1, \cdots, \theta_s)$,

$$\begin{aligned}\frac{\partial l_{ml}(\beta, \sigma^2, R_D \mid Y_D)}{\partial \theta_i} &= -\frac{1}{2}\frac{\partial |R_D|}{\partial \theta_i} - \frac{1}{2\sigma^2}(Y_D - H\beta)' \frac{\partial R_D^{-1}}{\partial \theta_i}(Y_D - H\beta) \\ &= -\frac{1}{2}\ \mathrm{tr}\left\{R_D^{-1}\frac{\partial R_D}{\partial \theta_i}\right\} \\ &\quad + \frac{1}{2\sigma^2}(Y_D - H\beta)' R_D^{-1}\frac{\partial R_D}{\partial \theta_i} R_D^{-1}(Y_D - H\beta) \qquad (8)\end{aligned}$$

does not generally yield an analytic solution for θ when set to zero for $i = 1, \cdots, s$. (Commonly $s = p$ or $2p$, but this need not be assumed.)

An alternative method of estimating θ is to use a nonlinear optimization routine using Equation (5) as the function to be optimized. For a given value of θ, estimates of β and σ^2 are calculated using Equations (6) and (7), respectively. Next, Equation (8) is used in calculating the partial derivatives of the objective function. See Mardia and Marshall [34] for an overview of the maximum likelihood procedure.

4.5 Estimating and using derivatives

In the manufacturing sciences, deterministic simulators help describe the relationships between product design, and the manufacturing process to the product's final characteristics. This allows the product to be designed and manufactured efficiently. Equally important are the effects of uncontrollable variation in the manufacturing parameters to the end product. If the product's characteristics are sensitive to slight variations in the manufacturing process, the yield, or percentage of marketable units produced, may decrease. Furthermore, understanding the sensitivities of the product's characteristics can help design more reliable products and increase the overall quality of the product.

Many simulators need to solve differential equations and can provide the gradient of the response at a design point with little or no additional computational cost. However, some simulators require that the gradient be approximated by a difference equation. Then the cost of finding a directional derivative at a point is equal to evaluating an additional point while approximating the total gradient requires p additional runs.

Consider Figure 7 for an example in $p = 1$ showing the effects of including gradient information on prediction. The solid lines, Y in Figure 7(a) and Y' in Figure 7(b), are the true function and it's derivative, respectively, while the long dashed lines are Kriging predictors $\hat{Y}_3$ and $\hat{Y}'_3$ based on $n = 3$ observations. As expected $\hat{Y}_3$ goes through the design points, $D = \{.2, .5, .8\}$, but $\hat{Y}'_3$ is a poor predictor of Y'. The short dashed lines are the $n = 3$ predictors with derivative information $\hat{Y}_{3'}$ and $\hat{Y}'_{3'}$. Notice that this predictor now matches Y' and Y at D and the interpolations are over all much better. The addition of gradient information substantially improves the fits of both Y and Y'. The dotted lines are the $n = 6$ predictors $\hat{Y}_6$ and $\hat{Y}'_6$ and is a fairer comparison if the derivative costs are equal to the response cost. The predictor $\hat{Y}_6$ is a little better on the interior of S but $\hat{Y}'_6$ is worse at $x = 0$ than $\hat{Y}'_{3'}$.

The Kriging methodology easily extends to model gradients. To see this for $p = 1$, let $\mathrm{E}[Y(\cdot)] = \mu$ and $d = t_2 - t_1$, then

$$\mathrm{Cov}\left[Y(t_1), Y'(t_2)\right] = \mathrm{E}\left[Y(t_1)Y'(t_2)\right] - \mathrm{E}\left[Y(t_1)\right]\mathrm{E}\left[Y'(t_2)\right].$$

Now due to the stationarity of $Y(\cdot)$, $\mathrm{E}[Y'(\cdot)] = 0$ and

$$\mathrm{Cov}\left[Y(t_1), Y'(t_2)\right] = \mathrm{E}\left[Y(t_1)Y'(t_2)\right]$$

$$
\begin{aligned}
&= \mathrm{E}\left[Y(t_1)\lim_{\delta\to 0}\frac{Y(t_2+\delta)-Y(t_2)}{\delta}\right] \\
&= \mathrm{E}\left[\lim_{\delta\to 0}\frac{Y(t_1)Y(x_2+\delta)-Y(t_1)Y(x_2)}{\delta}\right] \\
&= \sigma^2\lim_{\delta\to 0}\frac{R(d+\delta)-R(d)}{\delta} \\
&= \sigma^2 R'(d)
\end{aligned}
$$

for differentiable $R(\cdot)$. Similarly,

$$\mathrm{Cov}\left[Y'(t_1), Y(t_2)\right] = -\sigma^2 R'(d)$$

and

$$\mathrm{Cov}\left[Y'(t_1), Y'(t_2)\right] = -\sigma^2 R''(d)$$

For more general p and for higher derivatives, following Morris, Mitchell and Ylvisaker [45] let

$$Y^{(a_1,\ldots,a_p)}(t) = \frac{\partial^a}{\partial t_1^{(a_1)}\cdots\partial t_p^{(a_p)}}Y(t)$$

where $a = \sum_{j=1}^p a_j$ and t_j is the jth component of t. Then $\mathrm{E}[Y^{(a_1,\ldots,a_p)}] = 0$ and

$$\mathrm{Cov}\left[Y^{(a_1,\ldots,a_p)}(t_1), Y^{(b_1,\ldots,b_p)}(t_2)\right] = (-1)^a\sigma^2\prod_{j=1}^p R_j^{(a_j+b_j)}(t_{2j}-t_{1j})$$

for $R(d) = \prod_{j=1}^p R_j(d_j)$.

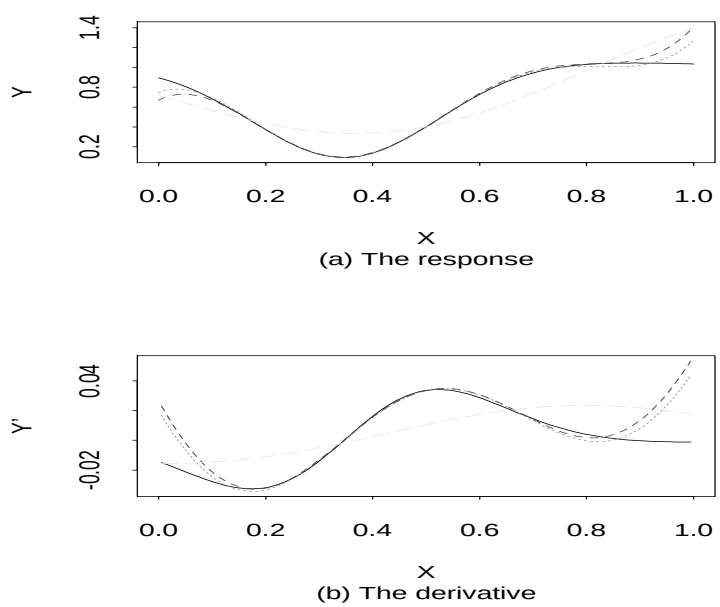

Figure 7: (a) An example of a response (Y) and three predictors ($\hat{Y}_3, \hat{Y}_{3'}, \hat{Y}_6$). (b) An example of a derivative (Y') and three predictors ($\hat{Y'}_3, \hat{Y'}_{3'}, \hat{Y'}_6$).

Furthermore, for directional derivatives, let $Y'_\nu(t)$ be the directional derivative of $Y(t)$ in the direction $\nu = (\nu_1, \ldots, \nu_p)'$, $\sum_{j=1}^p \nu_j^2 = 1$,

$$Y'_\nu(t) = \sum_{j=1}^p \frac{\partial Y(t)}{\partial t_j} \nu_j = \langle \bigtriangledown Y(t), \nu \rangle .$$

Then $\mathrm{E}[Y'_\nu(t)] = 0$ and for $d = t - s$,

$$\begin{aligned}
\mathrm{Cov}\left[Y(s), Y'_\nu(t)\right] &= \mathrm{E}\left[Y(s)Y'(t)\right] \\
&= \sum_{j=1}^p \mathrm{E}\left[Y(s)\frac{\partial Y(t)}{\partial t_j}\nu_j\right] \\
&= \sum_{j=1}^p \mathrm{Cov}\left[Y(s), \frac{\partial Y(t)}{\partial t_j}\right]\nu_j \\
&= \sigma^2 \sum_{j=1}^p \frac{\partial \dot{R}(d)}{\partial d_j}\nu_j \\
&= \sigma^2 \left\langle \dot{R}(d), \nu \right\rangle \qquad (9)
\end{aligned}$$

where $\dot{R}(d) = [\partial R(d)/\partial d_1, \ldots, \partial R(d)/\partial d_p]'$. Similarly,

$$\mathrm{Cov}\left[Y'_\nu(s), Y(t)\right] = -\sigma^2 \left\langle \dot{R}(d), \nu \right\rangle \qquad (10)$$

and

$$\mathrm{Cov}\left[Y'_{\nu_s}(s), Y'_{\nu_t}(t)\right] = -\sigma^2 \nu'_s \ddot{R}(d) \nu_t \qquad (11)$$

where

$$\left(\ddot{R}(d)\right)_{kl} = \frac{\partial^2 R(d)}{\partial d_k \partial d_l}$$

is the matrix of 2nd partial derivatives evaluated at d.

The Kriging methodology is modified to model gradient information by letting

$$y^*_D = \left[y(x_1), \ldots, y(x_n), y'_{\nu_{11}}(x_1), y'_{\nu_{12}}(x_1), \ldots, y'_{\nu_{nm}}(x_n)\right]'$$

where ν_{il} is the direction of the lth directional derivative at x_i. Also let

$$\mu^* = (\mu, \mu, \ldots, \mu, 0, 0, \ldots, 0)'$$

with n μs and mn 0s and let V^* be the combined covariance matrix for the design responses and derivatives with the entries as prescribed above (Equations (9), (10), and (11)). Then

$$\hat{Y}(x_0) = \mu + v'^*_{x_0} V^{*^{-1}} \left(y^*_D - \mu^*\right)$$

and

$$\hat{Y}'_\nu(x_0) = v'^*_{x_0,\nu} V^{*^{-1}} \left(y^*_D - \mu^*\right)$$

where $v'^*_{x_0} = \mathrm{Cov}[Y(x_0), Y^*_D]$, and $v'^*_{x_0,\nu} = \mathrm{Cov}[Y'(x_0), Y^*_D]$.

Notice that once differentiable random functions need twice differentiable correlation functions. One problem with using the total gradient information is the rapid increase in the covariance matrix. For each additional design point, V^* increases by $p + 1$ rows and columns. Fortunately, these new rows and columns generally have lower correlations than the corresponding rows and columns for an equal number of response. The inversion of V^* is more computationally stable than for an equally sized V_D. More research is needed to provide general guidelines for using gradient information efficiently.

4.6 Complexity of Computer Experiments

Recent progress in complexity theory, a branch of theoretical computer science, has shed some light on computer experiments. The dissertation of Ritter [63] contains an excellent summary of this area. Consider the case where $Y(x) = Z(x)$, that is where there is no regression function. If for $r \geq 1$ all of the r'th order partial derivatives of $Z(x)$ exist in the mean square sense and obey a Holder condition of order β, then it is possible (see Ritter, Wasilkowski and Wozniakowski [64]) to approximate $Z(x)$ with an L^2 error that decays as $O(n^{-(r+\beta)/p})$. This error is a root mean square average over randomly generated functions Z.

When the covariance has a tensor product form, like those considered here, one can do even better. Ritter, Wasilkowski and Wozniakowski [65] show that the error rate for approximation in this case is $n^{-r-1/2}(\log n)^{(p-1)(r+1)}$ for products of covariances satisfying Sacks-Ylvisaker conditions of order $r \geq 0$. When Z is a p dimensional Wiener sheet process, for which $r = 0$, the result is $n^{-1/2}(\log n)^{(p-1)}$ which was first established by Wozniakowski [85].

In the general case, the rate for integration is $n^{-1/2}$ times the rate for approximation. A theorem of Wasilkowski [84] shows that a rate n^{-d} for approximation can usually be turned into a rate $n^{-d-1/2}$ for integration by the simple device of fitting an approximation with $n/2$ function evaluations, integrating the approximation, and then adjusting the result by the average approximation error on $n/2$ more Monte Carlo function evaluations. For tensor product kernels the rate for integration is $n^{-r-1}(\log n)^{(p-1)/2}$ (see Paskov [56]), which has a more favorable power of $\log n$ than would arise via Wasilkowski's theorem.

The fact that much better rates are possible under tensor product models than for general covariances suggests that the tensor product assumption may be a very strong one. The tensor product assumption is at least strong enough that under it, there is no average case curse of dimensionality for approximation.

5 Bayesian designs

Selecting an experimental design, D, is a key issue in building an efficient and informative kriging model. Since there is no random error in this model, we wish to find designs that minimize squared-bias. While some experimental design theories [7, 78] do investigate the case where bias rather than solely variance plays a crucial role in the error of the fitted model, how good these designs are for the pure bias problem of computer experiments is unclear. Box and Draper [7] studied the effect of scaling factorial designs by using a first order polynomial model when the true function is a quadratic polynomial. Box and Draper [8] extended the results to using a quadratic polynomial model when the true response surface is a cubic polynomial. They found that mean squared-error optimal designs are close to bias optimal designs. Steinberg [78] extended these ideas further by using a prior model proposed by Young [90] that puts prior distributions on the coefficients of a sufficiently large polynomial. However, model (2) is more flexible than high ordered polynomials and therefore better designs are needed.

This section introduces four design optimality criteria for use with computer experiments: entropy, mean squared-error, maximin and minimax designs. Entropy designs maximize the amount of information expected for the design while mean squared-error designs minimize the expected mean squared-error. Both these designs require a priori knowledge of the correlation function $R(\cdot)$. The design criteria described below are for the case of fixed design size n. Simple sequential designs, where the location of the nth design point is determined after the first $n-1$ points have been evaluated, will not be presented due to their tendencies to replicate [69]. However, sequential block strategies could be used where the above designs could be used as starting blocks. Depending upon

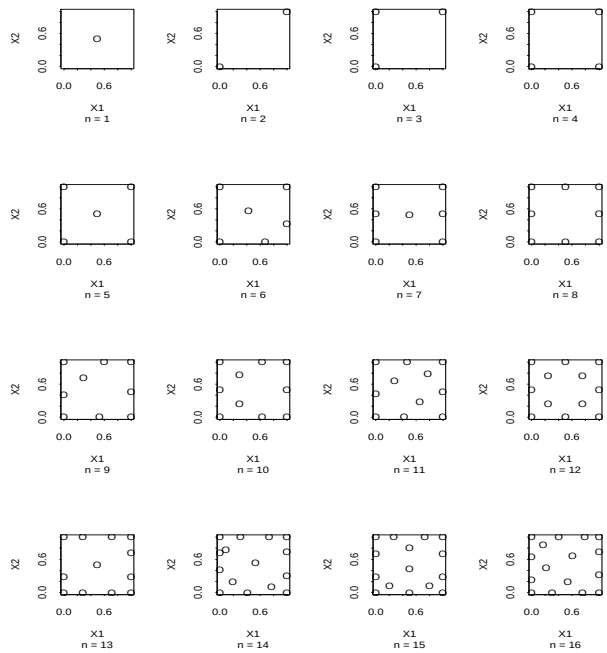

Figure 8: (a) Maximum entropy designs for $p = 2$, $n = 1$–16, and the Gaussian correlation function with $\theta = (0.5, 0.5)$.

the ultimate goal of the computer experiment, the first design block might be utilized to refine the design and reduce the design space.

5.1 Entropy designs

Lindley [31] introduced a measure, based upon Shannon's entropy [70], of the amount of information provided by an experiment. This Bayesian measure uses the expected reduction in entropy as a design criterion. This criterion has been used in Box and Hill [9] and Borth [6] for model discrimination. Shewry and Wynn [72] showed that, if the design space is discrete (i.e., a lattice in $[0,1]^p$), then minimizing the expected posterior entropy is equivalent to maximizing the prior entropy.

Definition 1 *A design D_E is a* <u>*Maximum Entropy Design*</u> *if*

$$\mathrm{E}_Y\left[-\ln P(Y_{D_E})\right] = \min_D \mathrm{E}_Y\left[-\ln P(Y_D)\right]$$

where $P(Y_D)$ is the density of Y_D.

In the Gaussian case, this is equivalent to finding a design that maximizes the determinant of the variance of Y_D. In the Gaussian prior case, where $\beta \sim N_k(b, \tau^2\Sigma)$, the determinant of the unconditioned covariance matrix is

$$\left|V_D + \tau^2 H\Sigma H'\right| = \begin{vmatrix} V_D + \tau^2 H\Sigma H' & H \\ 0 & I \end{vmatrix}$$

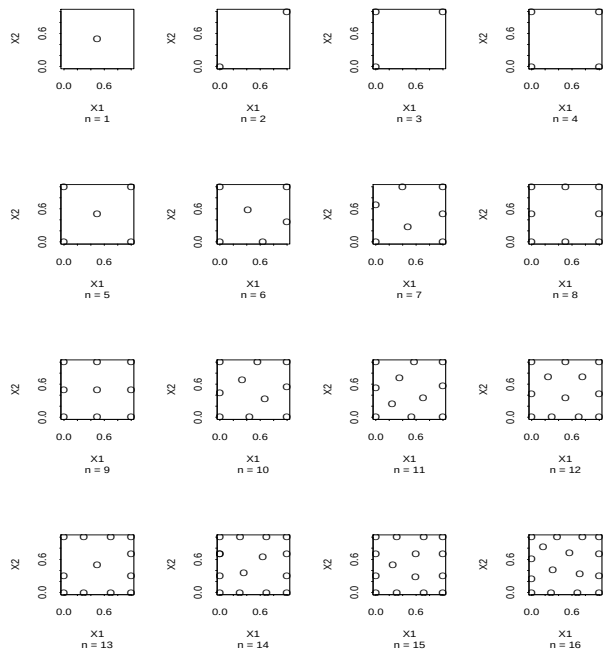

Figure 8: (b) Maximum entropy designs for $p = 2$, $n = 1$–16, and the Gaussian correlation function with $\theta = (2, 2)$.

$$
\begin{aligned}
&= \left| \begin{pmatrix} V_D & H \\ -\tau^2 \Sigma H' & I \end{pmatrix} \begin{pmatrix} I & 0 \\ \tau^2 \Sigma H' & I \end{pmatrix} \right| \\
&= \begin{vmatrix} V_D & H \\ -\tau^2 \Sigma H' & I \end{vmatrix} \\
&= \left| \begin{pmatrix} I & 0 \\ \tau^2 \Sigma H' V_D^{-1} & I \end{pmatrix} \begin{pmatrix} V_D & H \\ -\tau^2 \Sigma H' & I \end{pmatrix} \right| \\
&= \begin{vmatrix} V_D & H \\ 0 & \tau^2 \Sigma H' V_D^{-1} H + I \end{vmatrix} \\
&= |V_D| \left| \tau^2 \Sigma H' V_D^{-1} H + I \right| \\
&= |V_D| \left| H' V_D^{-1} H + \tau^{-2} \Sigma^{-1} \right| \left| \tau^2 \Sigma \right| .
\end{aligned}
$$

Since $\tau^2 \Sigma$ is fixed, the maximum entropy criterion is equivalent to finding the design D_E that maximizes

$$|V_D| \left| H' V_D^{-1} H + \tau^{-2} \Sigma^{-1} \right| .$$

If the prior distribution is diffuse, $\tau^2 \to \infty$, the maximum entropy criterion is equivalent to

$$|V_D| \left| H' V_D^{-1} H \right|$$

and if β is treated as fixed, then the maximum entropy criterion is equivalent to $|V_D|$.

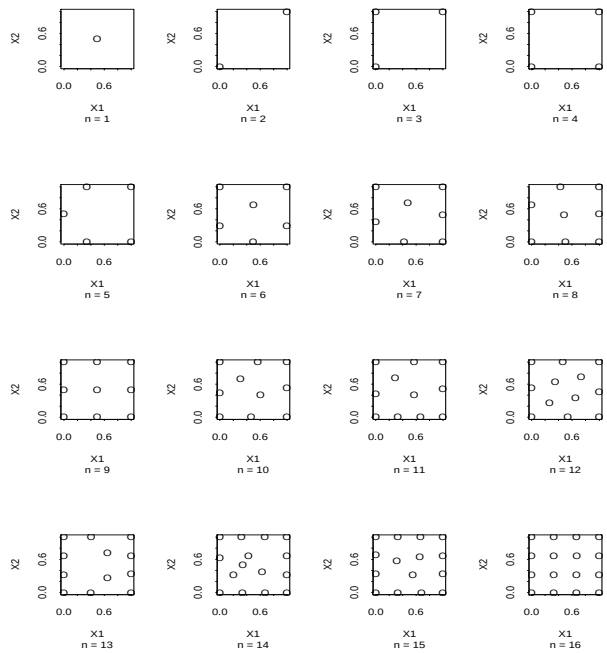

Figure 8: (c) Maximum entropy designs for $p = 2$, $n = 1$–16, and the Gaussian correlation function with $\theta = (10, 10)$.

Shewry and Wynn [72, 73] applied this measure in designs for spatial models. Currin et al [13] and Mitchell and Scott [43] have applied the entropy measure to finding designs for computer experiments. By this measure, the amount of information in experimental design is dependent on the prior knowledge of $Z(\cdot)$ through $R(\cdot)$. In general, $R(\cdot)$ will not be known a priori. Additionally, these optimal designs are difficult to construct due to the required $n \times p$ dimensional optimization of the n design point locations. Currin et al [13] describe an algorithm adopted from DETMAX [40] which successively removes and adds points to improve the design.

Figure 8 shows the optimal entropy designs for $p = 2$, $n = 1, \ldots, 16$, $R(d) = \exp\{-\theta \sum d_j^2\}$ where $\theta = 0.5, 2, 10$. The entropy designs tend to spread the points out in the plane and favor the edge of the design space over the interior. For example, the $n = 16$ designs displayed in Figure 8 have 12 points on the edge and only 4 points in the interior. Furthermore, most of the designs are similar across the different correlation functions although there are some differences. Generally, the ratio of the edge to interior points are constant. The entropy criterion appears to be insensitive to changes in the location of the interior points. Johnson, Moore, and Ylvisaker [28] indicate that entropy designs for extremely "weak" correlation functions are in a limiting sense maximin designs (see Section 5.3).

5.2 Mean squared-error designs

Box and Draper [7] proposed minimizing the normalized integrated mean squared-error (IMSE) of $\hat{Y}(x)$ over $[0, 1]^p$. Welch [86] extended this measure to the case when the bias is more complicated. Sacks and Schiller [67] and Sacks et al [68] discuss in more detail IMSE designs for computer experiments.

Definition 2 *A design D_I is an* <u>*Integrated Mean Squared-Error (IMSE) design*</u> *if*

$$J(D_I) = \min_D J(D)$$

where

$$J(D) = \frac{1}{\sigma^2} \int_{[0,1]^p} \mathrm{E}\left[Y(x) - \hat{Y}(x)\right]^2 dx.$$

$J(D)$ is dependent on $R(\cdot)$ through $Y(x)$. For any design, $J(D)$ can be expressed as

$$J(D) = \sigma^2 - \mathrm{trace}\left\{ \begin{bmatrix} 0 & H' \\ H & V_D \end{bmatrix}^{-1} \int \begin{bmatrix} h(x)h'(x) & h(x)v_x' \\ v_x h'(x) & v_x v_x' \end{bmatrix} dx \right\}$$

and, as pointed out by Sacks et al [68], if the elements of $h(x)$ and V_x are products of functions of a single input variable, the multidimensional integral simplifies to products of one-dimensional integrals. As in the entropy design criterion, the minimization of $J(D)$ is a optimization in $n \times p$ dimensions and is also dependent on $R(\cdot)$.

Sacks and Schiller [67] describe the use of a simulated annealing method for constructing IMSE designs for bounded and discrete design spaces. Sacks [69] use a quasi-Newton optimizer on a Cray X-MP48. They found that optimizing a $n = 16$, $p = 6$ design with $\theta_1 = \cdots = \theta_6 = 2$ took 11 minutes. The PACE program [30] uses the optimization program NPSOL [21] to solve the IMSE optimization for a continuous design space. For $n = 16, p = 6$, this optimization requires 13 minutes on a DEC3100, a much less powerful machine than the Cray. Generally, these algorithms can find only local minima and therefore many random starts are required.

Since $J(D)$ is dependent on $R(\cdot)$, robust designs need to be found for general $R(\cdot)$. Sacks et al [68] found that for $n = 9, p = 2$ and $R(d) = \exp\{-\theta \sum_{j=1}^{2} d_j^2\}$ (see Section 4.3.4 for details on the Gaussian correlation function) the IMSE design for $\theta = 1$ is robust in terms of relative efficiency. However, this analysis used a quadratic polynomial model and the results may not extend to higher dimensions nor different linear model components. Sacks [69] used the optimal design for the Gaussian correlation function with $\theta = 2$ for design efficiency-robustness.

Figure 9 displays IMSE designs for $p = 2$ and $n = 1, \cdots, 9$ for $\theta = .5, 2, 10$. The designs, in general lie in the interior of S. For fixed design size n, the designs usually are similar geometrically for different θ values with the scale decreasing as θ increases. They have much symmetry for some values of n, particularly $n = 12$. Notice that for the case when $n = 5$ that the design only takes on three unique values for each of the input variables. These designs tend to have clumped projections onto lower dimension marginals of the input space. Better projection properties are needed when the true function is only dependent on a subset of the input variables.

5.3 Maximin and minimax designs

Johnson, Moore and Ylvisaker [28] developed the idea of minimax and maximin designs. These designs are dependent on a distance measure or metric. Let $d(\cdot,\cdot)$ be a metric on $[0,1]^p$. Hence $\forall x_1, x_2, x_3 \in [0,1]^p$,

$$\begin{aligned} d(x_1, x_2) &= d(x_2, x_1) \\ d(x_1, x_2) &\geq 0 \\ d(x_1, x_2) = 0 &\iff x_1 = x_2 \\ d(x_1, x_2) &\leq d(x_1, x_3) + d(x_3, x_2) \end{aligned}$$

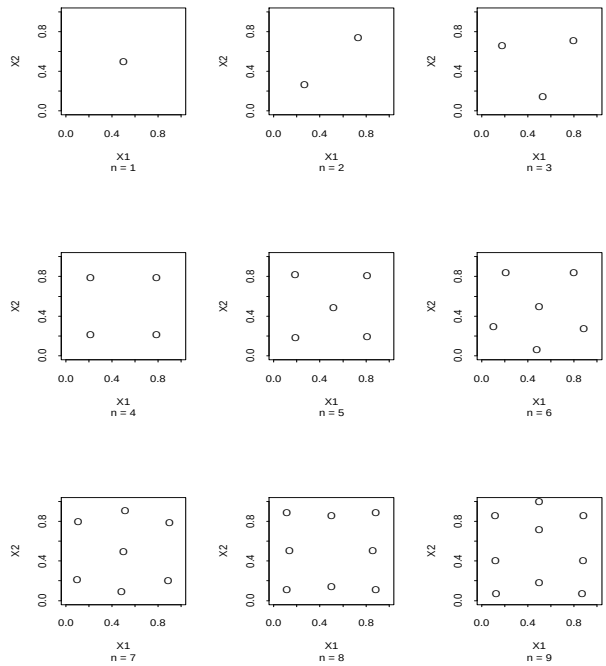

Figure 9: (a) Minimum integrated mean square error designs for $p = 2$, $n = 1$–9, and the Gaussian correlation function with $\theta = (.5, .5)$.

Definition 3 *Design D_{MI} is a* <u>*minimax distance design*</u> *if*

$$\min_{D} \max_{x} d(x, D) = \max_{x} d(x, D_{MI})$$

where

$$d(x, D) = \min_{x_0 \in D} d(x, x_0).$$

Minimax distance designs ensure that all points in $[0, 1]^p$ are not too far from a design point. Let $d(\cdot, \cdot)$ be Euclidean distance and consider placing a p-dimensional sphere with radius r around each design point. The idea of a minimax design is to place the n points so that the design space is covered by the spheres with minimal r. As an illustration, consider the owner of a petroleum corporation who wants to open some franchise gas stations. The gas company would like to locate the stations in the most convenient sites for the customers. A minimax strategy of placing gas stations would ensure that no customer is too far from one of the company's stations.

Figure 10(a) shows a minimax design for $p = 2$ and $n = 6$ with $d(\cdot, \cdot)$ being Euclidean distance. The maximum distance to a design point is .318. For small n, minimax designs will generally lie in the interior of the design space.

Definition 4 *A design D_{MA} is a* <u>*maximin distance design*</u> *if*

$$\max_{D} \min_{x_1, x_2 \in D} d(x_1, x_2) = \min_{x_1, x_2 \in D_{MA}} d(x_1, x_2)$$

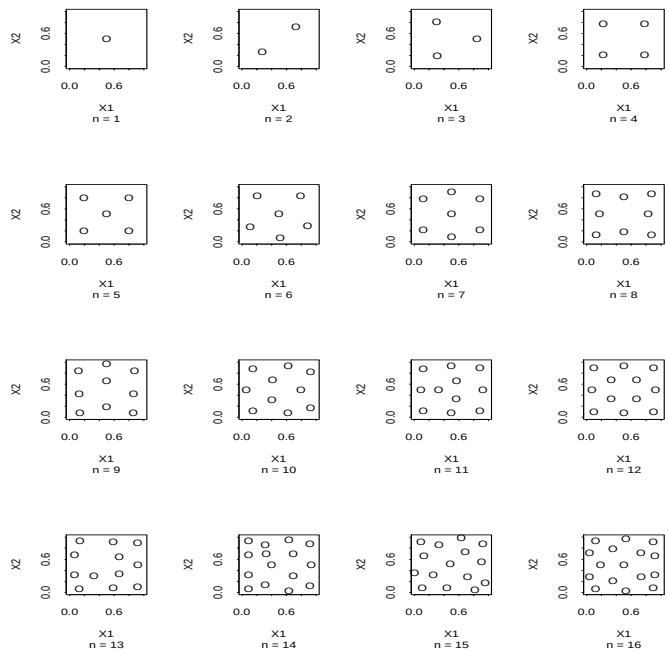

Figure 9: (b) Minimum integrated mean square error designs for $p = 2$, $n = 1$–16, and the Gaussian correlation function with $\theta = (2, 2)$.

Again, let $d(\cdot, \cdot)$ be Euclidean distance. Maximin designs pack the n design points, with their associated spheres, into the design space, S, with maximum radius. Parts of the sphere may be out of S but the design points must be in S. Analogous to the minimax illustration above is the position of the owners the gas station franchises. They wish to minimize the competition from each other by locating the stations as far apart as possible. A maximin strategy for placing the franchises would ensure that no two stations are too close to each other.

Figure 10(b) shows a maximin design for $p = 2, n = 6$ and $d(\cdot, \cdot)$ Euclidean distance. For small n, maximin designs will generally lie on the exterior of S and fill in the interior as n becomes large.

5.4 Hyperbolic cross points

Under the tensor product covariance models, it is possible to approximate and integrate functions with greater accuracy than in the general case. One gets the same rates of convergence as in univariate problems, apart from a multiplicative penalty that is some power of $\log n$. Hyperbolic cross point designs, also known as sparse grids have been shown to achieve optimal rates in these cases. See Ritter [63]. These point sets were first developed by Smolyak [75]. They were used in interpolation by Wahba [82] and Gordon [23] and by Paskov [56] for integration. Chapter 4 of Ritter [63] gives a good description of the construction of these points and lists other references.

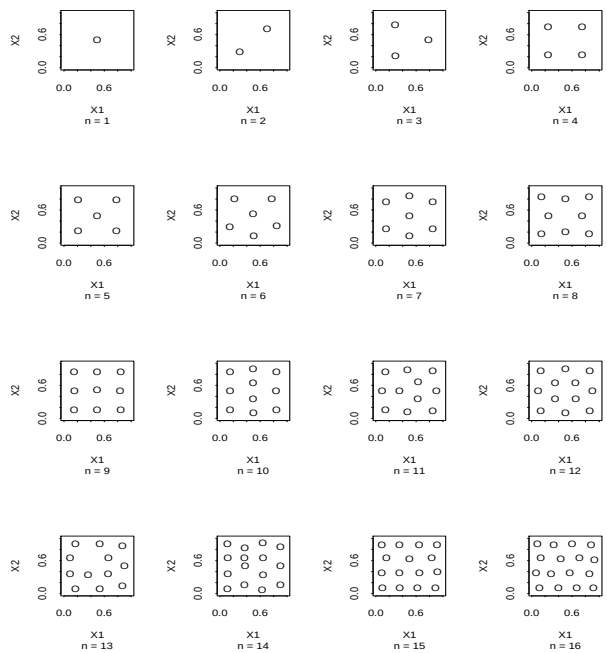

Figure 9: (c) Minimum integrated mean square error designs for $p = 2$, $n = 1$–16, and the Gaussian correlation function with $\theta = (10, 10)$.

6 Frequentist prediction and inference

The frequentist approach to prediction and inference in computer experiments is based on numerical integration. For a scalar function $Y = f(X)$, consider a regression model of the form

$$Y = f(X) \doteq Z(X)\beta \tag{12}$$

where $Z(X)$ is a row vector of predictor functions and β is a vector of parameters. Suitable functions Z might include low order polynomials, trigonometric polynomials wavelets, or some functions specifically geared to the application. Ordinarily $Z(X)$ includes a component that is always equal to 1 in order to introduce an intercept term into Equation (12).

It is unrealistic to expect that the function f will be exactly representable as the finite linear combination given by (12), and it is also unrealistic to expect that the residual will be a random variable with mean zero at every fixed X_0. This is why we only write $f \doteq Z\beta$. There are many ways to define the best value of β, but an especially natural approach is to choose β to minimize the mean squared error of the approximation, with respect to some distribution F on $[0, 1]^p$. Then the optimal value for β is

$$\beta_{LS} = \left(\int Z(X)'Z(X)dF \right)^{-1} \int Z(X)'f(X)dF.$$

So if one can integrate over the domain of X then one can fit regression approximations there.

The quality of the approximation may be assessed globally by the integrated mean squared error

$$\int (Y - Z(X)\beta)^2 dF.$$

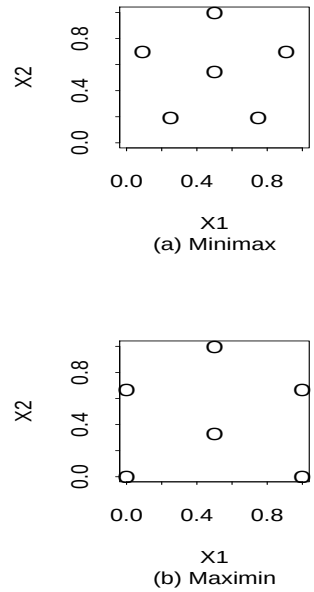

Figure 10: (a) Minimax and (b) maximin designs for $n = 6$ and $p = 2$ with Euclidean distance.

For simplicity we take the distribution F to be uniform on $[0,1]^p$. Also for simplicity the integration schemes to be considered usually estimate $\int g(X)dF$ by

$$\frac{1}{n}\sum_{i=1}^{n} g(x_i)$$

for well chosen points $x_1, \ldots, x_n$. Then β_{LS} may be estimated by linear regression

$$\hat{\beta} = \left(\frac{1}{n}\sum_{i=1}^{n} Z(x_i)'Z(x_i)\right)^{-1} \frac{1}{n}\sum_{i=1}^{n} Z(x_i)'f(x_i),$$

or when the integrals of squares and cross products of $Z's$ are known by

$$\tilde{\beta} = \left(\int Z(X)'Z(X)dF\right)^{-1} \frac{1}{n}\sum_{i=1}^{n} Z(x_i)'f(x_i). \tag{13}$$

Choosing the components of Z to be an orthogonal basis, such as tensor products of orthogonal polynomials, multivariate Fourier series or wavelets, Equation (13) simplifies to

$$\tilde{\beta} = \frac{1}{n}\sum_{i=1}^{n} Z(x_i)'f(x_i) \tag{14}$$

and one can avoid the cost of matrix inversion. The computation required by Equation (14) grows proportionally to nr not n^3, where $r = r(n)$ is the number of regression variables in Z. If $r = O(n)$ then the computations grow as n^2. Then, in the example from Section 3, an hour of function evaluation followed by a minute of algebra would scale into a day of function evaluation followed by 9.6 hours of algebra, instead of the 9.6 days that an n^3 algorithm would require. If the $Z(x_i)$ exhibit some sparsity then it may be possible to reduce the algebra to order n or order $n \log n$.

Thus the idea of turning the function into data and making exploratory plots can be extended to turning the function into data and applying regression techniques. The theoretically simplest technique is to take X_i iid $U[0,1]^p$. Then (X_i, Y_i) are iid pairs with the complication that Y has zero variance given X. The variance matrix of $\tilde{\beta}$ is then

$$\frac{1}{n}\left(\int Z'ZdF\right)^{-1} \mathrm{Var}\left(Z(X)'Y(X)\right)\left(\int Z'ZdF\right)^{-1} \tag{15}$$

and for orthogonal predictors this simplifies further to

$$\frac{1}{n}\mathrm{Var}\left(Z(X)'Y(X)\right). \tag{16}$$

Thus any integration scheme that allows one to estimate variances and covariances of averages of Y times components of Z allows one to estimate the sampling variance matrix of the regression coefficients $\tilde{\beta}$. For iid sampling one can estimate this variance matrix by

$$\frac{1}{n-r-1}\sum_{i=1}^{n}\left(Z(x_i)Y(x_i)-\tilde{\beta}\right)'\left(Z(x_i)Y(x_i)-\tilde{\beta}\right)$$

when the row vector Z comprises an intercept and r additional regression coefficients.

This approach to computer experimentation should improve if more accurate integration techniques are substituted for the iid sampling. Owen [49] investigates the case of Latin hypercube sampling for which a central limit theorem also holds.

Clearly more work is needed to make this method practical. For instance a scheme for deciding how many predictors should be in Z, or otherwise for regularizing $\tilde{\beta}$ is required.

7 Frequentist experimental designs

The frequentist approach proposed in the previous section requires a set of points $x_1, \ldots, x_n$ that are good for numerical integration and also allow one to estimate the sampling variance of the corresponding integrals. These two goals are somewhat at odds. Using an iid sample makes variance estimation easier while more complicated schemes described below improve accuracy but make variance estimation harder.

The more basic goal of getting points x_i into "interesting corners" of the input space, so that important features are likely to be found is usually well served by point sets that are good for numerical integration.

We assume that the region of interest is the unit cube $[0,1]^p$, and that the integrals of interest are with respect to the uniform distribution over this cube. Other regions of interest can usually be reduced to the unit cube and other distributions can be changed to the uniform by a change of variable that can be subsumed into f.

Throughout this section we consider an example with $p = 5$, and plot the design points x_i.

7.1 Grids

Since varying one coordinate at a time can cause one to miss important aspects of f, it is natural to consider instead sampling f on a regular grid. One chooses k different values for each of X^1 through X^p and then runs all k^p combinations. This works well for small values of p, perhaps 2 or 3, but for larger p it becomes completely impractical because the number of runs required grows explosively.

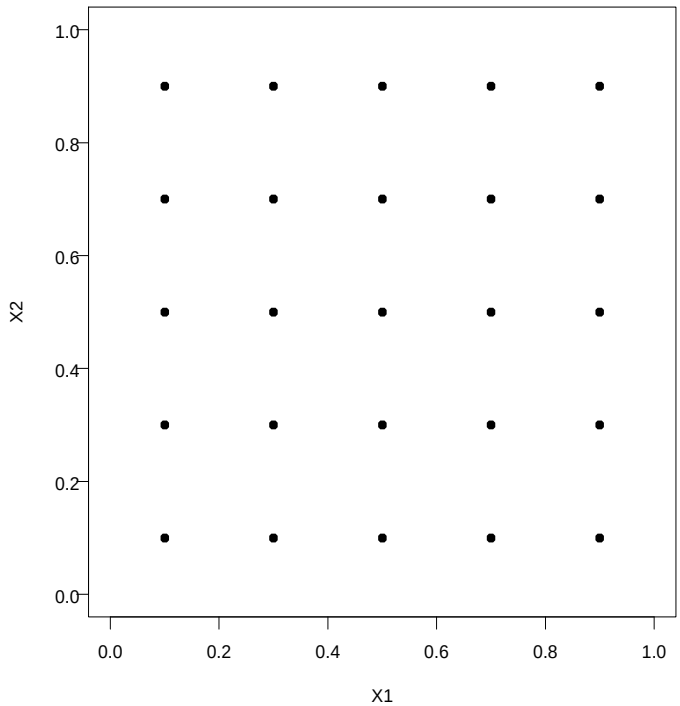

Figure 11: 25 distinct points among 625 points in a 5^5 grid.

Figure 11 shows a projection of $5^5 = 625$ points from a uniform grid in $[0,1]^5$ onto two of the input variables. Notice that with 625 runs, only 25 distinct values appear in the plane, each representing 25 input settings in the other three variables. Only 5 distinct values appear for each of input variable taken singly. In situations where one of the responses Y^k depends very strongly on only one or two of the inputs X^j the grid design leads to much wasteful duplication.

The grid design does not lend itself to variance estimation since averages over the grid are not random. The accuracy of a grid based integral is typically that of a univariate integral based on $k = n^{1/p}$ evaluations. (See Davis and Rabinowitz [15].) For large p this is a severe disadvantage.

7.2 Good lattice points

A significant improvement on grids may be obtained in integration by the method of good lattice points. (See Sloan and Joe [74] and Niederreiter [47] for background and Fang and Wang [18] for applications to statistics.)

For good lattice points

$$x_i^j = \left\{ \frac{h_j(i-1)+0.5}{n} \right\}$$

where $\{z\}$ is z modulo 1, that is, z minus the greatest integer less than or equal to z and h_j are integers with $h_1 = 1$. The points v_i with $v_i^j = ih_j/n$ for integer i form a lattice in R^p. The points x_i are versions of these lattice points confined to the unit cube, and the term "good" refers to a careful choice of n and h_j usually based on number theory.

Figure 12 shows the Fibonacci lattice for $p = 2$ and $n = 34$. For more details see [74]. Here $h_1 = 1$ and $h_2 = 21$. The Fibonacci lattice is only available in 2 dimensions. Appendix A of Fang and Wang [18] lists several other choices for good lattice points, but the smallest value of n there

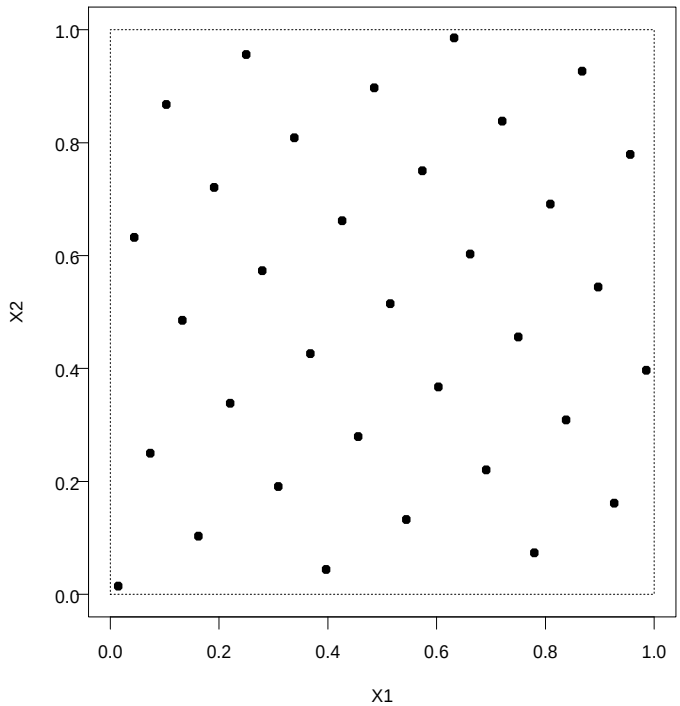

Figure 12: A 34 point Fibonacci lattice in $[0,1]^2$.

for $p = 5$ is 1069. Hickernell [25] discusses greedy algorithms for finding good lattice points with smaller n.

The recent text [74] discusses lattice rules for integration, which generalize the method of good lattice points. Cranley and Patterson [11] consider randomly perturbing the good lattice points by adding, modulo 1, a random vector uniform over $[0,1]^p$ to all the x_i. Taking r such random offsets for each of the n data points gives nr observations with $r-1$ degrees of freedom for estimating variance.

Lattice integration rules can be extraordinarily accurate on smooth periodic integrands and thus an approach to computer experiments based on Cranley and Patterson's method might be expected to work well when both $f(x)$ and $Z(x)$ are smooth and periodic. Bates, Buck, Riccomagno and Wynn [5] have explored the use of lattice rules as designs for computer experiments.

7.3 Latin hypercubes

While good lattice points start by improving the low dimensional projections of grids, Latin hypercube sampling starts with iid samples. A Latin hypercube sample has

$$X_i^j = \frac{\pi^j(i) - U_j^i}{n} \tag{17}$$

where the π^j are independent uniform random permutations of the integers 1 through n, and the U_i^j are independent $U[0,1]$ random variables independent of the π_j.

Latin hypercube sampling was introduced by McKay, Beckman and Conover [38] in what is widely considered to be the first paper on computer experiments. The sample points are stratified

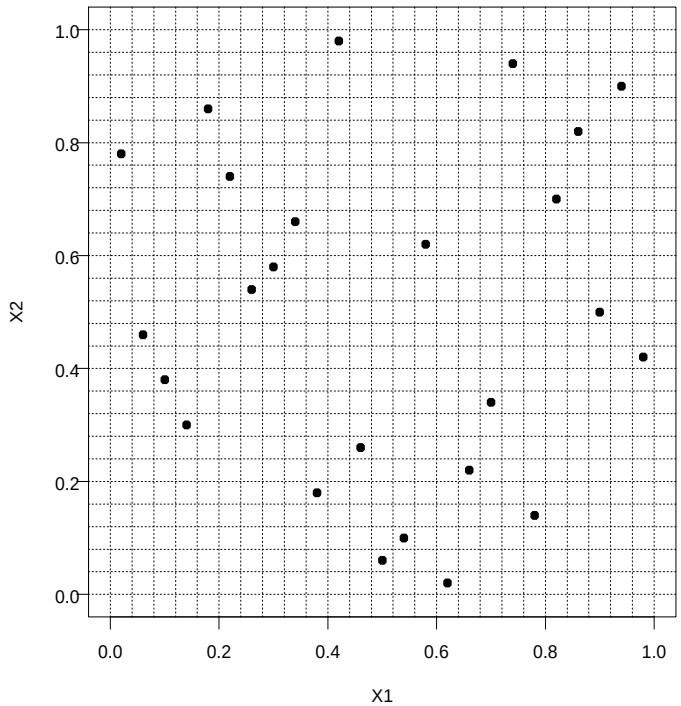

Figure 13: 25 points of a Latin hypercube sample. The range of each input variable may be partitioned into 25 bins of equal width, drawn here with horizontal and vertical dotted lines, and each such bin contains one of the points.

on each of p input axes. A common variant of Latin hypercube sampling has centered points

$$X_i^j = \frac{\pi^j(i) - 0.5}{n}. \tag{18}$$

Point sets of this type were studied by Patterson [59] who called them lattice samples.

Figure 13 shows a projection of 25 points from a (centered) Latin hypercube sample over 5 variables onto two of the coordinate axes. Each input variable gets explored in each of 25 equally spaced bins.

The stratification in Latin hypercube sampling usually reduces the variance of estimated integrals. Stein [76] finds an expression for the variance of a sample mean under Latin hypercube sampling. Assuming that $\int f(X)^2 dF < \infty$ write

$$f(X) = \mu + \sum_{j=1}^{p} \alpha_j(X^j) + e(X) \tag{19}$$

where $\mu = \int f(X)dF$ and $\alpha_j(x) = \int_{X:X^j=x}(f(X) - \mu)dF_{-j}$ in which $dF_{-j} = \prod_{k\neq j} dX^k$ is the uniform distribution over all input variables except the j'th. Equation (19) expresses f as the sum of a grand mean μ, univariate main effects α_j and a residual from additivity $e(X)$.

Stein shows that under Latin hypercube sampling

$$\mathrm{Var}\left(\frac{1}{n}\sum_{i=1}^{n} f(x_i)\right) = \frac{1}{n}\int e(X)^2 dF + o\left(\frac{1}{n}\right) \tag{20}$$

whereas under iid sampling

$$\mathrm{Var}\left(\frac{1}{n}\sum_{i=1}^{n} f(x_i)\right) = \frac{1}{n}\left(\int e(X)^2 dF + \sum_{j=1}^{p}\int \alpha_j(X^j)^2 dF\right) \tag{21}$$

By balancing the univariate margins, Latin hypercube sampling has removed the main effects of the function f from the error variance.

Owen [49] proves a central limit theorem for Latin hypercube sampling of bounded functions and Loh [32] proves a central limit theorem under weaker conditions. For variance estimation in Latin hypercube sampling see [76, 49].

7.4 Better Latin hypercubes

Latin hypercube samples look like random scatter in any bivariate plot, though they are quite regular in each univariate plot. Some effort has been made to find especially good Latin hypercube samples.

One approach has been to find Latin hypercube samples in which the input variables have small correlations. Iman and Conover [27] perturbed Latin hypercube samples in a way that reduces off diagonal correlation. Owen [52] showed that the technique in [27] typically reduces off diagonal correlations by a factor of 3, and presented a method that empirically seemed to reduce the off diagonal correlations by a factor of order n from $O(n^{-1/2})$ to $O(n^{-3/2})$. This removes certain bilinear terms from the lead term in the error. Dandekar [14] found that iterating the method in [27] can lead to large improvements.

Small correlations are desirable but not sufficient, because one can construct centered Latin hypercube samples with zero correlation (unless n is equal to 2 modulo 4) which are nonetheless highly structured. For example the points could be arranged in a diamond shape in the plane, thus missing the center and corners of the input space.

Some researchers have looked for Latin hypercube samples having good properties when considered as designs for Bayesian prediction. Park [57] studies the IMSE criterion and Morris and Mitchell [44] consider entropy.

7.5 Randomized orthogonal arrays

An orthogonal array A is an n by p matrix of integers $0 \le A_i^j \le b-1$. The array has strength $t \le p$ if in every n by t submatrix of A all of the b^t possible rows appear the same number λ of times. Of course $n = \lambda b^t$.

Independently Owen [50, 51] and Tang [79, 80] considered using orthogonal arrays to improve upon Latin hypercube samples.

A randomized orthogonal array [50] has two versions,

$$X_i^j = \frac{\pi_j(A_i^j) + U_i^j}{b} \tag{22}$$

and

$$X_i^j = \frac{\pi_j(A_i^j) + 0.5}{b} \tag{23}$$

just as Latin hypercube sampling has two versions. Indeed Latin hypercube sampling corresponds to strength $t = 1$, with $\lambda = 1$. Here the π_j are independent uniform permutations of $0, \ldots, b-1$. Patterson [59] considered some schemes like the centered version.

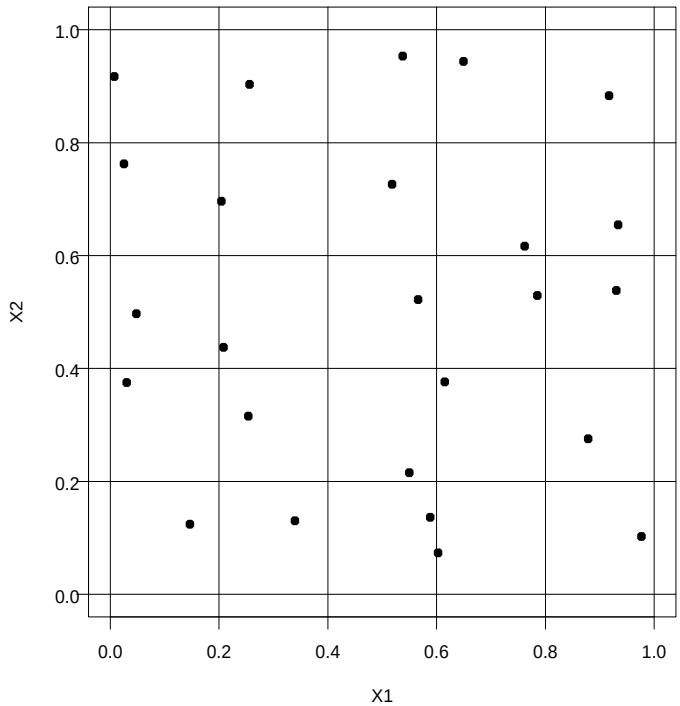

Figure 14: 25 points of a randomly centered randomized orthogonal array. For whichever two (of five) variables that are plotted, there is one point in each reference square.

If one were to plot the points of a randomized orthogonal array in t or fewer of the coordinates, the result would be a regular grid. The points of a randomized orthogonal array of strength 2 appear to be randomly scattered in 3 dimensions.

Figure 14 shows a projection of 25 points from a randomly centered randomized orthogonal array over 5 variables onto two of the coordinate points. Each pair of variables gets explored in each of 25 square bins. The plot for the centered version of a randomized orthogonal array is identical to that for a grid as shown in Figure 11.

The analysis of variance decomposition used above for Latin hypercube sampling can be extended to include interactions among 2 or more factors. See Efron and Stein [17], Owen[50] and Wahba [83] for details. Gu and Wahba [24] describe how to estimate and form confidence intervals for these main effects in noisy data.

Owen [50] shows that main effects and interactions of t or fewer variables do not contribute to the asymptotic variance of a mean over a randomized orthogonal array, and Owen [51] shows that the variance is approximately n^{-1} times the sum of integrals of squares of interactions among more than t inputs.

Tang [80] introduced orthogonal array based Latin hypercube samples. The points of these designs are Latin hypercube samples X_i^j, such that $\lfloor bX_i^j \rfloor$ is an orthogonal array. Here b is an integer and $\lfloor z \rfloor$ is the smallest integer less than or equal to z. Tang [80] shows that for a strength 2 array the main effects and two variable interactions do not contribute to the integration variance.

Figure 15 shows a projection of 25 points from an OA based Latin hypercube sample over 5 variables onto two of the coordinate points. Each variable individually gets explored in each of 25 equal bins and each pair of variables gets explored in each of 25 squares.

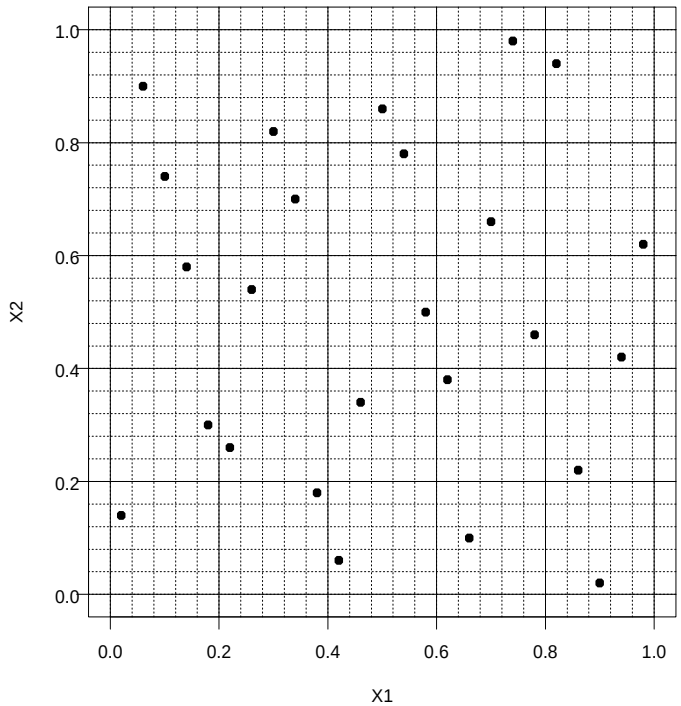

Figure 15: 25 points of an orthogonal array based Latin hypercube sample. For whichever two (of five) variables that are plotted, there is one point in each reference square bounded by solid lines. Each variable is sampled once within each of 25 horizontal or vertical bins.

7.6 Scrambled nets

Orthogonal arrays were developed to balance discrete experimental factors. As seen above they can be embedded into the unit cube and randomized with the result that sampling variance is reduced. But numerical analysts and algebraists have developed some integration techniques directly adapted to balancing in a continuous space. Here we describe (t, m, s)-nets and their randomizations. A full account of (t, m, s)-nets is given by Niederreiter [47]. Their randomization is described by Owen [53, 54].

Let $p = s \geq 1$ and $b \geq 2$ be integers. An elementary subcube in base b is of the form

$$E = \prod_{j=1}^{s} \Big[\frac{c_j}{b^{k_j}}, \frac{c_j + 1}{b^{k_j}}\Big)$$

for integers k_j, c_j with $k_j \geq 0$ and $0 \leq c_j < b^{k_j}$.

Let $m \geq 0$ be an integer. A set of points X_i, $i = 1, \ldots, b^m$ of from $[0, 1)^s$ is a $(0, m, s)$-net in base b if every elementary subcube E in base b of volume b^{-m} has exactly 1 of the points. That is, every cell that "should" have one point of the sequence does have one point of the sequence.

This is a very strong form of equidistribution and by weakening it somewhat, constructions for more values of s and b become available. Let $t \leq m$ be a nonnegative integer. A finite set of b^m points from $[0, 1)^s$ is a (t, m, s)-net in base b if every elementary subcube in base b of volume b^{t-m} contains exactly b^t points of the sequence. Cells that "should" have b^t points do have b^t points, though cells that "should" have 1 point might not.

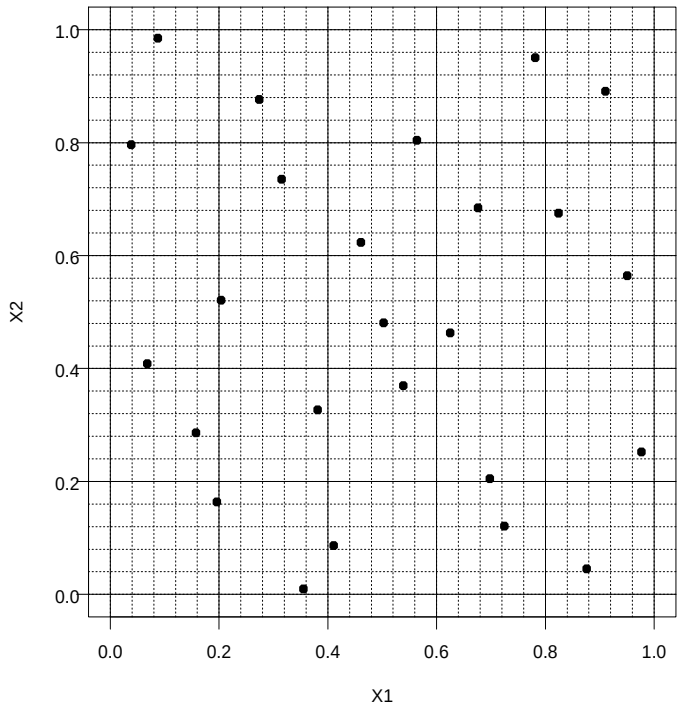

Figure 16: 25 points of a scrambled $(0, 2, 5)$-net in base 5. For whichever two (of five) variables that are plotted, there is one point in each reference square. Each variable is sampled once within each of 25 equal bins.

By common usage the name (t, m, s)-net assumes that the letter s is used to denote the dimension of the input space, though one could speak of (t, m, p)-nets. Another convention to note is that the subcubes are half-open. This makes it convenient to partition the input space into congruent subcubes.

The balance properties of a (t, m, s)-net are greater than those of an orthogonal array. If X_i^j is a (t, m, s)-net in base b then $\lfloor bX_i^j \rfloor$ is an orthogonal array of strength $\min\{s, m - t\}$. But the net also has balance properties when rounded to different powers of b on all axes, so long as the powers sum to no more than $m - t$. Thus the net combines aspects of orthogonal arrays and multi-level orthogonal arrays all in one point set.

In the case of a $(0, 4, 5)$-net in base 5, one has 625 points in $[0, 1)^5$ and one can count that there are 43750 elementary subcubes of volume $1/625$ of varying aspect ratios each of which has one of the 625 points.

For $t \geq 0$, an infinite sequence $(X_i)_{i \geq 1}$ of points from $[0, 1)^s$ is a (t, s)-sequence in base b if for all $k \geq 0$ and $m \geq t$ the finite sequence $(X_i)_{i=kb^m+1}^{(k+1)b^m}$ is a (t, m, s)-net in base b.

The advantage of a (t, s)-sequence is that if one finds that the first b^m points are not sufficient for an integration problem, one can find another b^m points that also form a (t, m, s)-net and tend to fill in places not occupied by the first set. If one continues to the point of having b such (t, m, s)-nets, then the complete set of points comprises a $(t, m + 1, s)$-net.

The theory of (t, m, s)-nets and (t, s)-sequences is given in [47]. A famous result of the theory is that integration over a (t, m, s) net can attain an accuracy of order $O(\log(n)^{s-1}/n)$ while restricting to (t, s) sequences raises this slightly to $O(\log(n)^s/n)$. These results require that the integrand be

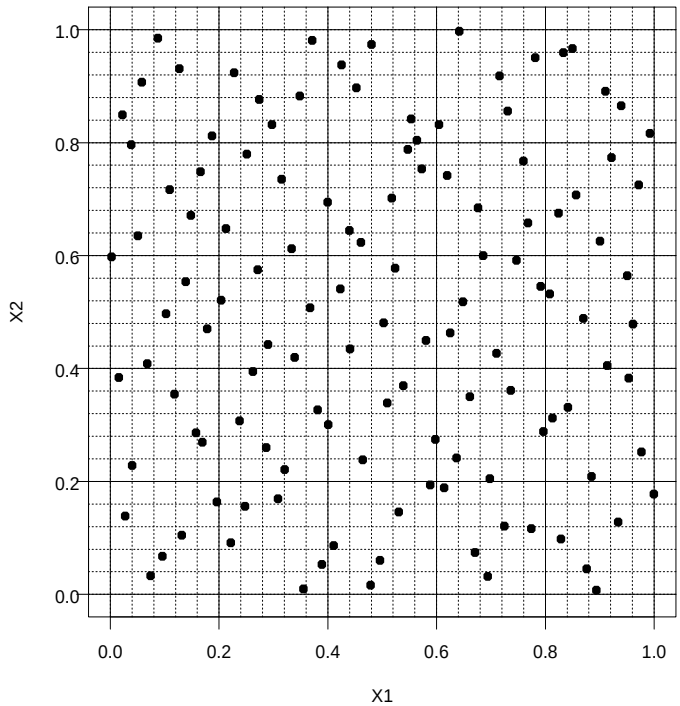

Figure 17: The 125 points of a scrambled $(0,3,5)$-net in base 5. For whichever two (of five) variables that are plotted, the result is a 5 by 5 grid of 5 point Latin hypercube samples. Each variable is sampled once within each of 125 equal bins. Each triple of variables can be partitioned into 125 congruent cubes, each of which has one point.

of bounded variation in the sense of Hardy and Krause. For large s, it takes unrealistically large n for these rates to be clearly better than $n^{-1/2}$ but in examples they seem to outperform simple Monte Carlo.

The construction of (t,m,s)-nets and (t,s)-sequences is also described in [47]. Here we remark that for prime numbers s a construction by Faure [19] gives $(0,s)$-nets in base s and Niederreiter extended the method to prime powers s. (See [47].) Thus one can choose b to be the smallest prime power greater than or equal to s and use the first s variables of the corresponding $(0,b)$-sequence in base b.

Owen [53] describes a scheme to randomize (t,m,s)-nets and (t,s)-sequences. The points are written in a base b expansion and certain random permutations are applied to the coefficients in the expansion. The result is to make each permuted X_i uniformly distributed over $[0,1)^s$ while preserving the (t,m,s)-net or (t,s)-sequence structure of the ensemble of X_i. Thus the sample estimate $n^{-1}\sum_{i=1}^n f(X_i)$ is unbiased for $\int f(X)dF$ and the variance of it may be estimated by replication. On some test integrands in [53] the randomized nets outperformed their unrandomized counterparts. It appears that the unscrambled nets have considerable structure, stemming from the algebra underlying them, and that this structure is a liability in integration.

Figure 16 shows the 25 points of a scrambled $(0,2,5)$-net in base 5 projected onto two of the five input coordinates. These points are the initial 25 points of a $(0,5)$-sequence in base 5. This design has the equidistribution properties of an OA-based Latin hypercube sample. Moreover every consecutive 25 points in the sequence $X_{25a+1}, X_{25a+2}, \ldots, X_{25(a+1)}$ has these equidistribution properties. The first 125 points, shown in Figure 17 have still more equidistribution properties:

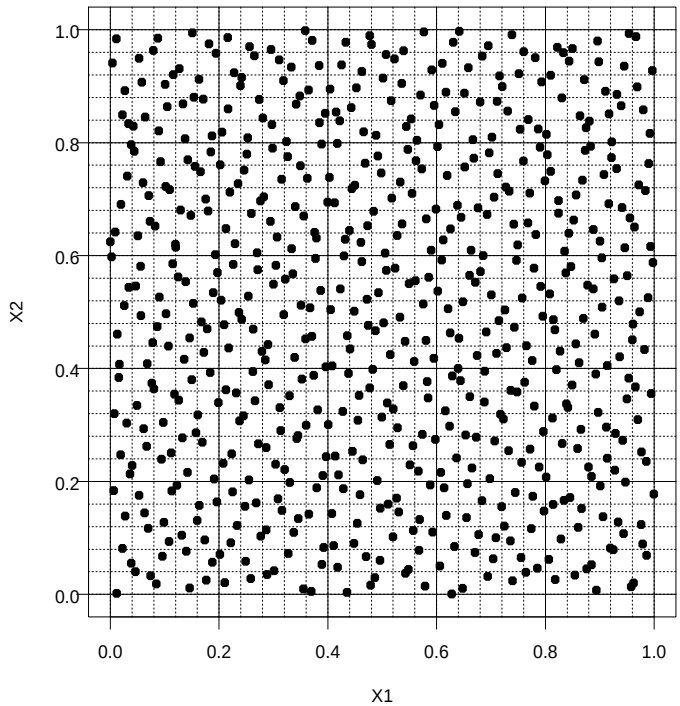

Figure 18: The 625 points of a scrambled $(0, 4, 5)$-net in base 5. For whichever two (of five) variables that are plotted, the square can be divided into 625 squares of side $1/25$ or into 625 rectangles of side $1/5$ by $1/125$ or into 625 rectangles of side $1/125$ by $1/5$ and each such rectangle has one of the points. Each variable is sampled once within each of 625 equal bins. Each triple of variables can be partitioned into 625 hyperrectangles in three different ways and each such hyperrectangle has one of the points. Each quadruple of variables can be partitioned into 625 congruent hypercubes of side $1/5$, each of which has one point.

any triple of the input variables can be split into 125 subcubes each with one of the X_i, in any pair of variables the points appear as a 5 by 5 grid of 5 point Latin hypercube samples and each individual input variable can be split into 125 cells each having one point. The first 625 points, are shown in Figure 18.

Owen [54] finds a variance formula for means over randomized (t, m, s)-nets and (t, s)-sequences. The formula involves a wavelet-like anova combining nested terms on each coordinate, all crossed against each other. It turns out that for any square integrable integrand, the resulting variance is $o(n^{-1})$ and it therefore beats any of the usual variance reduction techniques, which typically only reduce the asymptotic coefficient of n^{-1}.

For smooth integrands with $s = 1$, the variance is in fact $O(n^{-3})$ and in the general case Owen [55] shows that the variance is $O(n^{-3}(\log n)^{s-1})$.

8 Selected Applications

One of the largest fields using and developing deterministic simulators is in the designing and manufacturing of VLSI circuits. Alvarez et al [1] describe the use of SUPREM-III [26] and SEDAN-II [91] in designing BIMOS devices for manufacturability. Aoki et al [81], use CADDETH a two

dimensional device simulator, for optimizing devices and for accurate prediction of device sensitivities. Sharifzadeh et al [71] use SUPREME-III and PISCES-II [61] to compute CMOS device characteristics as a function of the designable technology parameters. Nasif, Strojwas and Director [46] describe the use of FABRICS-II to estimate circuit delay times in integrated circuits.

The input variables for the above work are generally device sizes, metal concentrations, implant doses and gate oxide temperatures. The multiple responses are threshold voltages, subthreshold slopes, saturation currents and linear transconductance although the output variables of concern depend on the technology under investigation. The engineers use the physical/numerical simulators to assist them in optimizing process, device, and circuit design before the costly step of building prototype devices. They are also concerned with minimizing transmitted variability as this can significantly reduce the performance of the devices and hence reduce yield. For example, Welch et al [87], Currin et al [13] and Sacks et al [69] discuss the use of simulators to investigate the effect of transistor dimensions on the asynchronization of two clocks. They want to find the combination of transistor widths that produce zero clock skews with very small transmitted variability due to uncontrollable manufacturing variability in the transistors.

TIMS, a simulator developed by T. Osswald and C. L. Tucker III, helps in optimizing a compression mold filling process for manufacturing automobiles [10]. In this process a sheet of molding compound is cut and placed in a heated mold. The mold is slowly closed and a constant force is applied during the curing reaction. The controlling variables of the process are the geometry and thickness of the part, the compound viscosity, shape and location within the charge, and the mold closing speed. The simulator then predicts the position of the flow front as a function of time.

Miller and Frenklach [39] discuss the use of computers to solve systems of differential equations describing chemical kinetic models. In their work, the inputs to the simulator are vectors of possibly unknown combustion rate constants and the outputs are induction-delay times and concentrations of chemical species at specified reaction times. The objectives of their investigations are to find values of the rate constants that agree with experimental data and to find the most important rate constant to the process. Sacks et al [68] explore some of the design issues and applications to this field.

TWOLAYER, a thermal energy storage model developed by Alan Solomon and his colleagues at the Oak Ridge National Laboratory, simulates heat transfer through a wall containing two layers of different phase change material. Currin et al utilize TWOLAYER in a computer experiment. The inputs into TWOLAYER are the layers dimensions, the thermal properties of the materials and the characteristics of the heat source. The object of interest was finding the configuration of the input variables that produce the highest value of a heat storage utility index.

FOAM [4] models the transport of polycyclic aromatic hydrocarbon spills in streams using structure activity relationships. Bartell et al [3] modified this model to predict the fate of anthracene when introduced into ponds. This model tracts the "evaporation and dissolution of anthracene from a surface slick of synthetic oil, volatilization and photolytic degradation of dissolved anthracene, sorption to suspended particulate matter and sediments and accumulation by pond biota" [3]. They used Monte Carlo error analyses to assess the effect of the uncertainty in model parameters on their results.

References

[1] Alvarez, A.R., Abdi, B.L., Young, D.L., Weed, H.D., Teplik, J. and Herald, E. Application of statistical design and response surface methods to computer-aided VLSI device design. *IEEE Trans. Computer Aided Design*, 7(2):271–288, 1988.

[2] Aoki, Y., Masuda, H., Shimada, S. and Sato, S. A new design-centering methodology for VLSI device development. *IEEE Trans. Computer Aided Design*, 6(3):452–461, 1987.

[3] Bartell, S.M., Gardner, R.H., O'Neill, R.V. and Giddings, J.M. Error analysis of predicted fate of anthracene in a simulated pond. *Environmental Toxicology and Chemistry*, 2:19–28, 1983.

[4] Bartell, S.M., Landrum, J.P., Giesy, J.P. and Leversee, G.J. Simulated transport of polycyclic aromatic hydrocarbons in artificial streams. In W.J. Mitch, R. W. Bosserman, and J.M. Klopatek, editors, *Energy and Ecological Modelling*, pages 133–143. Elsevier, New York, 1981.

[5] Bates, R. A., Buck, R. J., Riccomagno, E., Wynn, H. P. Experimental Design and Observation for Large Systems (with discussion). *J. Royal Stats. Soc. B*, **58**, No. 1, pp. 77–94, 1996.

[6] Borth, D.M. A total entropy criterion for the dual problem of model discrimination and parameter estimation. *Journal of the Royal Statistical Society, Series B*, 37:77–87, 1975.

[7] Box, G.E.P. and Draper, N.R. A basis for the selection of a response surface design. *Journal of the American Statistical Association*, 54:622–654, 1959.

[8] Box, G.E.P. and Draper, N.R. The choice of a second order rotatable design. *Biometrika*, 50:335–352, 1963.

[9] Box, G.E.P. and Hill, W.J. Discrimination among mechanistic models. *Technometrics*, 9:57–70, 1967.

[10] Church, A., Mitchell, T. and Fleming, D. Computer experiments to optimize a compression mold filling process. Talk given at the Workshop on Design for Computer Experiments in Oak Ridge, TN, November 1988.

[11] Cranley, R. and Patterson, T.N.L. Randomization of Number Theoretic Methods for Multiple Integration. *SIAM J. Numer. Anal.*, 23:904–14, 1976.

[12] Cressie, N. Kriging nonstationary data. *Journal of the American Statistical Association*, 81:625–34, 1986.

[13] Currin, C., Mitchell, M., Morris, M. and Ylvisaker, D. Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments. *Journal of the American Statistical Association*, 86:953–63, 1991.

[14] Dandekar, R. Performance improvement of restricted pairing algorithm for Latin hypercube sampling Draft Report, Energy Information Administration, U.S. D.O.E., 1993.

[15] Davis, P.J. and Rabinowitz, P. *Methods of Numerical Integration,* Second edition Academic Press, San Diego, 1984.

[16] Diaconis, P. Bayesian numerical analysis In S. S. Gupta and J. O. Berger, editors, *Statistical Decision Theory and Related Topics IV: Vol. 1*, pages 163–176, Springer-Verlag,New York, 1988.

[17] Efron, B. and Stein, C. The jackknife estimate of variance. *Ann. Statist.* 9, 586–596, 1981.

[18] Fang, K.T. and Wang, Y. *Number-theoretic Methods in Statistics* Chapman and Hall, London, 1994.

[19] Faure, H. Discrépances des suites associées à un système de numération (en dimension s) *Acta Arithmetica*, 41:337–351, 1982.

[20] Friedman, J.H. Multivariate Adaptive Regression Splines (with discussion) *Annals of Statistics*, 19:1–67, 1991.

[21] Gill, P.E., Murray, W., Saunders, M.A. and Wright, M.H. User's guide for npsol (version 4.0): a fortran package for nonlinear programming. SOL 86-2, Stanford Optimization Laboratory, Dept. of Operations Research, Stanford University, California, 94305, January 1986.

[22] Gill, P.E., Murray, W. and Wright, M.H. *Practical Optimization.* Academic Press, London, 1981.

[23] Gordon, W.J. Blending function methods of bivariate and multivariate interpolation and approximation. *SIAM J. Numer. Anal.* **8**, 158–177, 1971.

[24] Gu, C. and Wahba, G. Smoothing spline ANOVA with component-wise Bayesian "confidence intervals". *J. Comp. Graph. Stat.*, **2**, 97–117, 1993.

[25] Hickernell, F.J. Quadrature Error Bounds with Applications to Lattice Rules *SIAM J. Numer. Anal.*, 33, 1996.

[26] Ho, S.P., Hansen, S.E. and Fahey, P.M. Suprem III - a program for integrated circuit process modeling and simulation. TR-SEL84 1, Stanford Electronics Laboratories, 1984.

[27] Iman, R.L. and Conover, W.J. A distributon-free approach to inducing rank correlation among input variables *Commun. Stat* B11(3):311–334, 1982.

[28] Johnson, M.E., Moore, L.M. and Ylvisaker, D. Minimax and maximin distance designs. *Journal of Statistical Planning and Inference*, 26:131–148, 1990.

[29] Journel, A.G. and Huijbregts, C.J. *Mining Geostatistics.* Academic Press, London, 1978.

[30] Koehler, J.R. *Design and Estimation Issues in Computer Experiments.* Dissertation, Dept. of Statistics, Stanford University, 1990.

[31] Lindley, D.V. On a measure of the information provided by an experiment. *Annuals of Mathematical Statistics*, 27:986–1005, 1956.

[32] Loh, W.-L. On Latin hypercube sampling, Technical Report No. 93-52, Dept. of Statistics, Purdue University, 1993.

[33] Loh, W.-L. A combinatorial central limit theorem for randomized orthogonal array sampling designs, Technical Report No. 94-4, Dept. of Statistics, Purdue University, 1994.

[34] Mardia, K.V. and Marshall, R.J. Maximum likelihood estimation of models for residual covariance in spatial regression. *Biometrika*, 71(1):135–146, 1984.

[35] Matérn, B. Method of estimating the accuracy of line and sample plot surveys. *Medd. Skogsforskn Inst.*, 36(1), 1947.

[36] Matheron, G. Principles of geostatistics. *Econom. Geol.*, 58:1246–1266, 1963.

[37] McKay, M. *Evaluating prediction uncertainty.* Report NUREG/CR-6311, Los Alamos National Laboratory, 1995.

[38] McKay, M., Beckman, R. Conover, W. A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics*, 21(2):239–245, 1979.

[39] Miller, D. and Frenklach, M. Sensitivity analysis and parameter estimation in dynamic modeling of chemical kinetics. *International Journal of Chemical Kinetics*, 15:677–696, 1983.

[40] Mitchell, T.J. An algorithm for the construction of 'D-optimal' experimental designs. *Technometrics*, 16:203–210, 1974.

[41] Mitchell, T., Morris, M. and Ylvisaker, D. Existence of smoothed stationary processes on an interval. *Stochastic Processes and Their Applications*, 35:109–119, 1990.

[42] Mitchell, T., Morris, M. and Ylvisaker, D. Two-Level Fractional Factorials and Bayesian Prediction. *Statistica Sinica*, 5:559–573, 1995.

[43] Mitchell, T.J., and Scott, D.S. A computer program for the design of group testing experiments. *Communications in Statistics - Theory and methods*, 16:2943–2955, 1987.

[44] Morris, M.D. and Mitchell, T.J. Exploratory Designs for Computational Experiments *Journal of Statistical Planning and Inference*, 43:381-402, 1995.

[45] Morris, M.D., Mitchell, T.J. and Ylvisaker, D. Bayesian design and analysis of computer experiments: Use of derivative in surface prediction. *Technometrics*, 35(3):243–255, 93.

[46] Nassif, S.R., Strojwas, A.J. and Director, S.W. FABRICS II: a statistically based IC fabrication process simulator. *IEEE Trans. Computer-Aided Design*, 3:40–46, 1984.

[47] Niederreiter, H. *Random Number Generation and Quasi-Monte Carlo Methods* SIAM, Philadelphia, 1992.

[48] O'Hagan, A. Comment: Design and analysis of computer experiments. *Statistical Science*, 4(4), 1989.

[49] Owen, A.B. A central limit theorem for Latin hypercube sampling. *Journal of the Royal Statistical Society*, B 54, 541–551, 1992.

[50] Owen, A. B. Orthogonal arrays for computer experiments, integration and visualization *Statist. Sinica* 2 pp. 439–452, 1992.

[51] Owen, A. B. Lattice sampling revisited: Monte Carlo variance of means over randomized orthogonal arrays. *Ann. Statist.* 22, pp. 930–945, 1994.

[52] Owen, A. B. Controlling Correlations in Latin Hypercube Samples. *Journal of the American Statistical Association* 89, 1517–1522, 1994.

[53] Owen, A. B. Randomly permuted (t, m, s)-Nets and (t, s)-sequences, In H. Niederreiter and P.J.-S. Shiue, editors, *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, pp. 299–317, Springer, New York, 1995.

[54] Owen, A. B. Monte Carlo Variance of Scrambled Net Quadrature. *SIAM J. Numer. Anal.*, To appear, 1996.

[55] Owen, A. B. Scrambled net variance for integrals of smooth functions. Technical Report number 493, Department of Statistics, Stanford University, 1996.

[56] Paskov, S. H. Average case complexity of multivariate integration for smooth functions. *J. Complexity* **9**, 291–312, 1993.

[57] Park, J.-S. Optimal Latin-hypercube designs for computer experiments *Journal of Statistical Planning and Inference*, 39, 95–111, 1994.

[58] Parzen, A.B. *Stochastic Processes.* Holden-Day, San Francisco, 1962.

[59] Patterson, H.D. The Errors of Lattice Sampling. *Journal of the Royal Statistical Society*, B 16, 140–149, 1954.

[60] Phadke, M. *Quality engineering using robust design* Prentice Hall, Englewood Cliffs NJ, 1988.

[61] Pinto, M.R., Rafferty, C.S. and Dutton, R.W. PISCES-II–posson and continuity equation solver. DAGG-29-83-k 0125, Stanford Electron. Lab., 1984.

[62] Ripley, B. *Spatial Statistics.* Wiley, New York, 1981.

[63] Ritter, K. *Average Case Analysis of Numerical Problems* Dissertation, University of Erlangen, 1995.

[64] Ritter, K., Wasilkowski, G. and Wozniakowski H. On multivariate integration for stochastic processes. In *Numerical Integration*, H. Brass and G. Hammerlin, eds., ISNM **112**, pp. 331–347, Birkhauser Verlag, Basel, 1993.

[65] Ritter, K., Wasilkowski, G. and Wozniakowski H. Multivariate integration and approximation for random fields satisfying Sacks-Ylvisaker conditions. *Ann. Appl. Prob.* **5**, 518–540, 1995.

[66] Roosen, C.B. *Visualization and Exploration of High-Dimensional Functions using the Functional ANOVA Decomposition.* Dissertation, Dept. of Statistics, Stanford University, 1995.

[67] Sacks, J. and Schiller, S. Spatial designs. In S. S. Gupta and J. O. Berger, editors, *Statistical Decision Theory and Related Topics IV: Vol. 2*, pages 385–399, Springer-Verlag, New York, 1988.

[68] Sacks, J., Schiller, S.B. and Welch, W.J. Designs for computer experiments. *Technometrics*, 31(1), 1989.

[69] Sacks, J., Welch, W.J., Mitchell, T.J. and Wynn, H.P. Design and analysis of computer experiments. *Statistical Science*, 4(4), 1989.

[70] Shannon, C.E. A mathematical theory of of communication. *Bell System Technical Journal*, 27:379–423,623–656, 1948.

[71] Sharifzadeh, S., Koehler, J.R., Owen, A.B. and Shott, J.D. Using simulators to model transmitted variability in IC manufacturing. *IEEE Trans. Semicond. Manufact.*, 2(3):82–93, 1989.

[72] Shewry, M.C. and Wynn, H.P. Maximum entropy sampling. *Journal of Applied Statistics*, 14:165–170, 1987.

[73] Shewry, M.C. and Wynn, H.P. Maximum entropy sampling and simulation codes. In *Proceedings of the 12th World Congress on Scientific Computation*, volume 2, pages 517–519. IMAC88, 1988.

[74] Sloan, I.H. and Joe, S. *Lattice Methods for Multiple Integration,* Oxford Science Publications, Oxford, 1994.

[75] Smolyak, S.A. Quadrature and interpolation formulas for tensor products of certain classes of functions. *Soviet Math. Dokl.* **4**, 240–243, 1963.

[76] Stein, M.L. Large sample properties of simulations using Latin hypercube sampling. *Technometrics*, 29(2):143–151, 1987.

[77] Stein, M.L. Comment:design and analysis of computer experiments. *Statistical Science*, 4(4):432–433, 1989.

[78] Steinberg, D.M. Model robust response surface designs: Scaling two-level factorials. *Biometrika*, 72:513–26, 1985.

[79] Tang, B. *Latin Hypercubes and Supersaturated Designs* Dissertation, Dept. of Statistics and Actuarial Science, University of Waterloo, 1992.

[80] Tang, B. Orthogonal array-based Latin hypercubes *J. Amer. Statist. Assn.* 88 1392–1397, 1993.

[81] Toyabe, T., Masuda, H., Aoki, Y., Shukuri, H. and Hagiwara, T. Three-dimensional device simulator CADDETH with highly convergent matrix solution algorithms. *IEEE Trans. Electron Device*, ED-32:2038–2044, 1985.

[82] Wahba, G. Interpolating surfaces: High order convergence rates and their associated designs, with applications to X-ray image reconstruction. University of Wisconsin-Madison, Statistics Department TR 523, May 1978.

[83] Wahba, G. *Spline Models for Observational Data.* CBMS-NSF Regional Conference Series in Applied Mathematics, v. 59 SIAM, 1990.

[84] Wasilkowski, G. Integration and approximation of multivariate functions: average case complexity with Wiener measure, *Bull. Amer. Math. Soc. (N.S.)*, **28**, 308–314, 1993. Full version *J. Approx. Theory*, **77**, 212–227, 1994.

[85] Wozniakowski H. Average case complexity of multivariate integration. *Bull. Amer. Math. Soc. (N.S.)*, **24**, 185–194, 1991.

[86] Welch, W.J. A mean squared error criterion for the design of experiments. *Biometrika*, 70(1):201–13, 1983.

[87] Welch, W. Yu, T., Kang, and Sacks, J. Computer experiments for quality control by parameter design. *Journal of Quality Technology*, 22:15–22, 1990.

[88] Yaglom, A.M. *Correlation Theory of Stationary and Related Random Functions*, volume 1. Springer, New York, 1987.

[89] Ylvisaker, D. Designs on random fields. In J. N. Srivastava, editor, *A Survey of Statistical Design and Linear Models*, pages 593–607, Amsterdam, 1975. North-Holland.

[90] Young, A.S. A Bayesian approach to prediction using polynomials. *Biometrika*, 64:309–317, 1977.

[91] Yu, Z., Chang, G.G.Y. and Dutton, R.W. Supplementary report on sedan II. TR-G201 12, Stanford Electronics Laboratories, 1982.