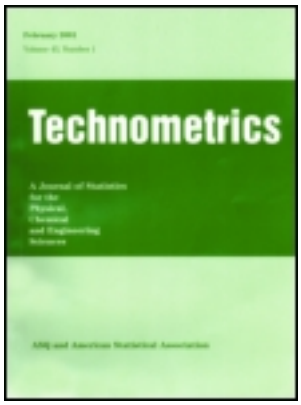

Technometrics
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/utch20

Choosing the Sample Size of a Computer Experiment: A Practical Guide

Jason L. Loeppky, Jerome Sacks and William J. Welch
Mathematics, Statistics, and Physics, University of British Columbia, Okanagan, Kelowna, BC V1V 1V7, Canada
National Institute of Statistical Sciences, Research Triangle Park, NC 27709
Department of Statistics, University of British Columbia, Vancouver, BC V6T 1Z2, Canada
Version of record first published: 01 Jan 2012.

To cite this article: Jason L. Loeppky, Jerome Sacks and William J. Welch (2009): Choosing the Sample Size of a Computer Experiment: A Practical Guide, Technometrics, 51:4, 366-376

To link to this article: http://dx.doi.org/10.1198/TECH.2009.08040

Choosing the Sample Size of a Computer Experiment: A Practical Guide

Jason L. LOEPPKY

Mathematics, Statistics, and Physics
University of British Columbia, Okanagan
Kelowna, BC V1V 1V7
Canada
(*jason@stat.ubc.ca*)

Jerome SACKS

National Institute of Statistical Sciences
Research Triangle Park, NC 27709
(*sacks@niss.org*)

William J. WELCH

Department of Statistics
University of British Columbia
Vancouver, BC V6T 1Z2
Canada
(*will@stat.ubc.ca*)

We provide reasons and evidence supporting the informal rule that the number of runs for an effective initial computer experiment should be about 10 times the input dimension. Our arguments quantify two key characteristics of computer codes that affect the sample size required for a desired level of accuracy when approximating the code via a Gaussian process (GP). The first characteristic is the total sensitivity of a code output variable to all input variables; the second corresponds to the way this total sensitivity is distributed across the input variables, specifically the possible presence of a few prominent input factors and many impotent ones (i.e., effect sparsity). Both measures relate directly to the correlation structure in the GP approximation of the code. In this way, the article moves toward a more formal treatment of sample size for a computer experiment. The evidence supporting these arguments stems primarily from a simulation study and via specific codes modeling climate and ligand activation of G-protein.

KEY WORDS: Curse of dimensionality; Effect sparsity; Gaussian process; Latin hypercube design; Prediction accuracy; Random function.

1. INTRODUCTION

Choosing the sample size of a deterministic computer experiment is important but lacks formal guidance. The reasons for this range from inadequate prior information about the process under study to inadequate results for making necessary calculations. Because physical experimentation is absent, the constraints on experimental size are typically from the time it takes to make runs of the code. Such constraints often are vague and flexible. Where budget issues prevail ("you get this much computer time to make your runs"), the choice of sample size, n, is taken out of our hands. Nevertheless, it is useful to have practical guidance in choosing n and to know whether the selected n is adequate to achieve stated goals.

Along with advice on the choice of n for a specific experiment, we consider more general questions—in particular: What is the role of dimensionality, d, of the input space? If the curse of dimensionality applies, then high-dimensional problems might require huge, even intractable, sample sizes for good prediction accuracy. On the other hand, if the total sensitivity of the function to all input variables is kept fixed, with this sensitivity just spread over more input variables, then dimensionality might conceivably have a limited effect on accuracy, as in Monte Carlo integration. In this article, how total sensitivity grows with d and how this sensitivity is spread across the dimensions are keys to understanding prediction accuracy and thus sample size. Indeed, the article is really about defining the properties of functions that arise in practice, from which simple rules about sample size follow *for that class of problems*.

Chapman et al. (1994) and Jones, Schonlau, and Welch (1998) introduced and used the $n = 10d$ rule of thumb that we study in this article. Otherwise, little has been written on sample size in the context of computer experiments. Sahama and Diamond (2001) produced a plot similar to that in Figure 1 (Section 3) to assess the effect of sample size on prediction error in a four-dimensional example. Their plot shows that 40 runs would provide reasonable accuracy and thus is consistent with the $n = 10d$ rule. A theoretical analysis by Chen (1996) showed that for $d = 1$, the order of the maximum mean squared prediction error is n^{-n} for very smooth output functions and equally spaced designs. For $d > 1$, Chen (1996) also produced results for product designs, indicating that in low dimensions ($d \leq 3$), there is still rapid decline in error rates with increasing n. Many applications have $d > 3$, however, and dense product designs are impractical for high-dimensional problems. Because the aim of this article is to provide a better theoretical underpinning for the empirical $n = 10d$ rule in practical settings, d ranges from 4 to 20 in our examples and simulations.

TECHNOMETRICS, NOVEMBER 2009, VOL. 51, NO. 4
DOI 10.1198/TECH.2009.08040

Following the path taken since 1989 (Sacks et al. 1989; Currin et al. 1991), we approximate the computer output using a Gaussian process (GP) constructed from a set of code runs. In general, characterization of the factors affecting approximation accuracy, and hence sample size, requires precise formulation of the goals of the experiment. Such a formulation is often elusive, however. Accordingly, we restrict our attention to the objective of approximating the code on the basis of sample runs and on the the question of how many runs are needed to obtain adequate prediction accuracy at untried inputs. The choice of a measure of accuracy is open to subjective judgment. The measures that we use are given in (5) and (6). Other issues, such as optimization of a target criterion, could raise other considerations, especially those surrounding fully sequential experimentation.

Along with the number of runs, the particular experimental design must be chosen. Considerable experience built up over a number of applications leads us to restrict our attention to designs that are space-filling. Liu (2005) studied the joint effect of sample size and design choice for a number of examples and concluded that sample size is more important than design choice. Throughout, we use Latin hypercube designs (LHDs), introduced by McKay, Beckman, and Conover (1979). The simplest form of LHDs are convenient for the theoretical analysis in Section 4. Our empirical studies use LHDs optimized via a maximin distance criterion averaged over all two-dimensional projections (Welch et al. 1996). Simpler-to-construct zero-correlation LHDs (Gough and Welch 1994; Owen 1994) also could be used.

While there are many issues associated with determining sample size, we focus on the following questions:

- Is $n = 10d$ a good rule? What are the limitations of such a rule?
- How does accuracy increase with n? When are feasible sample sizes available?
- What are the affects of criteria on sample size determination?
- What should be done when a criterion for accuracy is not met?

To answer these questions, we aim to quantify the complexity of a GP model. Of prime concern is the total sensitivity of an output variable to all of the input variables. How this sensitivity is distributed across the input variables is also important, especially for large d. These characteristics guide a simulation study. Our key conclusion is that the empirically based recommendation of $n = 10d$ runs will provide reasonable prediction accuracy for "tractable" functions and are sufficient to diagnose more difficult problems. For the latter, we show that the rate of improvement in accuracy with n is poor as well. Thus, unless a huge sample size is available, more than $n = 10d$ runs would be wasteful.

The article is organized as follows. In Section 2 we review the GP model and specify the measures of accuracy that we use. In Section 3 we explore an example to illustrate the issues. We quantify the complexity of a GP in Section 4; the characteristics developed guide the simulation study reported in Section 5. In Section 6 we relate the simulation study to various examples. Finally, in Section 7 we summarize our conclusions and comment on open issues.

2. THE GAUSSIAN PROCESS MODEL

Overcoming computational demands of complex computer codes has led, since 1989 (Sacks, Schiller, and Welch 1989; Sacks et al. 1989; Currin et al. 1991; O'Hagan 1992), to strategies that rely on computationally efficient statistical prediction (approximation, emulation) of the code. Following these pathways, we place a homogeneous GP prior on the possible output functions, leading to a predictor given by the posterior mean conditional on the data from the computer experiment. Although output from a computer model often is multivariate, we restrict our attention to scalar output here. Results for scalar output can be carried over via principal components analysis or wavelet decompositions of functional output, as described by Higdon et al. (2008) and Bayarri et al. (2007).

The computer code output is denoted by $y(\mathbf{x})$, where the vector of input variables, $\mathbf{x} = (x_1, \ldots, x_d)$, is in a d-dimensional unit cube. As long as the input space is rectangular, there is no loss of generality. The GP model places a prior on the class of possible $y(\mathbf{x})$. Let $Y(\mathbf{x})$ denote the random function whose distribution is determined by the prior. Suppose that $Y(\mathbf{x}) = \mu + Z(\mathbf{x})$, where μ is a mean parameter and $Z(\mathbf{x})$ is a Gaussian stochastic process with mean 0, constant variance σ^2, and correlation function given by

$$R(\mathbf{x}, \mathbf{x}') = \exp(-h(\mathbf{x}, \mathbf{x}')), \tag{1}$$

where

$$h(\mathbf{x}, \mathbf{x}') = \sum_{j=1}^{d} \theta_j |x_j - x_j'|^{p_j}, \tag{2}$$

with $\theta_j \geq 0$ and $1 \leq p_j \leq 2$. The parameters $\mu, \sigma^2, \theta_j, p_j$ are parameters of the prior. For analysis, we adopt an empirical Bayes approach (Currin et al. 1991): These parameters are estimated from the data from the computer model runs and "plugged into" the distribution of $Y(\mathbf{x})$.

Experience in a variety of circumstances (Higdon et al. 2004; Linkletter et al. 2006) suggests that very smooth, even analytic, output is typical, especially in engineering contexts. As such, it is often the case that p_j is fixed at 2 for all j, leading to the Gaussian correlation function. We adopt this special case for most of the article, but comment on the issue of $p_j < 2$ in Sections 6 and 7. With $p_j = 2$, it can be easily shown that

$$E\left|\frac{\partial Y(\mathbf{x})}{\partial x_j}\right|^2 = 2\sigma^2\theta_j.$$

Thus the weight θ_j may be interpreted as a measure of the "sensitivity" of $Y(\mathbf{x})$ to x_j. Characterizing the distribution of the distances in (2) across design points as a function of the values of the sensitivity measures, $\theta_1, \ldots, \theta_d$, leads to an understanding of the factors affecting prediction accuracy and hence sample size (Section 4).

Suppose that we make n runs of the code at a design D of input vectors $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$ in $[0, 1]^d$. (Other scales for $\mathbf{x}$ lead to rescaling of $\boldsymbol{\theta}$. Our comments about numerical values for $\boldsymbol{\theta}$ in the rest of the article also would need to be rescaled.) The data are denoted by $\mathbf{y} = (y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(n)}))^T$. The (empirical Bayes) predictor $\hat{Y}(\mathbf{x})$ of $Y(\mathbf{x})$ is the posterior mean of $Y(\mathbf{x})$ given the data and $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_d)$,

$$\hat{Y}(\mathbf{x}) = E(Y(\mathbf{x})|\mathbf{y}, \boldsymbol{\theta}) = \hat{\mu} + \mathbf{r}^T(\mathbf{x})\mathbf{R}^{-1}(\mathbf{y} - \mathbf{1}\hat{\mu}), \tag{3}$$

where $\mathbf{r}(\mathbf{x}) = (R(\mathbf{x}, \mathbf{x}^{(1)}), \ldots, R(\mathbf{x}, \mathbf{x}^{(n)}))^T$ is an $n \times 1$ vector of correlations from (1), $\mathbf{R}$ is an $n \times n$ matrix with element i, j given by $R(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$, $\hat{\mu}$ is an estimate of μ (often the maximum likelihood estimate), and $\mathbf{1}$ is an $n \times 1$ vector with all elements equal to 1. The mean squared error (MSE) of $\hat{Y}(\mathbf{x})$, taking into account the uncertainty from estimating μ by maximum likelihood, is given by

$$\begin{aligned}&\mathrm{MSE}(\hat{Y}(\mathbf{x}))\\&\quad = E(\hat{Y}(\mathbf{x}) - Y(\mathbf{x}))^2\\&\quad = \sigma^2\left(1 - \mathbf{r}^T(\mathbf{x})\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}) + \frac{(1 - \mathbf{1}^T\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}))^2}{\mathbf{1}^T\mathbf{R}^{-1}\mathbf{1}}\right). \quad (4)\end{aligned}$$

In practice, σ^2 and $\boldsymbol{\theta}$ also must be estimated, again often by maximum likelihood (Welch et al. 1992). This MSE is the Bayes posterior variance conditional on plugging in the covariance parameters.

The MSE in (4) can be computed directly given an experimental design and $\boldsymbol{\theta}$, and it is used in Section 4 for theoretical arguments. But for our empirical studies, we take a different path to defining prediction accuracy, using leave-one-out cross-validation (CV) (Currin et al. 1991; Chapman et al. 1994; Gough and Welch 1994). Denote the CV prediction of the output $y(\mathbf{x}^{(i)})$ from code run i by $\hat{Y}_{-i}(\mathbf{x}^{(i)})$, which is the predictor (3) based on the data from the $n - 1$ runs excluding run i. The n cross-validated errors, $\hat{Y}_{-i}(\mathbf{x}^{(i)}) - y(\mathbf{x}^{(i)})$, for $i = 1, \ldots, n$, are summarized by (normalized) average and maximum measures of inaccuracy,

$$e_{\mathrm{avg}} = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_{-i}(\mathbf{x}^{(i)}) - y(\mathbf{x}^{(i)}))^2}}{\text{range of } y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(n)})} \quad (5)$$

and

$$e_{\mathrm{max}} = \frac{\max_{\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(n)}} |\hat{Y}_{-i}(\mathbf{x}^{(i)}) - y(\mathbf{x}^{(i)})|}{\text{range of } y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(n)})}. \quad (6)$$

In these definitions, we could use the sample standard deviation of $y(\mathbf{x}^{(1)}), \ldots, y(\mathbf{x}^{(n)})$ in place of the range for normalization, but we find explanations using the range more appealing. Tolerable levels of inaccuracy are application-specific, but we typically take $e_{\mathrm{avg}} < 0.1$ as the target for a "useful" approximation of the code.

To gain insight into the effect of initial sample size, we can repeatedly simulate data from the GP model with given values of $\boldsymbol{\theta}$. We distinguish error measures based on data from code runs versus data from simulations by using $e_{\mathrm{avg|code}}$ and $e_{\mathrm{max|code}}$ versus e_{avg} and e_{max}. The collection of simulated e_{avg} values provides an empirical distribution for e_{avg} in (5), and averaging over them gives an estimate of $E(e_{\mathrm{avg}})$. Similarly, we can get an estimate of $E(e_{\mathrm{max}})$ in (6).

Why proceed with simulation rather than attempt direct computation of expected values, for example? For several reasons. First, the ratios in (5) and (6) are appealing measures but difficult to manipulate theoretically. Expected values or other quantities can be readily estimated via simulation. Second, as described in Section 6, after the sample size is selected and a computer experiment is run, we can evaluate $e_{\mathrm{avg|code}}$ and $e_{\mathrm{max|code}}$. The error measures from the experiment conducted can be compared with those from simulation (with $\boldsymbol{\theta}$ estimated from the code runs). By comparison with the the simulated empirical distribution, we can gauge whether the GP model and sample size are well matched to the actual code.

3. G–PROTEIN COMPUTER CODE

We now illustrate some of the issues raised in this article using an example, after which we attempt to generalize to a wide class of functions in the remainder of the article. A code modeling ligand activation of G-protein in yeast described by Yi et al. (2005) solves a system of ordinary differential equations (ODEs) with nine parameters that can vary. The system dynamics, the differential equations, are given by

$$\begin{aligned}\dot{\eta}_1 &= -u_1\eta_1 x + u_2\eta_2 - u_3\eta_1 + u_5,\\ \dot{\eta}_2 &= u_1\eta_1 x - u_2\eta_2 - u_4\eta_2,\\ \dot{\eta}_3 &= -u_6\eta_2\eta_3 + u_8(G_{\mathrm{tot}} - \eta_3 - \eta_4)(G_{\mathrm{tot}} - \eta_3),\\ \dot{\eta}_4 &= u_6\eta_2\eta_3 - u_7\eta_4,\end{aligned}$$

where $\eta_1, \ldots, \eta_4$ are concentrations of four chemical species, $\dot{\eta}_i \equiv \frac{\partial \eta_i}{\partial t}$, x is the concentration of the ligand, and $u_1, \ldots, u_8$ is a vector of eight kinetic parameters. The output, $y = (G_{\mathrm{tot}} - \eta_3)/G_{\mathrm{tot}}$, is the normalized concentration of a relevant part of the complex, where G_{tot} is the (fixed) total concentration of G-protein complex after 30 seconds.

For demonstration purposes, we fix five of the kinetic parameters, allowing only u_1, u_6, u_7, and x to vary. We use the GP model to construct an approximation of y as a function of the transformed variables $\log(u_1)$, $\log(u_6)$, $\log(u_7)$, and $\log(x)$, and then further transform each of these to $[0, 1]$. Thus these are $d = 4$ input variables, called $x_1, \ldots, x_4$ in the notation of Section 2. The ODE solver can be run quickly and allows us to evaluate the effect of n on the e_{avg} criterion in (5) using a real model. The designs used are maximin LHDs.

The values of n that we use are multiples (5, 7, 10, 15, and 20) of the dimension, $d = 4$. For each choice of n, we run the ODE solver to obtain data $\{y(\mathbf{x}^{(i)}); \mathbf{x}^{(i)} \in D\}$. The data are modeled as a realization of a GP (Section 2), the parameters are refit using maximum likelihood, and the code runs are used to calculate $e_{\mathrm{avg|code}}$ in (5). We also compute a version of $e_{\mathrm{avg|code}}$ using a set of new test points rather than CV. We generate the test points by running the ODE solver for each point in a 120-run maximin LHD. The same 120 test runs are used for all evaluations. The analogous version of e_{avg} replaces the numerator in (5) by $\sqrt{\frac{1}{120}\sum(\hat{Y}(\mathbf{x}) - y(\mathbf{x}))^2}$.

Figure 1 shows how the CV and test set $e_{\mathrm{avg|code}}$ measures change with n. Note that $e_{\mathrm{avg|code}}$ is <0.05 for all sample sizes; this is an easy function. Moreover, the rate of improvement with n appears to be small here in *absolute* terms, because the $e_{\mathrm{avg|code}}$ values are small. Thus there is little change in $e_{\mathrm{avg|code}}$ as n increases past $n = 10d = 40$. In *relative* terms, however, $e_{\mathrm{avg|code}}$ can be reduced substantially from its already small value by increasing the sample size. Furthermore, the differences between using CV and a new test sample to compute $e_{\mathrm{avg|code}}$ are not substantial; both measures point to an easy prediction problem. That CV generally leads to larger errors is not surprising, because leaving out one point can produce a big gap

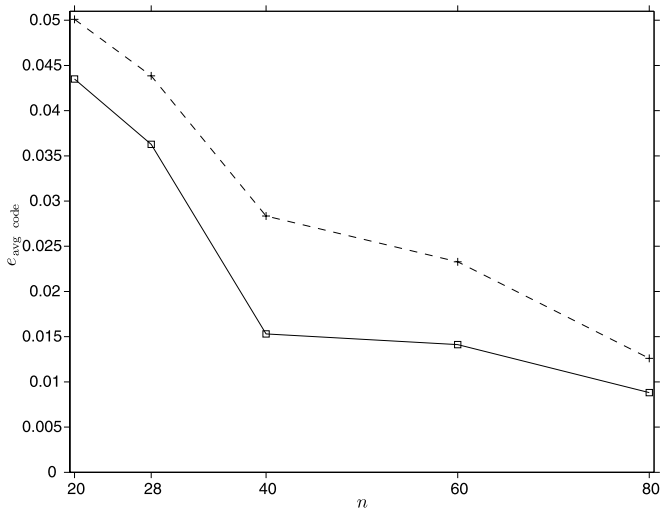

Figure 1. $e_{\text{avg|code}}$ against n for the G-protein example. $e_{\text{avg|code}}$ is computed via leave-one-out cross-validation (dashed line) or using 120 new test points (solid line).

in the design, making it hard to predict the omitted point. Because the use of new test data is a luxury, enjoyed only if the code can be run quickly, we usually are led to rely on CV for measuring accuracy.

We would not try many values of n in practice. Suppose that we were to conduct only the experiment with $n = 10d = 40$ runs. For this design, $e_{\text{avg|code}} = 0.028$ from CV (see Figure 1). For most applications, this would be considered small, and there would be nothing further to do. On the other hand, suppose that we wanted to reduce $e_{\text{avg|code}}$ by half. How many more runs would be needed? Based on the GP parameter estimates from the 40-run design, we can simulate data for other values of n and compute (simulated) cross-validated e_{avg} values. Figure 2 shows $e_{\text{avg}}/e_{\text{avg|code}}$, where e_{avg} from simulation varies with n

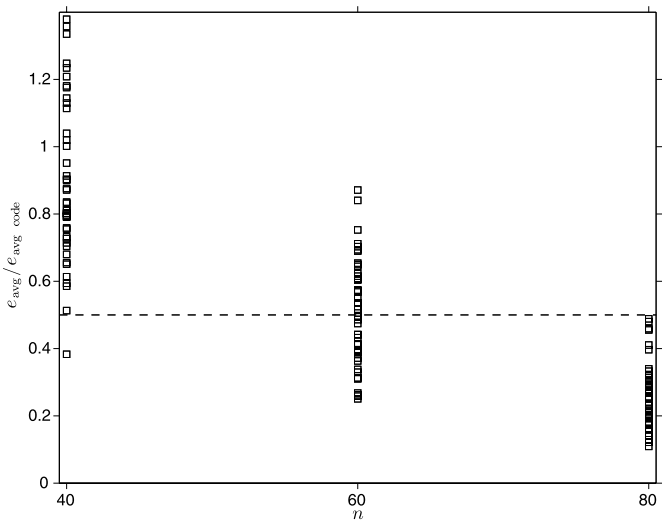

Figure 2. $e_{\text{avg}}/e_{\text{avg|code}}$ against n for the G-protein example. e_{avg} is from simulation and varies with n, and $e_{\text{avg|code}}$ is computed for a fixed 40-run design. Both measures are based on leave-one-out cross-validation. The horizontal line at $e_{\text{avg}}/e_{\text{avg|code}} = 0.5$ shows the desired reduction in error.

but $e_{\text{avg|code}}$ is for the fixed 40-run design. The figure suggests that about $n = 80$ runs would almost certainly cut the average error rate in half. A similar conclusion follows from using e_{avg} based on the 120 test points. Returning to Figure 1, we see that the experiment with $n = 80$ runs indeed leads to almost exactly the desired reduction in error.

The G-protein application establishes that simulation from a GP model may have properties at least close to mimicking reality. But this is just one example, and we would like to know the effect of n on prediction accuracy for a wider class of functions of higher dimensionality and greater complexity. For an efficient, insightful, and more general simulation study, we need to know the important factors determining the predictability of functions generated by a GP. This is the subject of the next section.

4. EFFECT OF d, $\boldsymbol{\theta}$, AND n ON PREDICTION ACCURACY

Intuitively, we know that design-point neighbors of $\mathbf{x}$ will tend to be closer as n grows larger, leading to improved accuracy in predicting $Y(\mathbf{x})$. But if $\boldsymbol{\theta}$ has many large values, then the correlation between $Y(\mathbf{x})$ and Y for the neighbors will be low, even for nearby points, leading to poorer prediction accuracy. Here we develop this intuition into some quantitative rules relating d, $\boldsymbol{\theta}$, and n to distances and the correlation structure, shedding some light on how prediction accuracy depends on these quantities.

First, we consider how the theoretical MSE, $\text{MSE}(\hat{Y}(\mathbf{x}))$ in (4), depends on d, $\boldsymbol{\theta}$, and n. Recall that the empirical definitions of prediction accuracy in (5) and (6) are normalized for scale, so, without loss of generality, we can ignore σ^2 in a normalized version of mean squared prediction error (MSPE),

$$\text{MSE}_{\text{norm}}(\hat{Y}(\mathbf{x})) = 1 - \mathbf{r}^T(\mathbf{x})\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}) + \frac{(1 - \mathbf{1}^T\mathbf{R}^{-1}\mathbf{r}(\mathbf{x}))^2}{\mathbf{1}^T\mathbf{R}^{-1}\mathbf{1}}. \tag{7}$$

$\text{MSE}_{\text{norm}}(\hat{Y}(\mathbf{x}))$ is determined by $\mathbf{R}$ and $\mathbf{r}(\mathbf{x})$ only. Thus it is a function of n, because $\mathbf{R}$ is an $n \times n$ matrix and $\mathbf{r}(\mathbf{x})$ is an $n \times 1$ vector, and of the correlations in $\mathbf{R}$ and $\mathbf{r}(\mathbf{x})$. Dimensionality, d, affects $\text{MSE}_{\text{norm}}(\hat{Y}(\mathbf{x}))$ only indirectly via these correlations.

For simplicity, we explore the factors affecting $\text{MSE}_{\text{norm}}(\hat{Y}(\mathbf{x}))$ for completely random LHDs (where the columns are permuted independently). For fixed n, we derive the mean and variance of the distribution of the distance (2) between design points. This leads to a distribution for the correlations in $\mathbf{R}$. We also illustrate that under the same conditions, the distribution of correlations in $\mathbf{r}(\mathbf{x})$ for $\mathbf{x}$ drawn randomly from $[0, 1]^d$ is similar. The matrix inverse in (7) makes MSE_{norm} much more complicated than can be explained by these distributions. Nonetheless, the simulations in Section 5 show that the mean and variance of the distance distribution explain much of the effect of d and $\boldsymbol{\theta}$ on our empirical accuracy measures.

Take two points, $\mathbf{x}$ and $\mathbf{x}'$, at random from a random LHD. An LHD is defined here to have fixed grid points $\{0, 1/(n-1), \ldots, 1\}$ for each variable x_j. Let $h_j = |x_j - x'_j|$ be the unweighted distance in dimension j appearing in h in (2). The first two moments of h_j^2 are given by Lemma 1.

Lemma 1. Let h_j be the distance between two randomly chosen points, x_j and x_j' in dimension j for a random LHD. Then

$$P(h_j = i/(n-1)) = \frac{n-i}{\binom{n}{2}} \quad \text{for } i = 1, \ldots, n-1,$$

$$E(h_j^2) \equiv m_1(n) = \frac{1}{6}\frac{n(n+1)}{(n-1)^2},$$

and

$$\mathrm{Var}(h_j^2) \equiv m_2(n) = \frac{1}{180}\frac{n(n-2)(n+1)(7n+9)}{(n-1)^4}.$$

The proof of this lemma is provided in the Appendix. Note that the two moments converge to $1/6$ and $7/180$ as $n \to \infty$.

If $d = 1$, then the probability distribution $P(h_j = i/(n-1))$ in Lemma 1 is the distribution of all possible distances between distinct points x and x'. That is, because the design points cover the grid, every possible distance $i/(n-1)$ occurs $n-i$ times.

But if $d > 1$, not all of the possible distances over all dimensions will be observed in any one design, and we rely on the moments given in Lemma 1 to describe behavior. Specifically, for two randomly chosen points and all $p_j = 2$, the squared distance in (2) has expectation

$$E(h) = m_1(n)\sum_{j=1}^{d}\theta_j. \tag{8}$$

For a completely random LHD that has independently permuted columns,

$$\mathrm{Var}(h) = m_2(n)\sum_{j=1}^{d}\theta_j^2. \tag{9}$$

Figure 3(a) gives the empirical distribution of the correlation from two points chosen at random from a single random LHD,

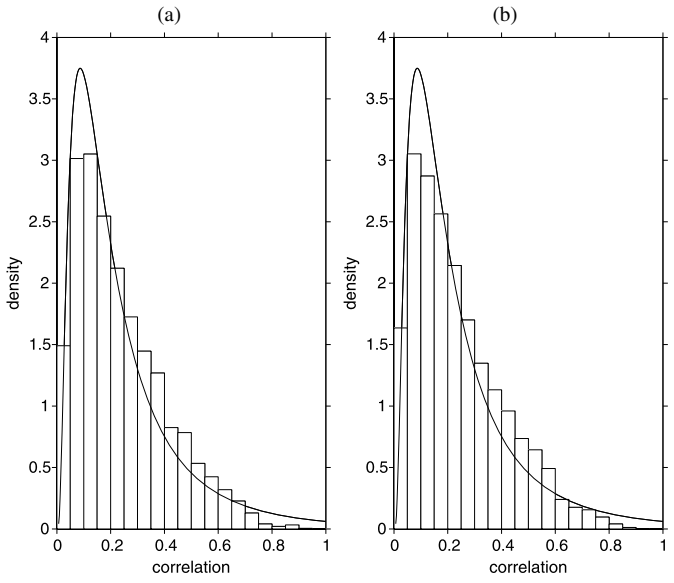

Figure 3. Correlation distribution. The correlation is from two design points randomly chosen from a random LHD (a) or from one design point and a new, random test point (b). The histograms are the respective empirical distributions, and the curves show the lognormal approximation.

with $d = 10$, $n = 100$, and $\boldsymbol{\theta} = (2.71, 2.17, 1.69, 1.27, 0.91, 0.61, 0.37, 0.19, 0.07, 0.01)$. The θ_j values comprise a canonical configuration, described in Section 5. Also shown is a lognormal approximation based on an assumption that h is approximately normal with mean and variance given by (8) and (9), and thus the correlation in (1) is approximately lognormal with these moments (after a change of sign). It can be seen that the approximation is fairly good but not perfect. The effect of the central limit theorem in the sum (2) is limited by the highly skewed distribution of h_j in Lemma 1, and hence that of h_j^2, and also by the few dominating θ_j weights here. All of this suggests that the moments (8) and (9) are important but do not completely characterize the distribution of correlations. Similarly, the right panel gives the empirical distribution and the lognormal approximation for the correlations in $\mathbf{r}(\mathbf{x})$, the vector arising from a randomly chosen $\mathbf{x} \in [0, 1]^d$ and the n design points. The distributions in the two panels are similar.

Intuitively, we would expect a favorable impact on prediction accuracy when the mean distance in (8) decreases, and hence the mean correlation increases. The impact on prediction accuracy also would be expected to be favorable when the variance in (9) *increases*. A larger spread for the distribution may push more mass out toward high correlations, and highly correlated neighboring points aid prediction accuracy.

There are two practical consequences of the dependence of the distributions of correlations on $\tau = \sum_{j=1}^{d}\theta_j$ and $\psi = \sum_{j=1}^{d}\theta_j^2$. First, these two quantities are used to plan the simulations in Section 5. We find that the behavior of the empirical analog, e_{avg}, of $\mathrm{MSE}_{\mathrm{norm}}$ is largely explained by τ and ψ. Second, the distributions of the correlations in $\mathbf{r}(\mathbf{x})$ (for random test points, $\mathbf{x}$) and in $\mathbf{R}$ (between design points) are similar. The implication is that accuracy estimates based on leave-one-out CV will be similar to estimates using predictions at random test points.

There are many possible $\theta_1, \ldots, \theta_d$ configurations. Here we examine three special cases describing the effect of dimensionality:

1. Suppose that $\theta_1 = \cdots = \theta_d = \theta$; that is, as dimensionality increases, further equally active variables are added. Then $\tau = d\theta$ and $\psi = d\theta^2$. Thus the mean of the distribution of h increases linearly with d, the standard deviation of the distribution increases as $\sqrt{d}$, and the h distribution becomes stochastically larger with d. For sufficiently large d, prediction accuracy will be poor, even if θ is small.
2. Suppose that τ is kept constant; that is, a fixed amount of total sensitivity is spread across all dimensions. Clearly, ψ takes its minimum value of $\psi = \tau^2/d$ when $\theta_1 = \cdots = \theta_d = \tau/d$. Thus equally active factors appear to be the worst for prediction accuracy. Moreover, as $\psi = \tau^2/d$ decreases with d, this effect becomes even worse as d increases. For sufficiently large d, the h distribution will become concentrated at its mean, $m_1(n)\tau$, and the limiting accuracy will depend on τ. In this sense, if the total amount of sensitivity is kept constant, then the *worst-case* effect of dimensionality is small.
3. Still keeping τ constant, larger values of ψ, and hence greater accuracy, result when $\theta_1, \ldots, \theta_d$ vary across the

input variables. The limiting *best-case* performance occurs when $\psi = \tau^2$ from $d-1$ inactive input variables with $\theta_j = 0$, that is, there is maximal *effect sparsity*.

The argument that accuracy decreases as $\tau = \sum_{j=1}^{d} \theta_j$ increases or as $\psi = \sum_{j=1}^{d} \theta_j^2$ decreases is borne out by the simulations in Section 5. The quantitative effect of n on accuracy is less obvious, however. The mean and variance in (8) and (9) depend weakly on n, and so $\mathbf{R}$ and $\mathbf{r}(\mathbf{x})$ in (7) have elements that depend only weakly on n. In contrast, MSE_{norm} depends on n because $\mathbf{R}$ and $\mathbf{r}(\mathbf{x})$ have *more* elements. A full analysis is complicated by the inverse of $\mathbf{R}$ in (7). Intuitively, harder problems (i.e., larger τ and smaller ψ) require larger sample sizes, regardless of dimensionality to a large extent; this is borne out by the simulations in Section 5.

5. SIMULATION RESULTS FOR AVERAGE ERROR

The arguments in Section 4 suggest that the effect of the correlation parameters on e_{avg} is largely through τ and ψ, thereby diminishing the role of d. To investigate this further, we perform a simulation study that changes dimension but keeps τ, ψ fixed. Before doing so, however, we must decide on the configurations of the $\boldsymbol{\theta}$ vectors to be explored. Past experience has indicated that for well-behaved input-output functions, there may be a few large components of $\boldsymbol{\theta}$ and a few moderately sized components, with the remainder small. For example, for the G-protein model and the 40-run experiment, $\hat{\boldsymbol{\theta}} = (0.11, 0.63, 0.56, 1.80)$ has one larger value, two moderate values, and one small value. From this standpoint, we adopt a two-parameter class of *canonical configurations* of $\boldsymbol{\theta}$, defined by

$$\theta_j = \tau\left[\left(1 - \frac{j-1}{d}\right)^b - \left(1 - \frac{j}{d}\right)^b\right] \quad \text{for } j = 1, \ldots, d \text{ and } b \geq 1, \tau > 0. \quad (10)$$

Here $\boldsymbol{\theta}$ is scaled overall by $\sum_{j=1}^{d} \theta_j = \tau$, and θ_j decreases with j at a rate controlled by b. A larger value of b leads to a larger value of ψ. The generated $\boldsymbol{\theta}$ vector tends to have the characteristics that we expect, especially as d grows large. Examples of $\boldsymbol{\theta}$ configurations for $d = 10$ and $\tau = 1$ are given in Table 1. When $\tau \neq 1$, the value of $\boldsymbol{\theta}$ is found by multiplying each canonical θ_j in the table by τ.

Data for the simulation study are generated as follows:

1. Given d and n, select a maximin LHD, D, of n points in $[0, 1]^d$.
2. Fix values of $\mu = 0, \sigma^2 = 1$, $\mathbf{p} = 2$ and select a canonical $\boldsymbol{\theta}$ (as specified earlier) for the parameters of the GP given in (2). Because the measure of accuracy in (5) or (6) is standardized by the range, the particular value of $\sigma^2 = 1$ is irrelevant.
3. Generate 50 independent realizations of the GP, resulting in 50 different sets of observations $\{y(\mathbf{x}^{(i)}); \mathbf{x}^{(i)} \in D\}$.

For each data set, form a predictor using (3) and a value of e_{avg} in (5), with the value of $\boldsymbol{\theta}$ as in the data-generation step. Alternatively, for each data set, we could estimate $\boldsymbol{\theta}$ and construct a predictor with $\hat{\boldsymbol{\theta}}$. We found no essential difference between predictors based on $\boldsymbol{\theta}$ and $\hat{\boldsymbol{\theta}}$ in terms of our summary measures of prediction accuracy, and using the fixed $\boldsymbol{\theta}$ takes much less time.

We start with $d = 5$ and $b = 1$ in (10), which results in $\theta_j = \tau/5$ for $j = 1, \ldots, 5$. As we argue in Section 4, this choice of $\boldsymbol{\theta}$ minimizes ψ for a fixed τ and represent a "worst case" starting point. Thus for any given τ value, $\psi = \tau^2/5$ when $d = 5$. If τ and $\psi = \tau^2/5$ are kept constant as d changes, then the canonical $\boldsymbol{\theta}$ vector must satisfy $\sum_{j=1}^{d} \theta_j^2 = \tau^2/5$. For $d = 10$, 15, and 20, this means that $b = 3.45$, 5.51, and 7.55, respectively, in (10). Values of $\tau = 3$, 10, 20, and 40 are chosen to cover problems ranging in difficulty from "easy" to "very hard."

The results of Section 4 suggest that similar accuracy should be obtained for fixed values of n, τ and ψ, largely independent of d. Intuitively, however, we expect that n must increase with d. Figure 3 illustrates that n, τ, and ψ do not fully explain the behavior of the tails of the correlation distribution, and that large correlations in particular play a prominent role. Thus, we allow n to increase modestly with d, specifically linearly. We also allow different rates, that is, $n = kd$, where $k = 7$, 10, 15, or 20. The four panels in Figure 4 correspond to $\tau = 3$, 10, 20, and 40, respectively, with ψ fixed at $\tau^2/5$ in each. In each panel, curves are plotted for $d = 5$, 10, 15, and 20. A curve shows the mean of e_{avg} computed from the 50 realizations of the GP, which we denote by $\bar{e}_{\text{avg}}$, plotted against k (recall that $n = kd$). Several features of these plots are worth highlighting:

- The case $\tau = 3$ is an "easy" problem due to the small components of $\boldsymbol{\theta}$.
- When ψ is fixed, the curves for $d = 5$, 10, 15, and 20 are all quite close.
- When $n = 10d$ and $\tau \leq 10$, predictions on average are accurate to within about 10% of the range of the data; reliable fits are barely (or not) obtainable for $\tau \geq 20$.
- The improvement in fit for sample sizes exceeding $n = 10d$ is marginal.

Suppose that the decrease of $\bar{e}_{\text{avg}}$ with n is approximately of order n^{-c}. The rate c can be estimated from the points shown in Figure 4 from the slope of the least squares fit of $\log(\bar{e}_{\text{avg}})$ regressed on $\log(k)$. The estimated rates are given in Table 2. For easy problems ($\tau = 3$), convergence rates close to 1 are achievable for dimensions as large as $d = 20$, so that doubling sample size can reduce e_{avg} by about half. In contrast, in hard problems the rates of convergence can be very small; for example,

Table 1. Canonical configurations of $\boldsymbol{\theta}$ for $d = 10$ (to be scaled by τ)

b	ψ	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	θ_8	θ_9	θ_{10}
1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
3	0.18	0.271	0.217	0.169	0.127	0.091	0.061	0.037	0.019	0.007	0.001
9	0.45	0.613	0.253	0.094	0.030	0.008	0.002	0	0	0	0

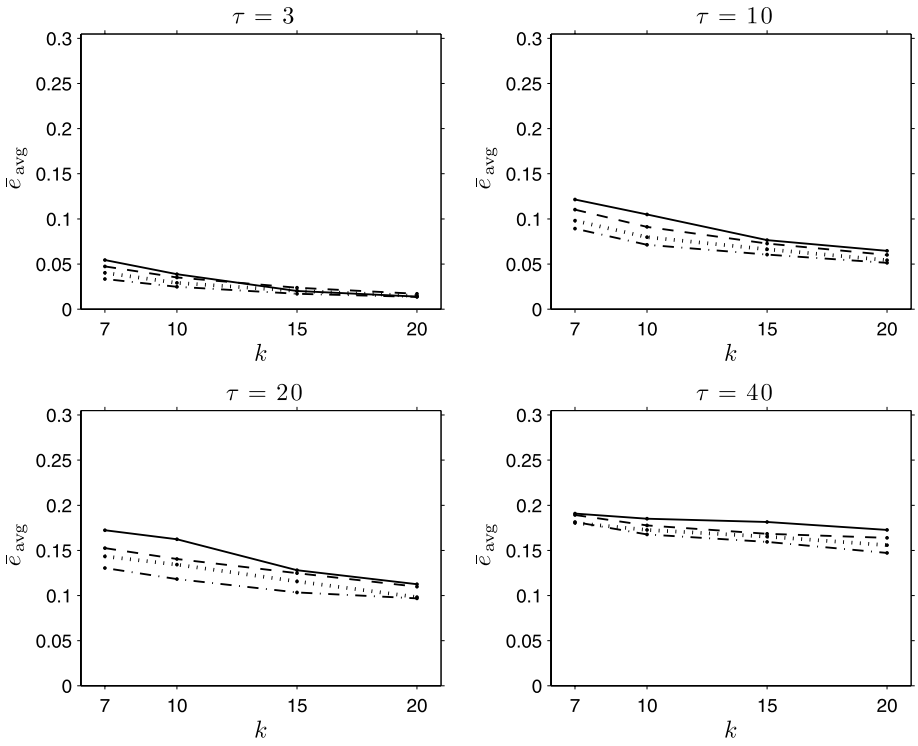

Figure 4. $\bar{e}_{\text{avg}}$ against k (with $n = kd$). The four panels correspond to $\tau = 3$, 10, 20, and 40. In each panel, $\psi = \tau^2/5$, and $d = 5$ (solid line), $d = 10$ (dashed line), $d = 15$ (dotted line), or $d = 20$ (dotted–dashed line).

when $d = 15$ and $\tau = 20$, it takes about 8 times as many runs to reduce e_{avg} by half. When $\tau = 40$, reducing e_{avg} substantially without enormous sample sizes appears to be impossible. In such situations, the experiment may need to be reformulated and restricted.

The arguments in Section 4 suggest that for fixed total sensitivity τ, dividing τ equally across the d input variables is the worst case for prediction accuracy, that is, $\psi = \tau^2/d$. Figure 5 explores worst-case problems by plotting e_{avg} against τ. There is a separate plot for $d = 5$, 10, 15, 20, and $n = 10d$ throughout. Fifty simulated realizations are made for each value of τ. The lines in Figure 5 drawn through the averages of e_{avg} show little difference as d increases, as predicted in Section 4 for the worst case studied here. There is a small dimensionality effect, and $n = 10d$ is increasing with d, but the total sensitivity, τ, is the important factor. For $\tau \geq 20$, e_{avg} is above the target of 0.1 for all d studied. In fact, if we replace the range in (5) by the sample standard deviation of $y_1, \ldots, y_n$, then all values of e_{avg} are close to 1 for values of $\tau > 20$, indicating that the sample mean is about as good a predictor as the GP model.

Table 2. Estimated convergence rates for $\bar{e}_{\text{avg}}$ (with $\psi = \tau^2/5$)

	τ			
d	3	10	20	40
5	1.34	0.63	0.43	0.08
10	0.97	0.57	0.31	0.14
15	0.96	0.53	0.36	0.13
20	0.87	0.51	0.28	0.19

To investigate the more realistic situation in which the problem has some degree of sparsity, we allow ψ to vary. We fix $d = 10$ and $n = 100$. As suggested by Figure 4, for fixed values of τ and ψ, results for other values of d (with $n = 10d$) are similar. For each fixed value of τ, we increase the value of ψ so that the sparsity is increased, and the total sensitivity of the function is shifted to increasingly fewer dimensions.

Even a moderate degree of sparsity can result in drastic reduction of error. The $\tau = 40$ panel in Figure 6 is interesting, because even in such a complex problem, reasonable accuracy can be obtained when there is a degree of sparsity. In particular, the last few values of ψ represent situations in which the 10-dimensional problem contains 5 or fewer active dimensions.

6. EXAMPLES

We now briefly revisit the G-protein example and discuss two climate codes in light of the foregoing results. From an initial design of $n = 10d = 40$ runs for the four-dimensional G-protein code in Section 3, $\hat{\boldsymbol{\theta}} = (0.11, 0.63, 0.56, 1.80)$, and we have $\hat{\tau} = 3.10$ and $\hat{\psi} = 3.95$. This is an even easier problem than that shown in the top left panel of Figure 4, where $\tau = 3$ is roughly the same but $d = 5$ is larger and ψ takes a worst-case value. In the figure, the average of e_{avg} across realizations is well below 0.05 for $n = 10d$; thus it is no surprise that $n = 10d$ runs led to $e_{\text{avg|code}} = 0.028$ for the easier G-protein problem.

The simulation in Section 3 suggests that 80 runs would reduce average error by approximately half relative to $n = 40$ for the G-protein example. This has been confirmed by a new experiment. Table 2 provides an alternative to simulation. For $d = 5$ and $\tau = 3$ in Table 2, which as noted earlier appears to

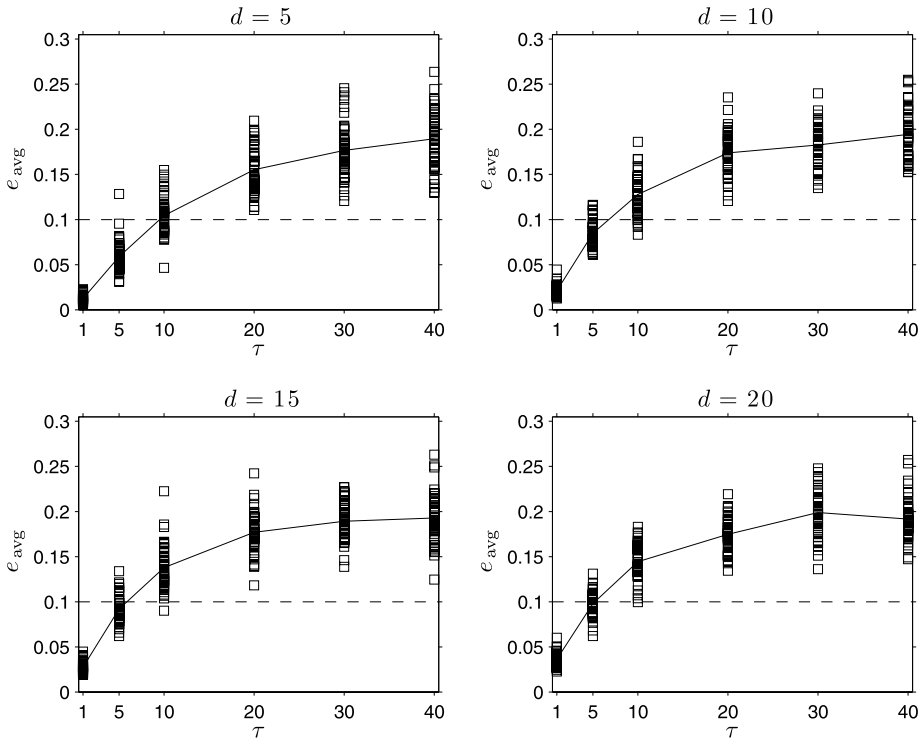

Figure 5. e_{avg} (squares) and $\bar{e}_{\text{avg}}$ (solid line) from 50 realizations against τ. The four panels correspond to $d = 5$, 10, 15, and 20. For all d and τ values, ψ is set at its worst-case value of τ^2/d, and $n = 10d$. The horizontal line indicates $e_{\text{avg}} = 0.1$.

characterize a harder problem, the rate of convergence is 1.34. Thus a sample size of $n = 40(2^{1/1.34}) = 67$ is required. But this rate calculation is based on average behavior across simulated realizations, and Figure 2 shows considerable variation in e_{avg}. Thus the rates in Table 2 provide a guide, but simulation of the e_{avg} distribution is safer.

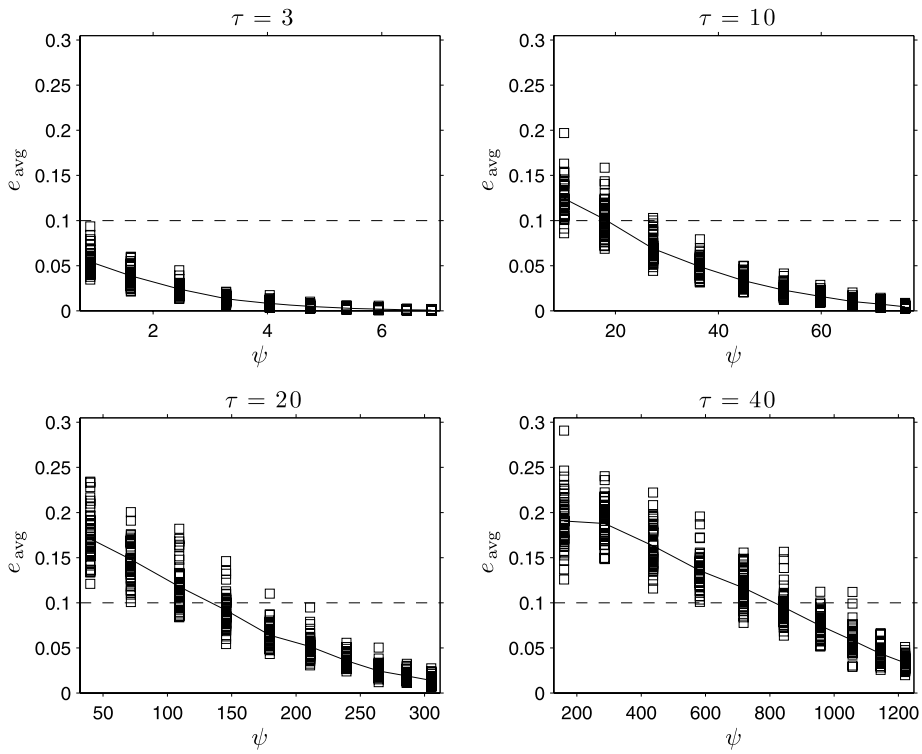

Figure 6. e_{avg} (squares) and $\bar{e}_{\text{avg}}$ (solid line) from 50 realizations against ψ. The four panels correspond to four values of τ. In each panel, $d = 10$, $n = 100$, and the horizontal line indicates $e_{\text{avg}} = 0.1$.

An ocean circulation model (Gough and Welch 1994) with $d = 7$ input variables had an initial sample size of $n = 36$ successful runs from 51 attempted. Thus, due to computing constraints, the sample size was about half the recommended value of $n = 10d$. Even so, fairly good fits were obtained for all six output variables. A closer look at this application (Loeppky, Sacks, and Welch 2008) reveals that all output variables have $\hat{\tau} < 10$. Moreover, $\hat{\boldsymbol{\theta}}$ always has elements that are near 0 for at least three (different) input variables; that is, there is considerable effect sparsity. Figure 6 suggests that very high accuracy would be produced by $n = 10d$ runs under these conditions, and it is not surprising that even fewer runs were sufficient. Alternatively, one can argue that with at least three impotent variables, the input space is effectively four-dimensional, leading to a recommendation of $n = 40$, about the same as in the experiment performed.

Chapman et al. (1994) analyzed a computer code for the seasonal growth and decline of Arctic sea ice. The code has $d = 13$ input variables and four outputs, $y_1, \ldots, y_4$. From an initial design producing $n_1 = 69$ runs, GPs were fit separately for each output. Every fitted GP had at least one input variable with $\hat{p}_j < 2$ (based on a substantially improved likelihood vs. $\hat{p}_j = 2$). The values of $e_{\text{avg|code}}$ are given in Table 3; each is below 0.1, and we might be tempted to stop. The $e_{\text{max|code}}$ values for y_3 and y_4, about 0.5, are of concern, however.

Faced with similar concerns about approximation accuracy in the initial experiment, Chapman et al. (1994) opted to make additional runs and ended up with a total of 157 good code runs. The $e_{\text{avg|code}}$ values in Table 3 show modest improvement for 157 runs, but the $e_{\text{max|code}}$ values show less change. For the troublesome y_3 and y_4, they remain close to 0.5.

Could simulation predict the impact of such a follow-up experiment? First consider the estimated means and standard deviations for e_{avg} and e_{max} in Table 3 for $n = 69$. There the $e_{\text{avg|code}}$ values are broadly consistent with the respective e_{avg} distributions. In contrast, simulation is less useful for e_{max} here; for example, $e_{\text{max|code}}$ for y_4 is 3.7 estimated standard deviations higher than the estimated mean. Estimated means and standard deviations of the e_{avg} and e_{max} distributions are also given in Table 3 for $n = 157$. To mimic practice, these simulations use the GP parameter estimates from the initial experiment. Relative to $n = 69$, simulation suggests only modest reduction in e_{avg}. For y_4, even this modest reduction is not realized by $e_{\text{avg|code}}$. With $n = 157$ runs, the simulated values of e_{max} again are inconsistent with $e_{\text{max|code}}$. Although the magnitude of the maximum error is again underestimated, the simulations correctly predict that there will be little impact on $e_{\text{max|code}}$ from the further runs. Thus the simulation study leads to the same conclusion that Chapman et al. (1994) reached after the follow-up experiment: Taking more runs is not effective.

The sea ice code failed to converge for 12 of 81 attempted runs in the initial design (hence the 69 good runs). This suggests erratic behavior of the code in some parts of the input space and is a possible explanation for the difference between the measure $e_{\text{max|code}}$ and the distribution of e_{max} based on simulations from the GP.

7. SUMMARY AND OPEN ISSUES

Our main conclusion is that $n = 10d$ runs is a reasonable rule of thumb for an initial experiment. For tractable problems like the G-protein and ocean-circulation models, good prediction accuracy is obtained. When initial accuracy is good, the rate of improvement with n also tends to be high, so error can be readily reduced by more runs, albeit *relative* to already small error summaries. When $n = 10d$ runs gives low accuracy, the rate of improvement also tends to be small. A small initial experiment is sufficient to show that huge sample sizes are suggested, perhaps calling for problem reformulation. Thus our results also provide some guidance regarding follow-up strategies.

Problem difficulty is characterized by $\theta_1, \ldots, \theta_d$ in (2), which are measures of the sensitivity to $x_1, \ldots, x_d$ in a GP model. Of primary concern is the value of $\tau = \sum \theta_j$, the total sensitivity. If τ is small (<10, say), then the problem is tractable with $n = 10d$ runs, for d up to 20, as studied in this work, possibly higher. Especially for large d, the degree of effect sparsity measured by $\psi = \sum \theta_j^2$ is also important. With a very high degree of sparsity, problems with larger values of τ (say up to 20 or 40) can be tackled with $n = 10d$ runs. In contrast, if the total sensitivity is spread equally across all d input dimensions, then even $\tau = 10$ leads to fairly poor prediction accuracy for $d > 5$. In such worst-case scenarios, the impact of dimensionality is minimal.

The values of τ discussed earlier assume that all x_j's are on a $[0, 1]$ scale. Consider a function with all x_j's on $[-1, 1]$, for instance. The θ_j would need to be multiplied by 4 to convert to this study's x_j scaling, because the Gaussian correlation works with squared distances. Another caveat is that we did not consider $d < 4$. Fast convergence of the maximum MSPE to 0 occurs for small d according to Chen (1996), even for the product designs that he considered. This is consistent with characterization via the ψ measure that we introduced; it is always favorable for functions of very low dimension, which have few (active) factors by definition.

Table 3. Actual and simulated accuracy measures for the sea ice code

		Average error					Maximum Error			
n		y_1	y_2	y_3	y_4		y_1	y_2	y_3	y_4
69	$e_{\text{avg\|code}}$	0.043	0.044	0.093	0.099	$e_{\text{max\|code}}$	0.249	0.124	0.466	0.559
	$\bar{e}_{\text{avg}}$	0.048	0.044	0.079	0.089	$\bar{e}_{\text{max}}$	0.139	0.128	0.225	0.263
	$\widehat{\text{SD}}(e_{\text{avg}})$	0.011	0.013	0.019	0.018	$\widehat{\text{SD}}(e_{\text{max}})$	0.039	0.052	0.071	0.079
157	$e_{\text{avg\|code}}$	0.032	0.031	0.079	0.096	$e_{\text{max\|code}}$	0.189	0.116	0.446	0.494
	$\bar{e}_{\text{avg}}$	0.029	0.029	0.056	0.062	$\bar{e}_{\text{max}}$	0.103	0.096	0.182	0.203
	$\widehat{\text{SD}}(e_{\text{avg}})$	0.008	0.009	0.011	0.011	$\widehat{\text{SD}}(e_{\text{max}})$	0.035	0.033	0.045	0.055

Effect sparsity is related to the notion of effective dimension. If there are good *a priori* reasons to expect that the number of active inputs, d_0, is less than d, then choosing $n = 10d_0$ is a useful complement to the recommended strategy, especially if there are serious budget constraints. In the sea ice example in Section 6, prior belief that there were likely to be no more than eight active inputs for each output accounted for the reliance on an initial sample size of 81.

Criteria can make a difference. In the sea ice example of Section 6, there is a conflict between the e_{avg} and $e_{\max}$ criteria in terms of whether satisfactory levels of accuracy are achievable without huge sample sizes. But calculations that are omitted here (Loeppky, Sacks, and Welch 2008) show that both criteria support the "$n = 10d$" rule.

The use of the *range* to normalize in (5) and (6) is driven by our sense that it is easier to interpret in reporting error rates. This is particularly true when the numerator is the maximum error. Normalizing using the sample standard deviation of the code output values is an alternative approach that leads to the same conclusions reported in Section 5. It has one possible advantage: For very hard problems the revised definition of e_{avg} will be close to 1 and readily recognized as a context in which the GP fit is of little use, either because the GP model is inappropriate or because the function is just too complex.

Good strategies for coping with poor accuracy from the GP model are not readily available. The approach of Gramacy and Lee (2008) may be useful when runs are plentiful. The technique used by Aslett et al. (1998) and Gramacy and Lee (2008) of narrowing the space of inputs often will lead to a less-complex function and allow better approximation of code output by a homogeneous GP; the assumption of homogeneity is less sustainable when the input space is too large. How to do this in a measured way is not clear and requires further research, however.

In most of the work reported in this article, we fixed $\mathbf{p} = 2$ in (2), leaving only the sensitivity parameters, $\boldsymbol{\theta}$, to vary. We chose a simple two-parameter canonical representation for $\boldsymbol{\theta}$ in our analyses of Section 5 to represent possible configurations. We have found that even if $\boldsymbol{\theta}$ is not a canonical configuration, there is little to no difference in the distributions of e_{avg} or $e_{\max}$ relative to a canonical $\boldsymbol{\theta}$, provided that τ and ψ are the same. Thus there is little impact on choice of initial sample size.

When $\mathbf{p} \neq 2$ (as in the sea ice example), the interpretation of τ and ψ values must be reexamined. In the case of the exponential correlation function ($\mathbf{p} = 1$), the implied prior distribution is on a much larger class of functions, and achieving good accuracy is more difficult. It is easy to work out the mean and variance of h_j^1 as in Lemma 1, and again we find (Loeppky, Sacks, and Welch 2008) that τ and ψ should be important. The mean of h_j^1 is now approximately twice that for the case $p_j = 2$, indicating that larger samples might be needed to achieve desired accuracy. When $1 < p_j < 2$, exact calculations of the mean and variance of $h_j^{p_j}$ are not available, but approximations are obtainable (Loeppky, Sacks, and Welch 2008), namely

$$E(h(\mathbf{x}, \mathbf{x}')) \approx \sum_{j=1}^{d} \theta_j \frac{2}{(p_j+1)(p_j+2)}$$

and

$$\operatorname{Var}(h(\mathbf{x}, \mathbf{x}')) \approx \sum_{j=1}^{d} \theta_j^2 \left(\frac{1}{(p_j+1)(2p_j+1)} - \frac{4}{(p_j+1)^2(p_j+2)^2} \right).$$

Defining canonical sets of correlation parameters is now more complicated. Not surprisingly, some preliminary calculations for the sea ice application suggest that the convergence rates for $\mathbf{p} \neq 2$ differ from those obtained when $\mathbf{p} = 2$, and thus we must examine rates for various combinations of both $\boldsymbol{\theta}$ and $\mathbf{p}$. How this all plays out in analogs of the analyses in Section 5 to enable follow-up recommendations has yet to be explored.

APPENDIX: PROOF OF LEMMA 1

Let D be an $n \times d$ random LHD, and let x_j and x_j' be any two randomly chosen runs of the design in dimension j. The construction of the LHD ensures that $x_j \neq x_j'$, and thus x_j and x_j' are dependent random variables. There are a total of $\binom{n}{2}$ possible pairs of points and each pair is equally likely. Clearly, $P(x_j = i/(n-1)) = 1/n$ and $P(x_j' = k/(n-1) | x_j = i/(n-1)) = 1/(n-1)$. Consider any two points that are an absolute distance of $i/(n-1)$ apart. By a simple counting argument, there are $n - i$ pairs giving rise to this distance. This establishes

$$P(h_j = i/(n-1)) = \frac{(n-i)}{\binom{n}{2}} = \frac{2(n-i)}{n(n-1)}, \qquad i = 1, \ldots, n-1.$$

The expected value of h_j^2 is

$$E(h_j^2) = E\left(\frac{i^2}{(n-1)^2}\right) = \frac{1}{(n-1)^2}\left(\frac{2}{n(n-1)}\sum_{i=1}^{n-1} i^2(n-i)\right)$$

$$= \frac{2}{n(n-1)^3}\left(n\sum_{i=1}^{n-1} i^2 - \sum_{i=1}^{n-1} i^3\right) = \frac{1}{6}\frac{n(n+1)}{(n-1)^2}.$$

Similarly,

$$\operatorname{Var}(h_j^2) = E(h_j^4) - E(h_j^2)^2 = E\left(\frac{i^4}{(n-1)^4}\right) - \left(\frac{1}{6}\frac{n(n+1)}{(n-1)^2}\right)^2$$

$$= \frac{1}{(n-1)^4}\left(\frac{2}{n(n-1)}\sum_{i=1}^{n-1} i^4(n-i)\right) - \left(\frac{1}{6}\frac{n(n+1)}{(n-1)^2}\right)^2$$

$$= \frac{1}{180}\frac{n(n-2)(n+1)(7n+9)}{(n-1)^4}.$$

Algebra was carried out in Maple.

ACKNOWLEDGMENTS

The research of Loeppky and Welch was supported by grants from the Natural Sciences and Engineering Research Council of Canada. The authors thank the associate editor and referees for improvements in the presentation.

[Received February 2008. Revised June 2009.]

REFERENCES

Aslett, R., Buck, R. J., Duvall, S. G., Sacks, J., and Welch, W. J. (1998), "Circuit Optimization via Sequential Computer Experiments: Design of an Output Buffer," *Applied Statistics*, 47, 31–48.

Bayarri, M. J., Berger, J. O., Cafeo, J., Garcia-Donato, G., Liu, F., Palmo, J., Parthasarathy, R. J., Paulo, R., Sacks, J., and Walsh, D. (2007), "Computer Model Validation With Function Output," *The Annals of Statistics*, 35, 1874–1906.

Chapman, W. L., Welch, W. J., Bowman, K. P., Sacks, J., and Walsh, J. E. (1994), "Arctic Sea Ice Variability: Model Sensitivities and a Multidecadal Simulation," *Journal of Geophysical Research*, 99, 919–935.

Chen, X. (1996), "Properties of Models for Computer Experiments," Ph.D. thesis, University of Waterloo, Dept. of Statistics and Actuarial Science.

Currin, C., Mitchell, T., Morris, M., and Ylvisaker, D. (1991), "Bayesian Prediction of Deterministic Functions, With Applications to the Design and Analysis of Computer Experiments," *Journal of the American Statistical Association*, 86, 953–963.

Gough, W. A., and Welch, W. J. (1994), "Parameter Space Exploration of an Ocean General Circulation Model Using an Isopycnal Mixing Parameterization," *Journal of Marine Research*, 52, 773–796.

Gramacy, R. B., and Lee, H. K. H. (2008), "Bayesian Treed Gaussian Process Models With an Application to Computer Modeling," *Journal of the American Statistical Association*, 103, 1119–1130.

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

Higdon, D., Kennedy, M., Cavendish, J. C., Cafeo, J. A., and Ryne, R. D. (2004), "Combining Field Data and Computer Simulations for Calibration and Prediction," *SIAM Journal on Scientific Computing*, 26, 448–466.

Jones, D. R., Schonlau, M., and Welch, W. J. (1998), "Efficient Global Optimization of Expensive Black-Box Functions," *Journal of Global Optimization*, 13, 455–492.

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

Liu, L. (2005), "Could Enough Samples Be More Important Than Better Designs for Computer Experiments?" in *Proceedings of the 38th Annual Simulation Symposium*, Washington, DC: IEEE Computer Society, pp. 107–115.

Loeppky, J. L., Sacks, J., and Welch, W. J. (2008), "Choosing the Sample Size of a Computer Experiment: A Practical Guide," Technical Report 170, National Institute of Statistical Sciences, available at *http://www.niss.org/sites/default/files/pdfs/technicalreports/tr170.pdf*.

McKay, M. D., Beckman, R. J., and Conover, W. J. (1979), "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output From a Computer Code," *Technometrics*, 21, 239–245.

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

Owen, A. B. (1994), "Controlling Correlations in Latin Hypercube Samples," *Journal of the American Statistical Association*, 89, 1517–1522.

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

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

Sahama, T. R., and Diamond, N. T. (2001), "Sample Size Considerations and Augmentation of Computer Experiments," *Journal of Statistical Computation and Simulation*, 68, 307–319.

Welch, W. J., Buck, R. J., Sacks, J., Wynn, H. P., Mitchell, T. J., and Morris, M. D. (1992), "Screening, Predicting, and Computer Experiments," *Technometrics*, 34, 15–25.

Welch, W. J., Buck, R. J., Sacks, J., Wynn, H. P., Morris, M. D., and Schonlau, M. (1996), "Response to James M. Lucas," *Technometrics*, 38, 199–203.

Yi, T.-M., Fazel, M., Liu, X., Otitoju, T., Goncalves, J., Papachristodoulou, A., Prajna, S., and Doyle, J. (2005), "Application of Robust Model Validation Using SOSTOOLS to the Study of G-Protein Signaling in Yeast," in *Proceedings of Foundations of Systems Biology and Engineering*, Santa Barbara, CA.