Using Orthogonal Arrays in the Sensitivity Analysis of Computer Models

Max D. Morris

Departments of Statistics and
Industrial and Manufacturing Systems Engineering
Iowa State University
Ames, IA 50010

Leslie M. Moore and Michael D. McKay

Statistical Sciences Group, CCS-6
Los Alamos National Laboratory
Los Alamos, NM 87545

We consider a class of input sampling plans, called permuted column sampling plans, that are popular in sensitivity analysis of computer models. Permuted column plans, including replicated Latin hypercube sampling, support estimation of first-order sensitivity coefficients, but these estimates are biased when the usual practice of random column permutation is used to construct the sampling arrays. Deterministic column permutations may be used to eliminate this estimation bias. We prove that any permuted column sampling plan that eliminates estimation bias, using the smallest possible number of runs in each array and containing the largest possible number of arrays, can be characterized by an orthogonal array of strength 2. We derive approximate standard errors of the first-order sensitivity indices for this sampling plan. We give two examples demonstrating the sampling plan, behavior of the estimates, and standard errors, along with comparative results based on other approaches.

KEY WORDS: Computer experiment; Experimental design; First-order variance coefficient; Replicated Latin hypercube sample; Uncertainty analysis

1. INTRODUCTION

Sensitivity analysis is the name often given to computational exercises in which the aim is to discover which inputs of a computer model are most influential in determining its outputs. In this article a computer model is taken to be a deterministic function,

$$y(x_1, x_2, x_3, \dots, x_k) = y(\mathbf{x}), \qquad \mathbf{x} \in \Delta,$$

where $\mathbf{x}$ is the k-element vector of inputs restricted to a domain Δ and y is the (for our purposes) scalar-valued output. For simple models, sensitivity analysis sometimes can be accomplished analytically, by direct examination of the computer model. But many computer models of interest to scientists and engineers are very complex; in these cases, sensitivity analysis is generally an empirical exercise in which a number of runs (or evaluations) of the model are made at selected input vectors, and the resulting output values are used to estimate sensitivity indices of interest.

One popular sensitivity index is the *first-order* (or *main-effect*) *sensitivity coefficient*, defined for each input as a measure of that input's importance. The definition of the first-order sensitivity coefficient (along with some other popular indices used in sensitivity analysis) depends on the specification of a probability distribution for the input vector $\mathbf{x}$. We denote this distribution by its probability density function $f(\mathbf{x})$ and limit attention here to situations in which the inputs are statistically independent, that is, $f(\mathbf{x}) = \prod_i f_i(x_i)$. In some cases, f has a physical interpretation, representing lack of control or uncertainty in the precise input values corresponding to a physical scenario of interest. In other cases, it may be a representation of expert judgement, and in still others, it may be selected fairly arbitrarily to cover input values deemed to be of interest. With respect to this distribution, the first-order sensitivity coefficient associated with input x_i is defined as

$$\theta_i = V_i\big[E_{-i}\{y(\mathbf{x})\}\big],$$

where V_i denotes the variance with respect to x_i and E_{-i} denotes the expectation with respect to all inputs except x_i. A normalized, unitless version of this index is $\eta_i^2 = \theta_i / V[y(\mathbf{x})]$, which always must be between 0 and 1. [Our notation for the normalized index is taken from the historical development of the *correlation ratio* (Pearson 1903); an example of a more accessible reference is Stuart and Ord 1991, pp. 1001–1002.] By a well-known variance decomposition formula, the first-order sensitivity coefficient also can be written as

$$\theta_i = V[y(\mathbf{x})] - E_i\big[V_{-i}\{y(\mathbf{x})\}\big].$$

θ_i is only one of a family of sensitivity coefficients that are related to the "functional ANOVA" decomposition of $y(\mathbf{x})$ (e.g., Hoeffding 1948), developed for this context by Archer, Saltelli, and Sobol' (1997).

2. ESTIMATION OF FIRST–ORDER SENSITIVITY COEFFICIENTS

The foregoing expression of θ_i is a convenient basis for estimating the coefficient using output values computed from sampled input vectors. For simplicity, consider a situation involving $k = 3$ inputs, and suppose that two random values are drawn independently from the marginal distributions of each input, resulting in input vectors

$$\mathbf{x}_1 = (x_1', x_2', x_3') \qquad \text{and} \qquad \mathbf{x}_2 = (x_1'', x_2'', x_3'').$$

Half of the squared difference of responses corresponding to these two runs is an unbiased estimate of $V[y(\mathbf{x})]$. Now suppose that a third input vector is constructed by drawing additional

TECHNOMETRICS, MAY 2008, VOL. 50, NO. 2
DOI 10.1198/004017008000000208

new values for x_2 and x_3 but "reusing" the first value drawn for x_1,

$$\mathbf{x}_3 = (x_1', x_2''', x_3''').$$

Half of the squared difference of responses corresponding to $\mathbf{x}_1$ and $\mathbf{x}_3$ is an unbiased estimate of $E_1[V_{-1}\{y(\mathbf{x})\}]$. These estimates can be improved by constructing sampling plans that contain, for each of several randomly selected values of x_1, several runs for which the values of all other inputs are drawn independently from run to run. If sensitivity coefficients are required for all inputs, then sets of runs with this property must be included for each individual x_i. Sampling plans must also include groups of runs within which all input values are different (and randomly selected), providing an estimate of $V[y(\mathbf{x})]$ based on a pooled sample variance. Then θ_i can be estimated as $\widehat{V[y(\mathbf{x})]} - E_i[\widehat{V_{-i}\{y(\mathbf{x})}\}]$.

2.1 Substituted Column Sampling

A straightforward approach to generating the kind of input sampling plan suggested in the previous paragraph was introduced by Sobol' (1993). The structure of this sampling plan can be specified by defining a collection of $k + 1$ arrays, $\mathbf{A}_0, \mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_k$, each of n rows and k columns. These can be thought of as coded design matrices, in that each array row is related to the values of the k inputs used in one run of the computer model.

For this kind of input sampling, we define two vectors,

$$\mathbf{u} = (1, 2, 3, \ldots, n)' \qquad \text{and}$$
$$\mathbf{v} = (n+1, n+2, n+3, \ldots, 2n)',$$

for the selected value of n. Then we define the arrays as

$$\mathbf{A}_0 = (\mathbf{u}, \mathbf{u}, \mathbf{u}, \ldots, \mathbf{u}), \qquad \mathbf{A}_1 = (\mathbf{u}, \mathbf{v}, \mathbf{v}, \ldots, \mathbf{v}),$$
$$\mathbf{A}_2 = (\mathbf{v}, \mathbf{u}, \mathbf{v}, \ldots, \mathbf{v}), \qquad \ldots,$$
$$\mathbf{A}_k = (\mathbf{v}, \mathbf{v}, \mathbf{v}, \ldots, \mathbf{u}).$$

Thus $\mathbf{A}_0$ and $\mathbf{A}_i$ have identical elements in their ith columns but different elements in all other columns. These coded arrays are finally converted to matrices of input vectors by drawing $2n$ values from each input distribution $[f_i(x_i), i = 1, 2, 3, \ldots, k]$. Values are drawn independently for each input; so, for example, "1" in column 1 always stands for the same drawn value for x_1, but "1" in any other column stands for a value drawn from the distribution of the corresponding input.

$V[y(\mathbf{x})]$ can be estimated as the pooled sample variance of outputs associated with each array, because the n input vectors specified by the rows of each array constitute a sample of $f(\mathbf{x})$. $E_i[V_{-i}\{y(\mathbf{x})\}]$ can be estimated as the pooled sample variance of n groups of two outputs associated with

the 1st rows of $\mathbf{A}_0$ and $\mathbf{A}_i$,

the 2nd rows of $\mathbf{A}_0$ and $\mathbf{A}_i$,

...

the nth rows of $\mathbf{A}_0$ and $\mathbf{A}_i$

because the input vectors in each group share a common value of x_i but different independent values of all other inputs.

The substituted column sampling plan is easy to implement, and has been widely used in applications. With the addition of n runs, it can be generalized to support estimation of *total sensitivity coefficients*, $E_{-i}[V_i\{y(\mathbf{x})\}]$, as described by Saltelli (2002). It is somewhat inefficient, however, because the estimate of $E_i[V_{-i}\{y(\mathbf{x})\}]$ is based only on the runs specified in arrays $\mathbf{A}_0$ and $\mathbf{A}_i$, that is, only $2n$ runs from the total of $(k+1)n$ runs required. The sampling plans described in the next section are more efficient in this sense.

2.2 Permuted Column Sampling

One very efficient input sampling plan that supports estimation of all $\theta_i, i = 1, \ldots, k$, which we call a *permuted column* sample, was described by McKay (1995). We note explicitly that these plans are not easily or obviously extendible to support estimation of total sensitivity indices, and so are of primary interest where first-order indices are of central interest. To specify such a plan, we continue to rely on the notation

$$\mathbf{u} = (1, 2, 3, \ldots, n)'.$$

The coded form of this sampling plan is denoted by a set of a arrays, each of dimension $n \times k$, and each comprising columns that are permutations of $\mathbf{u}$, that is,

$$\begin{aligned}\mathbf{A}_j &= \{\mathbf{a}_1^j, \mathbf{a}_2^j, \mathbf{a}_3^j, \ldots, \mathbf{a}_k^j\} \\ &= \{p_1^j(\mathbf{u}), p_2^j(\mathbf{u}), p_3^j(\mathbf{u}), \ldots, p_k^j(\mathbf{u})\}, \qquad j = 1, 2, 3, \ldots, a.\end{aligned}$$

One way to do this in practice is to simply randomly and independently select the $a \times k$ permutation operators, p_i^j, from the $n!$ selections available. Again, final input matrices are constructed by substituting n randomly selected values of x_i for the symbols "1" through "n" in the ith column of each array. As distinct from the substituted column sampling plan of Sobol', the same n values of x_i are used in each of the a arrays but are "matched" with different values of the other inputs in each array through the permutation operators. $V[y(\mathbf{x})]$ can be estimated as the pooled sample variance of outputs generated by the runs within each array; that is, if $S^2(\mathbf{A}_j)$ is the sample variance of the n outputs resulting from the runs in $\mathbf{A}_j$, then we define

$$\widehat{V[y(\mathbf{x})]} = \frac{1}{a}\sum_{j=1}^{a} S^2(\mathbf{A}_j).$$

$E_i[V_{-i}\{y(\mathbf{x})\}]$ can be estimated as the average of sample variances computed by grouping runs across arrays by common values of x_i; that is, for each input, there are n groups of runs (one run from each array) that share a common value of x_i but for which the remaining inputs have been "scrambled" by the permutation of columns in each array. If $S^2(x_{i,r})$ is the sample variance of the a outputs sharing the rth drawn value of x_i, then we define

$$E_i\big[\widehat{V_{-i}\{y(\mathbf{x})}\}\big] = \frac{1}{n}\sum_{r=1}^{n} S^2(x_{i,r}).$$

Combining these, the estimate of θ_i is

$$\hat{\theta}_i = \frac{1}{a}\sum_{j=1}^{a} S^2(\mathbf{A}_j) - \frac{1}{n}\sum_{r=1}^{n} S^2(x_{i,r}).$$

Thus all $a \times n$ runs are used in the estimation of both $V[y(\mathbf{x})]$ and $E_i[V_{-i}\{y(\mathbf{x})\}]$, for $i = 1, 2, 3, \ldots, k$.

2.2.1 Approximate Standard Errors for Permuted Column Samples. The sampling-based approach to sensitivity analysis is motivated in part by the desire to make as few a priori assumptions as possible concerning the relationship between y and $\mathbf{x}$. Constructing accurate, explicit standard errors of $\hat{\theta}_i$ and $\hat{\eta}_i^2$ is difficult in this context unless additional distributional assumptions are made. But standard errors based on the same data used in computing the estimates are important to the practical interpretation of a sensitivity analysis. In this section we construct approximate standard errors of $\hat{\theta}_i$ and $\hat{\eta}_i^2$ based on arguments analogous to two-way random-effects ANOVA models, even though this structure is at best a rough approximation.

For a specified input x_i, we relabel output values with subscripts, letting y_{jr} represent the output generated from running the model with the input vector specified in array j in which x_i is assigned the rth selected value. This notation suggests an unreplicated two-way table of output values with a rows and n columns. For derivation of approximate standard errors only, we adopt the working assumption that

$$y_{jr} = \mu + \alpha_j + \beta_r + \gamma_{jr},$$

where

$$\alpha_j \sim \mathrm{N}(0, \sigma_\alpha^2), \qquad \beta_r \sim \mathrm{N}(0, \sigma_\beta^2), \qquad \text{and}$$
$$\gamma_{jr} \sim \mathrm{N}(0, \sigma_\gamma^2),$$

and assume that each of these random quantities is independent of all others. Then $\hat{\theta}_i$ can be rewritten as

$$\begin{aligned}\hat{\theta}_i &= \frac{1}{a(n-1)} \sum_{j=1}^{a}\sum_{r=1}^{n}(y_{jr} - \bar{y}_{j.})^2 - \frac{1}{(a-1)n}\sum_{r=1}^{n}\sum_{j=1}^{a}(y_{jr} - \bar{y}_{.r})^2 \\ &= \frac{1}{a(n-1)}\left\{\sum_{j=1}^{a}\sum_{r=1}^{n}(\bar{y}_{.r} - \bar{y}_{..})^2 + \sum_{j=1}^{a}\sum_{r=1}^{n}(y_{jr} - \bar{y}_{j.} - \bar{y}_{.r} + \bar{y}_{..})^2\right\} \\ &\quad - \frac{1}{(a-1)n}\left\{\sum_{r=1}^{n}\sum_{j=1}^{a}(\bar{y}_{j.} - \bar{y}_{..})^2 + \sum_{r=1}^{n}\sum_{j=1}^{a}(y_{jr} - \bar{y}_{j.} - \bar{y}_{.r} + \bar{y}_{..})^2\right\} \\ &= \frac{1}{a}MS_\beta + \frac{1}{n}MS_\alpha + \left[\frac{a-1}{a} - \frac{n-1}{n}\right]MS_\gamma,\end{aligned}$$

where $MS_\alpha = \frac{1}{a-1}\sum_{j=1}^{a} n(\bar{y}_{j.} - \bar{y}_{..})^2$, $MS_\beta = \frac{1}{n-1}\sum_{r=1}^{n} a(\bar{y}_{.r} - \bar{y}_{..})^2$, and $MS_\gamma = \frac{1}{(a-1)(n-1)}\sum_{j=1}^{a}\sum_{r=1}^{n}(y_{..} - \bar{y}_{j.} - \bar{y}_{.r} + \bar{y}_{..})^2$. Regarding each mean square times the ratio of degrees-of-freedom to expected mean square as a chi-squared random variable, this leads to

$$V(\hat{\theta}_i) \approx 2\left[\frac{1}{a^2(n-1)}E(MS_\beta)^2 + \frac{1}{n^2(a-1)}E(MS_\alpha)^2 + \frac{(a-n)^2}{a^2n^2(a-1)(n-1)}E(MS_\gamma)^2\right].$$

Finally, the normality assumption implies that an unbiased estimate of $E(MS_\alpha)^2$ is $\frac{a-1}{a+1}MS_\alpha^2$, and similarly for $E(MS_\beta)^2$ and $E(MS_\gamma)^2$, so that under the random-effects ANOVA model, an unbiased estimate of $V(\hat{\theta}_i)$ is

$$2\left[\frac{1}{a^2(n+1)}MS_\beta^2 + \frac{1}{n^2(a+1)}MS_\alpha^2 + \frac{(a-n)^2}{a^2n^2[(a-1)(n-1)+2]}MS_\gamma^2\right].$$

We take the square root of this quantity for the standard error of $\hat{\theta}_i$.

Using the same notation, $\hat{\eta}_i^2$ can be reexpressed as

$$\hat{\eta}_i^2 = 1 - \frac{\frac{1}{n}MS_\alpha + \frac{n-1}{n}MS_\gamma}{\frac{1}{a}MS_\beta + \frac{a-1}{a}MS_\gamma}.$$

Again assuming independence among mean squares and relationships between expectations and variances based on chi-squared distributions, the delta method leads to an approximate variance

$$\begin{aligned}V(\hat{\eta}_i^2) \approx 2\frac{a^2}{n^2}\left(\frac{E(MS_\alpha) + (n-1)E(MS_\gamma)}{E(MS_\beta) + (a-1)E(MS_\gamma)}\right)^2 \\ \times \Bigg[\frac{1}{a-1}\frac{E(MS_\alpha)^2 + (n-1)E(MS_\gamma)^2}{(E(MS_\alpha) + (n-1)E(MS_\gamma))^2} \\ + \frac{1}{n-1}\frac{E(MS_\beta)^2 + (a-1)E(MS_\gamma)^2}{(E(MS_\beta) + (a-1)E(MS_\gamma))^2} \\ - 2E(MS_\gamma)^2\big((E(MS_\alpha) + (n-1)E(MS_\gamma)) \\ \times (E(MS_\beta) + (a-1)E(MS_\gamma))\big)^{-1}\Bigg].\end{aligned}$$

In this case, because the approximate variance involves squares and ratios of mean squares, we simply substitute mean squares for their expectations and take the square root of the foregoing expression as the standard error of $\hat{\eta}_i^2$.

The empirical performance of these standard errors, along with that of the point estimates themselves, is examined for various permuted column sampling plans in the example of Section 4.

2.3 Replicated Latin Hypercube Sampling

The *replicated Latin hypercube sample* (rLHS) is a refinement of the randomized permuted column sample, described by McKay (1995). To generate a Latin hypercube sample (LHS) corresponding to array $\mathbf{A}_1$, the range of each input is divided into an equal-probability partition of n intervals, that is,

$$x_i^{[1]} < x_i^{[2]} < \cdots < x_i^{[n-1]};$$
$$I_1 = (-\infty, x_i^{[1]}], \qquad I_2 = (x_i^{[1]}, x_i^{[2]}], \qquad \ldots,$$
$$I_n = (x_i^{[n-1]}, \infty);$$

and

$$P(x_i \in I_r) = n^{-1}, \qquad r = 1, 2, 3, \ldots, n.$$

The input value corresponding to "1" in the ith column of $\mathbf{A}_1$ is selected by picking (randomly) one of the elements of this partition and then drawing a value from the *conditional* distribution of x_i restricted to that interval. The input value corresponding to "2" is selected in a similar way, except that the partition element selected for "1" is eliminated; that is, intervals are drawn without replacement as the input values are selected. The result is a stratified sample, in which the strata are the equal-probability intervals, for each input. The resulting matrix of input values is then an LHS as introduced by McKay, Conover, and Beckman (1979), and the collection of a matrices generated through column permutation is an LHS. McKay et al. showed that Latin hypercube sampling is more efficient than unconstrained random sampling when the goal of the experiment is estimation of the mean output and the output is a monotonic function of each of the inputs. Subsequent research has led to further characterization of sampling efficiency for LHS and related sampling schemes (e.g., Stein 1987; Tang 1993; Owen 1994).

2.4 Estimation Bias Associated With Randomized Column Permutation

The intent of column permutation in the aforementioned sampling plans is to "scramble" individual input values into different combinations within each array, so that groups of a runs can be identified within which all inputs except one vary. Thus, for example, the a runs corresponding to the rows of $\mathbf{A}_1$ through $\mathbf{A}_a$ within which the first element is "4" are identified, and the sample variance of the corresponding values of y is computed as a component of the estimate of $E_1[V_{-1}\{y(\mathbf{x})\}]$. The resulting estimate would be unbiased if the random column permutations resulted in "perfect mixing" of the input values from array to array, that is, *if no two input values paired together in one array also were paired in another array*. But unconstrained random permutation of columns does not guarantee this result; in the foregoing example, there is some nonzero probability that two runs sharing the fourth selected value of x_1 also share a common value of another input. The result is that each estimate of $E_i[V_{-i}\{y(\mathbf{x})\}]$ is negatively biased (i.e., the within-group variances are too small on average), and thus the estimate of θ_i is positively biased. The extent of this bias was characterized by Morris, Moore, and McKay (2006); the magnitude of the bias is generally larger when the number of influential inputs is large or the number of runs in each array (n) is small. In the next section we describe a deterministic process for selecting the column permutations used to construct $\mathbf{A}_1$–$\mathbf{A}_a$, that eliminates estimator bias from such "accidental matching" of input values.

3. UNBIASED PERMUTED COLUMN SAMPLES

The following formal definitions provide useful structure and notation. Except in the statement of Theorem 1, we disregard how the actual values are selected for each input and focus on the combinatorial aspect of constructing the integer arrays discussed earlier. Proofs of all results are given in the Appendix. Both Latin hypercube and unconstrained random sampling are used in the demonstration exercise reported in Section 4.

Definition 1. Let $\mathbf{u} = (1, 2, 3, \ldots, n)'$. A *permuted column sample* (PCS) is a set of arrays, $\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3, \ldots, \mathbf{A}_a$, where the array columns are denoted as

$$\mathbf{A}_j = \{\mathbf{a}_1^j, \mathbf{a}_2^j, \mathbf{a}_3^j, \ldots, \mathbf{a}_k^j\},$$

such that each $\mathbf{a}_i^j$ is a permutation of $\mathbf{u}$ for all $i = 1, 2, 3, \ldots, k$ and $j = 1, 2, 3, \ldots, a$.

Definition 2. An *unbiased permuted column sample* (UPCS) satisfies Definition 1 and is constructed such that

$$(\{\mathbf{a}_i^j\}_r, \{\mathbf{a}_{i'}^j\}_r) \neq (\{\mathbf{a}_i^{j'}\}_{r'}, \{\mathbf{a}_{i'}^{j'}\}_{r'})$$

for all $i = 1, 2, 3, \ldots, k$, $i' = 1, 2, 3, \ldots, k$, $i \neq i'$; $j = 1, 2, 3, \ldots, a$, $j' = 1, 2, 3, \ldots, a$, $j \neq j'$; and $r = 1, 2, 3, \ldots, n$, $r' = 1, 2, 3, \ldots, n$.

Note that Definition 1 specifies only that each array column contain the same set of symbols, as explained in Section 2.2. Definition 2 also requires that no pair of inputs take the same pair of values in runs contained in any two arrays.

The following theorem is the formal statement of how the UPCS improves the quality of estimation of θ_i when unconstrained random sampling is used to select values for each input.

Theorem 1. Given a UPCS plan, with unconstrained independent sampling used to select values of the inputs, $E[\hat{\theta}_i] = \theta_i$, $i = 1, 2, 3, \ldots, k$.

Theorem 2, the main result of this article, describes the relationship between UPCS and orthogonal arrays of strength 2. The following two lemmas, which lead up to this theorem, state restrictions among a, k, and n that are necessary for the existence of a UPCS.

Lemma 1. If $n < k$, then a UPCS with $a > 1$ arrays does not exist.

Lemma 2. If $n = k$, then the number of arrays in any UPCS is $a \leq k$.

Here we limit our attention to unbiased permuted column samples for which each array is of minimum size ($n = k$; Lemma 1), and the number of arrays is as large as possible ($a = k$; Lemma 2). Thus $n = a = k$.

Theorem 2. Require that $n = a = k$. Let $\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3, \ldots, \mathbf{A}_n$ be a PCS. Define an n^2-by-$(k+1)$ array as

$$\mathbf{C} = \left(\begin{array}{ll}
1 & \\
1 & \\
1 & \mathbf{A}_1 \\
\cdots & \\
1 & \\
\hline
2 & \\
2 & \\
2 & \mathbf{A}_2 \\
\cdots & \\
2 & \\
\hline
\cdots & \cdots \\
\hline
n & \\
n & \\
n & \mathbf{A}_n \\
\cdots & \\
n &
\end{array}\right).$$

Then $\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3, \dots, \mathbf{A}_n$ is a UPCS if and only if $\mathbf{C}$ is an orthogonal array of strength 2, that is, an OA$(n^2, k+1, n, 2)$.

Recall that an array of n "symbols" is an orthogonal array of strength s if for each collection of s columns, each of the n^s possible ordered s-tuples appears in the same number of rows. (See, e.g., Hedayat, Sloane, and Stufken 1999 for a thorough treatment of the structure and construction techniques for orthogonal arrays.) For our result, $s = 2$, and so the defining geometric property applies in two-dimensional projections, reflecting the fact that we are focusing on *pairs* of input values in multiple runs. Further, Theorem 2 requires that the OA contain n^2 runs, immediately implying that each set of input values for each pair of inputs is included exactly once, eliminating the possibility of replication leading to the bias discussed in Section 2.4. Table 1 contains an example of an orthogonal array of strength 2, used in the demonstration described in Section 4. Note that for any $s = 2$ columns of this array, each of the $n^2 = 64$ ordered pairs of $n = 8$ symbols appears once.

Theorem 2 establishes that an OA$(n^2,\ k+1,\ n,\ 2)$ can be used to construct any UPCS when $n = a$. For a given number of inputs, k, the smallest such OA will require some value $n = a$ at least as great as k. Of course, once the OA is identified, not all $n = a$ generated arrays or k columns of each array need be used; any subset of the columns selected from any subset of the constructed $\mathbf{A}_j, j = 1, 2, 3, \dots, n$, also constitutes a UPCS with $k \le n$ and $a \le n$.

Orthogonal arrays also are useful for other purposes in computer experiments. Owen (1992) exploited the fact that they generalize the one-dimensional stratification structure of the LHS and thus can reduce the variance of multidimensional numerical integration relative to less-structured Monte Carlo methods.

Other related forms of structure also have been considered in the selection of input vectors for computer experiments. *Minimum discrepancy sequences*, such as those described by Faure, Niederreiter, and Sobol' (e.g., Niederreiter and Shiue 1995) are often used in numerical integration. In particular, sampling plans based on Sobol' sequences have been used, along with bootstrap techniques, as the basis for estimating sensitivity indices.

Table 1. OA(64, 9, 8, 2), g-function example

Rows 1–16	Rows 17–32	Rows 33–48	Rows 49–64
111111111	313333333	515555555	717777777
122345678	325162487	523271846	726854213
133456782	331624875	532718463	738542136
144567823	346248751	547184632	745421368
155678234	352487516	551846327	754213685
166782345	364875162	568463271	762136854
177823456	378751624	574632718	771368542
188234567	387516248	586327184	783685421
212222222	414444444	616666666	818888888
221583764	428617352	627438125	824726531
235837641	436173528	634381257	837265314
248376415	441735286	643812574	842653147
253764158	457352861	658125743	856531472
267641583	463528617	661257438	865314726
276415837	475286173	672574381	873147265
284158376	482861735	685743812	881472653

4. DEMONSTRATION: g-FUNCTION

Saltelli and Sobol' (1995) described the "g-function" and used it as a test case in demonstrating and comparing methods of sensitivity analysis. The g-function is defined as

$$y = \prod_{i=1}^{k} \frac{|4x_i - 2| + c_i}{1 + c_i}, \qquad \mathbf{x} \in [0, 1]^k.$$

Here the nonnegative function parameters c_i, $i = 1, 2, 3, \dots, k$, control the relative importance of each input; y is relatively sensitive to the values of inputs x_i for which the corresponding values of c_i are near 0 and less sensitive to inputs for which the parameters are larger. Although the model is not mathematically complicated, the product form and discontinuous derivative at $\frac{1}{2}$ in each dimension hampers empirical evaluation of sensitivity for some methods. An additional benefit for demonstration purposes is that the sensitivity coefficients can be easily determined by direct analysis.

In this demonstration, we investigated the behavior of a g-function in $k = 8$ arguments, with coefficients $\mathbf{c} = (0, 1, 1, 2, 3, 5, 8, 13)'$. Permuted column samples were generated consisting of $a = 8$ arrays, each of size $n = 8$ runs, for a total sample size of 64. Arrays $\mathbf{A}_j$ were constructed both by unconstrained random selection of column permutations and by construction based on an orthogonal array, OA(64, 9, 8, 2), as described in the previous section. The specific orthogonal array used was obtained from *http://www.research.att.com/~njas/oadir/* and is listed in Table 1. Table 2 shows how $\mathbf{A}_1$ was constructed as the first eight rows of columns 2–9 of the array; $\mathbf{A}_2$–$\mathbf{A}_8$ were constructed similarly using subsequent batches of eight array runs each. In each case, values of the inputs were drawn by unconstrained random sampling and by Latin hypercube sampling. The means and standard deviations of estimates of θ_i, $i = 1, 2, 3, \dots, 8$, based on a simulation study of 1,000 replications, are summarized in Table 3 for each of the 4 combinations of array structure and input selection. Here average (over the 1,000 simulations) values of each standard error also are tabulated, along with coverage proportions for intervals constructed as $\hat{\theta}_i \pm 2$ standard error $(\hat{\theta}_i)$. Similar results for $\hat{\eta}_i^2 = \hat{\theta}_i / \hat{V}(y)$ are presented in Table 4. Note that by symmetry of the g-function and because $c_2 = c_3 = 1$, any differences between the entries for inputs 2 and 3 reflect only simulation error.

The general patterns of Tables 3 and 4 are quite similar. The UPCS plan, using unconstrained sampling of input values, results in mean estimates that are within simulation error of the

Table 2. $\mathbf{A}_1$ Constructed from the first eight rows of the OA(64, 9, 8, 2)

1	1	1	1	1	1	1	1
2	2	3	4	5	6	7	8
3	3	4	5	6	7	8	2
4	4	5	6	7	8	2	3
5	5	6	7	8	2	3	4
6	6	7	8	2	3	4	5
7	7	8	2	3	4	5	6
8	8	2	3	4	5	6	7

NOTE: $\mathbf{A}_2$–$\mathbf{A}_8$ are constructed similarly from subsequent sets of 8 rows.

Table 3. Means and standard deviations (SDs) of $\hat{\theta}_i$, average standard errors (ASEs), and coverage of approximate 95% confidence intervals, g-function example

i	1	2	3	4	5	6	7	8
c_i	0	1	1	2	3	5	8	13
θ_i (analytical)	.3333	.0833	.0833	.0370	.0208	.0093	.0041	.0017
Random arrays, unconstrained sampling								
Mean	.3723	.1509	.1519	.1087	.0912	.0815	.0767	.0752
SD	.2085	.1126	.1140	.0866	.0767	.0665	.0624	.0635
ASE	.1814	.0772	.0777	.0575	.0493	.0448	.0426	.0419
Coverage	.8520	.9050	.9210	.9610	.9900	.9930	.9720	.9370
Random arrays, Latin hypercube sampling								
Mean	.4105	.1602	.1581	.1177	.0981	.0866	.0830	.0807
SD	.1337	.0861	.0872	.0754	.0652	.0558	.0535	.0559
ASE	.2007	.0830	.0820	.0631	.0540	.0486	.0470	.0459
Coverage	.9900	.9840	.9750	.9930	.9950	.9970	.9860	.9620
Orthogonal arrays, unconstrained sampling								
Mean	.3287	.0838	.0843	.0378	.0213	.0087	.0045	.0016
SD	.1740	.0592	.0604	.0327	.0215	.0146	.0123	.0113
ASE	.1615	.0466	.0468	.0255	.0181	.0128	.0111	.0100
Coverage	.8300	.7870	.7800	.8030	.8310	.8930	.9610	.9900
Orthogonal arrays, Latin hypercube sampling								
Mean	.3647	.0896	.0890	.0394	.0225	.0099	.0040	.0018
SD	.0783	.0323	.0325	.0232	.0196	.0151	.0133	.0125
ASE	.1801	.0511	.0508	.0281	.0208	.0156	.0133	.0125
Coverage	.9990	.9890	.9830	.9500	.9440	.9570	.9870	.9940

analytical values. The UPCS plan with Latin hypercube sampling of input values leads to estimates that are mildly biased but have substantially smaller standard deviations for the most important inputs (i.e., smallest values of c_i). In contrast, the estimates based on random permutation arrays, with either form of input sampling, have substantial positive bias. The use of Latin hypercube sampling reduces standard errors in this case as well, but for a given type of input sampling (either unconstrained or Latin hypercube), the precision is no worse (and in some cases is better) for UPCS plans than for plans based on

Table 4. Means and standard deviations (SD) of $\hat{\eta}_i^2$, average standard errors (ASEs), and coverage of approximate 95% confidence intervals, g-function example

i	1	2	3	4	5	6	7	8
c_i	0	1	1	2	3	5	8	13
η_i^2 (analytical)	.4890	.1223	.1223	.0543	.0306	.0136	.0060	.0025
Random arrays, unconstrained sampling								
Mean	.5427	.2154	.2160	.1560	.1307	.1178	.1126	.1091
SD	.1335	.0999	.0990	.0836	.0733	.0660	.0652	.0638
ASE	.1356	.0984	.0987	.0796	.0706	.0659	.0638	.0623
Coverage	.6620	.9020	.8950	.9070	.9130	.9020	.8820	.8840
Random arrays, Latin hypercube sampling								
Mean	.5450	.2091	.2061	.1532	.1278	.1150	.1099	.1074
SD	.0838	.0843	.0864	.0780	.0674	.0644	.0614	.0640
ASE	.1420	.0988	.0977	.0799	.0707	.0657	.0638	.0625
Coverage	.7730	.9490	.9490	.9350	.9370	.9210	.9140	.8940
Orthogonal arrays, unconstrained sampling								
Mean	.4831	.1250	.1250	.0553	.0310	.0127	.0061	.0022
SD	.1288	.0679	.0676	.0374	.0265	.0187	.0159	.0152
ASE	.1383	.0699	.0699	.0410	.0298	.0214	.0185	.0169
Coverage	.8220	.9560	.9600	.9600	.9830	.9940	1.0000	1.0000
Orthogonal arrays, Latin hypercube sampling								
Mean	.4850	.1183	.1176	.0514	.0293	.0127	.0051	.0022
SD	.0615	.0334	.0346	.0264	.0237	.0190	.0172	.0163
ASE	.1463	.0702	.0698	.0407	.0305	.0231	.0199	.0187
Coverage	.9840	1.0000	1.0000	.9940	.9900	.9930	1.0000	1.0000

randomly constructed arrays. The approximate standard errors derived in Section 2.2.1 are reasonably accurate estimates of the respective simulation standard deviations in most cases but tend to be too large (as estimates of the standard deviation) for the most important inputs when Latin hypercube sampling is used. Coverage of ± 2 standard error intervals can be substantially greater or less than .95; this observation is not surprising, because the working distributional assumptions used to derive the standard errors, although heuristically convenient, are not technically justified.

5. SOME ALTERNATIVE APPROACHES

The methods discussed in Sections 1–4 are entirely nonparametric in that they require no assumptions about the nature of y as a function of $\mathbf{x}$. Indeed, y can be nondifferentiable or even discontinuous in $\mathbf{x}$, and the arguments for estimation of θ_i remain valid. Other methods have been developed that are built at least tacitly on assumptions about $y(\mathbf{x})$, and that may provide better efficiency when these assumptions are warranted. We briefly mention a few that have been mentioned by the reviewers of this article:

- The Fourier amplitude sensitivity test (FAST), introduced by Cukier, Fortuin, Shuler, Petschek, and Schaibly (1973), method uses input vectors selected as points along a parametric curve through Δ, characterized in part by a stepsize s and a set of integer frequencies $\{\omega_1, \omega_2, \ldots, \omega_k\}$. The resulting output data support a Fourier analysis from which estimates of the θ_i can be constructed.
- Because the standard approach to selecting frequencies leads to rapidly increasing sample size with increasing k, the suggested FAST sample sizes can often be quite large. Tarantola, Gatelli, and Mara (2006) recently suggested ways of modifying the FAST sampling and analysis process using random balance designs (Satterthwaite 1959) that can reduce data requirements in some cases.
- Oakley and O'Hagan (2004) described a Bayesian approach to estimation of θ_i, as well as other indices related to model sensitivity, based on regarding $y(\mathbf{x})$ as a *random function*, a realization of a Gaussian stochastic process indexed in Δ. By specifying prior distributions for the parameters of the process and computing output values at a collection of input vectors, Bayesian arguments can be used to derive a posterior process for $y(\mathbf{x})$, and posterior distributions for functionals of y, including θ_i.
- Other methods have been developed that more explicitly rely on assumptions about the smoothness of y as a function of $\mathbf{x}$. For example, Ratto, Pagano, and Young (2007) described another methodology based on metamodels related to linear regression models and Kalman filters.

Assumptions about $y(\mathbf{x})$ are fundamental to any approach based on output modeling and are typically reflected in choices that must be made in implementation. These include, for example, the number of "harmonic" frequencies considered to be adequate in FAST, the spatial correlation functions used in methods based on spatial stochastic processes, and the particular selection of functional approximations used in regression-based methods. As is always true in statistical estimation problems, various methods can be developed based on different assumptions. Nonparametric sampling-based methods can be expected to yield valid estimates of θ_i for a wide variety of model functions, but techniques based on stronger assumptions should provide more precision when the assumptions are justified. Taken together, these methods allow an experimenter to select the approach that is best matched to the application at hand.

6. DEMONSTRATION: ENVIRONMENTAL PATHWAYS MODEL

The environmental pathways model is a mathematical model of an ecosystem. It simulates the flow of material among eight different compartments, as prescribed by a set of linear differential equations that relate concentrations in the compartments as functions of time. In this analysis, 25 transfer coefficients are regarded as model inputs and were examined to assess their effects on two scalar-valued outputs. Each of these 25 inputs was assigned an appropriate uniform distribution, reflecting the uncertainty in their respective values. The model computes material concentration as a function of time for each of the eight compartments. Interest in this study was in component C3; the output time series for this compartment are displayed in Figure 1 for the 841 model runs described later. Because the shape of the output function is similar in each case, output can be effectively reduced to an equilibrium (large time) concentration value and the time at which 99% of the equilibrium value is reached. In our analysis, we also transformed equilibrium concentration to a logarithmic scale because of the skewed distribution of values generated; the equilibrium concentrations for a relatively few runs were larger than the bulk of the values observed.

The orthogonal array used as the design for this computer experiment was an OA$(29^2, 30, 29, 2)$ constructed by the Rao–Hamming technique (e.g., Hedayat et al. 1999). One of the 30 columns served as the array index, 25 columns were used to represent the inputs, and the remaining 4 columns were discarded. As described in Section 3, the orthogonal array (or matrix $\mathbf{C}$) is the basis for up to 29 input sampling arrays, each specifying

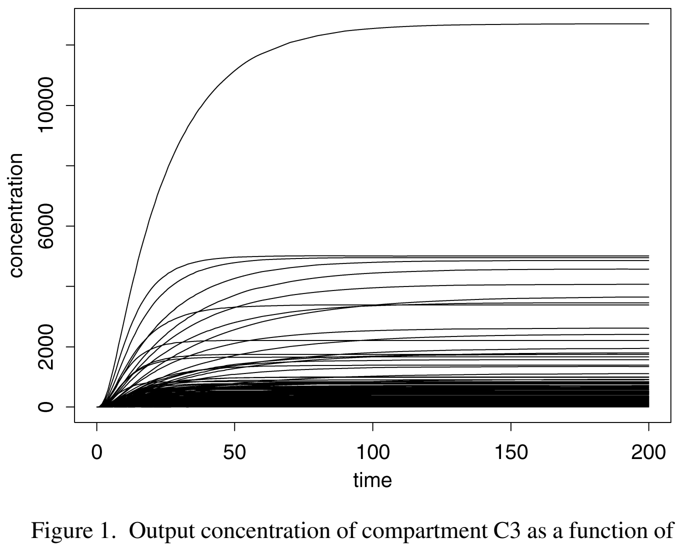

Figure 1. Output concentration of compartment C3 as a function of time, for each of 841 runs of the environmental pathways model.

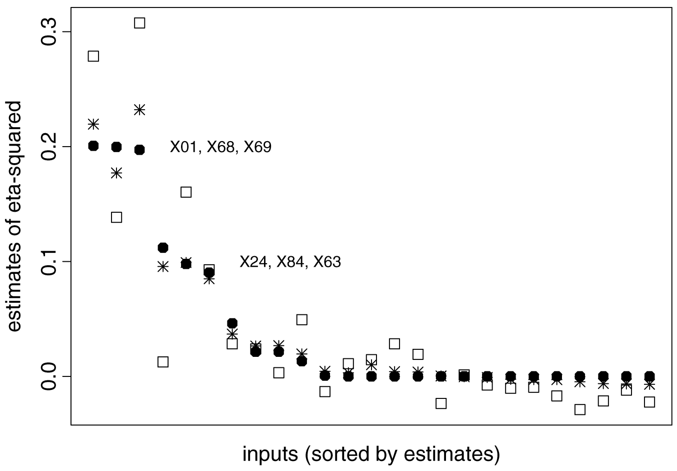

Figure 2. Estimates of η_i^2, sorted by size, for 25 inputs in the environmental pathways model, for log(equilibrium concentration). Sensitivity analyses were based on computer experiments of 841 runs ($a = 29$; ●), 580 runs ($a = 20$; ✳), and 290 runs ($a = 10$; □).

a set of 29 input vectors. Twenty-nine values were generated for each of the 25 inputs; the full collection of values for each input was represented in each array, and the orthogonal array structure ensured that no pair of values for two inputs appeared together more than once.

In this exercise, the actual values of the input vectors were generated by a popular variant of the Latin hypercube sampling technique. The uncertainty distribution for each input was partitioned into 29 equal-probability intervals as described in Section 2.2; however, the midpoints of these intervals, rather than random selections from them, were used as the input values. With as many as 29 intervals, any bias introduced by this simplification was likely to be minor, unless some input distributions had very long tails (not the case in this example).

Figures 2 and 3 display the results of sensitivity analyses for the two outputs of interest. The solid dots depict values of $\hat{\eta}_i^2$ based on an analysis of the $29^2 = 841$ model evaluations specified by the entire orthogonal array. These results suggest that equilibrium concentration in C3 was most heavily influenced by three inputs (identified as X01, X68, and X69), that three additional inputs also may be of some importance (identified as X24, X84, and X63), and that the remaining 19 inputs had relatively less or minimal influence. The results for time to 99% equilibrium concentration in C3 were much simpler; a single input (X63) clearly accounted for nearly all of the variability in this output.

The environmental pathways model used in this example does not require substantial computational effort; a single evaluation executes in less than 1 second on a modern workstation. But many important models are far more computationally intensive, and a budget of 841 evaluations often would be impractical. Recall that it is not necessary to include all 29 sampling arrays associated with the complete orthogonal array; any subset of these sampling arrays also represents a UPCS. We repeated the sensitivity analysis using only the first 10 and first 20 arrays constructed from the $\mathrm{OA}(29^2, 30, 29, 2)$, requiring computer experiments of approximately $\frac{1}{3}$ and $\frac{2}{3}$ the number of model executions of the analysis described earlier. Results for these also are displayed in Figures 2 and 3 (with squares for $a = 10$ and asterisks for $a = 20$). In this case little information was lost by using 20 arrays instead of the full 29; similar conclusions would be reached as to which inputs are most important (although the ranking of the apparently most important three inputs is not the same). With $a = 10$ arrays (290 model runs), 5 of the 6 most important inputs likely would be correctly identified, but the estimate for input X24 would not suggest an "active" input in this case. None of the apparently unimportant inputs would be incorrectly identified as being among the most important in the sensitivity analyses based on 10 or 20 arrays.

6.1 Comparisons With Other Approaches

We also carried out sensitivity analyses of the environmental pathways code using the FAST analysis of Cukier et al. (1973) and the Bayesian modeling approach of Oakley and O'Hagan (2004). As mentioned in Section 5, FAST requires data collected according to a particular form of sampling plan or experimental design; these were constructed in 291, 581, and 841 runs to correspond in size to the arrays used in demonstrating our approach. (The smaller designs were increased in size by one run to accommodate the need for an odd number of function evaluations with this method.) The frequencies used in this construction, given in Table 5, were selected so as to elimi-

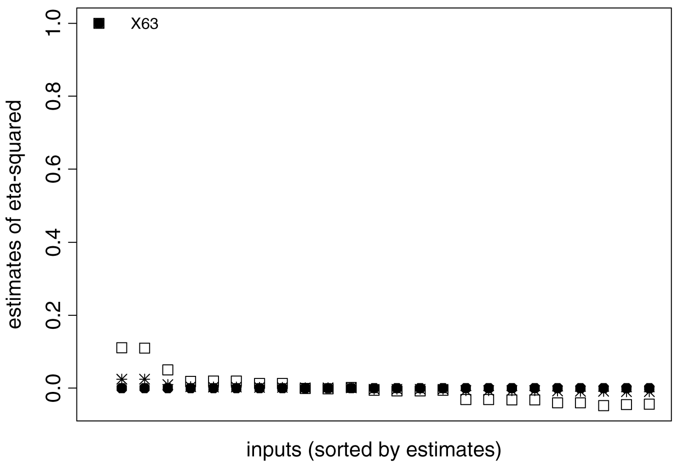

Figure 3. Estimates of η_i^2, sorted by size, for 25 inputs in the environmental pathways model, time to 99% equilibrium concentration. Sensitivity analyses were based on computer experiments of 841 runs ($a = 29$; ●), 580 runs ($a = 20$; ✳), and 290 runs ($a = 10$; □).

Table 5. Frequencies (ω) used in FAST analyses of the environmental pathways model

		291 and 581 runs		
3	5	7	11	13
17	19	23	25	27
29	31	35	37	41
43	45	47	49	53
55	59	61	63	65
		841 runs		
5	13	17	19	23
31	35	37	59	63
71	73	77	85	143
145	155	159	161	169
171	185	191	203	207

nate second-order aliasing and as much third-order aliasing as possible, although the sample sizes that we used appear to be substantially smaller than those usually suggested for FAST. We evaluated the environmental pathways model for each input vector in these designs, and used FAST analysis with these data to estimate first-order variance ratios for each input. We carried out the Oakley and O'Hagan analysis using GEP–SA software (available at *http://ctcd.group.shef.ac.uk/gem.html*), and because there is no requirement for a special experimental design with this approach, we used the output generated from our original 290-run design ($a = 10$). GEP–SA imposes an upper limit of 400 on the number of model evaluations that can be accommodated, so we made no comparisons to our 580- and 841-run exercises.

Estimates of η^2 for each input based on FAST and Bayesian modeling approaches are displayed in Figures 4 and 5; for comparison, the 25 inputs are sorted in the same order in these graphs as in Figures 2 and 3. The general conclusions concerning which inputs appear to be most important were essentially the same with each method; FAST and the Bayesian modeling methodology also identified X01, X68, and X69 as having the most important influence on equilibrium concentration and X24, X84, and X63 as also having notable effects on this output, and identified X63 as the primary "driver" of time to equilibrium. Strong agreement is not surprising in this case, because the environmental pathways model actually is a fairly simple function with fairly few influential inputs. Results are not plotted for the 291-run FAST analysis, because these were essentially identical to those for the 581-run result. Because both methods are based on substantially stronger assumptions than the sampling approach described in Sections 1–4, they appear to offer more precise results for the smallest array sizes. For more complex functional forms, this advantage can be offset by bias arising from inadequacy of the Fourier approximation assumed by FAST and the preference for simple functions embodied in the Bayesian prior.

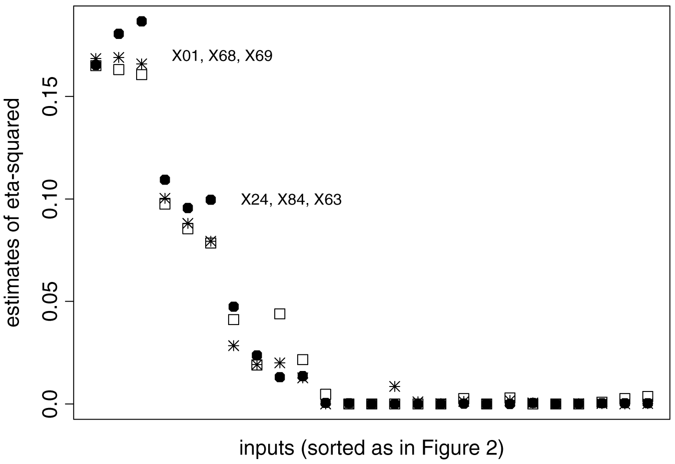

Figure 4. Estimates of η_i^2, sorted as in Figure 2, for 25 inputs in the environmental pathways model, for log(equilibrium concentration), based on FAST analyses of 581 runs (□) and 841 runs (✳), and Bayesian analysis of 290 runs (●).

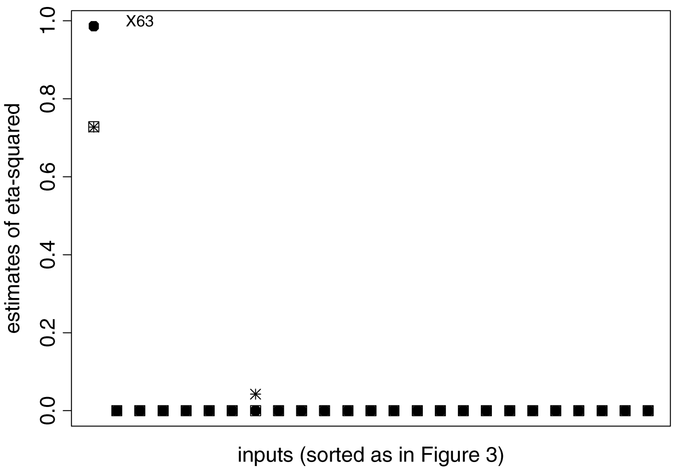

Figure 5. Estimates of η_i^2, sorted as in Figure 3, for 25 inputs in the environmental pathways model, time to 99% equilibrium concentration, based on FAST analyses of 581 runs (□) and 841 runs (✳), and Bayesian analysis of 290 runs (●).

7. CONCLUSION

As computer speed and memory capacity have increased, scientists and others who develop computer models have written increasingly complex representations of the systems that they study. In many cases, the resulting computer models, although completely defined on paper, cannot be fully understood because of this complexity, even by their creators. The techniques summarized by Saltelli, Chan and Scott (2000) have been developed to provide experimental evidence of which model inputs are most crucial in determining output values.

First-order sensitivity indices θ_i and η_i^2 are useful tools for characterizing the individual importance of model inputs. Saltelli's generalization of Sobol's substituted column plan supports estimation of both first-order and total sensitivity indices. Permuted column sampling plans, such as the replicated Latin hypercube arrays described by McKay (1995), support efficient estimation of first-order sensitivity indices. But if the column permutations are selected at random, then these estimates can be substantially biased, especially when the number of influential inputs is large or the number of runs in each sampling array is small. Like the sampling plans presented here, the *balanced replication array* sampling plans (Morris et al. 2006) also support unbiased estimation, but generally with less precision than can be attained with permuted column samples of comparable size.

In this article we have shown that UPCSs can be constructed if the column permutations are selected deterministically, and that UPCSs containing up to the largest possible number of arrays, in the smallest possible number of runs per array, are characterized by orthogonal arrays of strength 2. In practice, the size requirements can be avoided by constructing sampling plans for more than k inputs (thereby increasing the array size n; the unneeded columns can be simply ignored) or by not using the entire collection of arrays specificed by the OA. As with other permuted column sampling plans, UPCS plans can be implemented using either Latin hypercube sampling or unconstrained sampling of input values. In the demonstration exercise of Section 4, the UPCS plans also resulted in more precise estimation of first-order variance indices than did the random permutation plans. In addition, UPCS plans are more flexible than balanced replication arrays in that there are fewer restrictions on the size of sample that can be constructed.

One operational advantage of selecting column permutations randomly is that this procedure is very simple; a sampling design for any number of inputs can be easily constructed using only a random number generator. The construction of systematic sampling arrays requires more effort. The result of this article is that the considerable effort that has gone into construction techniques for orthogonal arrays (described by, e.g., Hedayat et al. 1999) and websites that contain collections of orthogonal arrays (e.g., *http://www.research.att.com/~njas/oadir/*) can be directly used to construct UPCS plans.

ACKNOWLEDGMENTS

The work of M. D. Morris was partially supported through the Visiting Faculty Program of Los Alamos National Laboratory and by National Science Foundation grant DMS 0502347 EMSW21-RTG awarded to the Department of Statistics, Iowa State University.

APPENDIX: PROOFS OF LEMMAS AND THEOREMS OF SECTION 3

To facilitate the proof of unbiasedness of $\hat{\theta}_i$ for UPCS plans, we extend the notation of Section 3. Let $S^2(\mathbf{A}_j)$ represent the sample variance of output values associated with the rows (input vectors) of array $\mathbf{A}_j$. For given $i = 1, 2, 3, \ldots, k$, define $\mathbf{B}_r$, $r = 1, 2, 3, \ldots, n$, to be the $a \times k$ array in which row j is the single row from $\mathbf{A}_j$ in which the ith element is r. Let $S^2(\mathbf{B}_r)$ represent the sample variance of output values associated with rows of array $\mathbf{B}_r$. Thus $\mathbf{B}_r$ specifies a set of input vectors that share a common value for input i, and each $S^2(\mathbf{B}_r)$ is an estimate of the output variance conditioned on this fixed value of x_i. Note that in this notation,

$$\hat{\theta}_i = \frac{1}{a}\sum_{j=1}^{a} S^2(\mathbf{a}_j) - \frac{1}{n}\sum_{r=1}^{n} S^2(\mathbf{B}_r).$$

We prove the unbiasedness of $\hat{\theta}_i$ by examining the expectation of each term.

Proof of Theorem 1

By construction, the ith column of $\mathbf{A}_j$ is a permutation of $\mathbf{u}$, and the corresponding set of input values (of x_i) is a random sample of $f_i(x_i)$. This is true independently for each column (value of $i = 1, 2, 3, \ldots, k$), and so the set of input vectors ($\mathbf{x}$) corresponding to the rows of $\mathbf{A}_j$ compose a random sample from $f(\mathbf{x}) = \prod_{i=1}^{k} f_i(x_i)$. Therefore, $E\{S^2(\mathbf{A}_j)\} = V[y(\mathbf{x})]$ and, consequently, $E\{\frac{1}{a}\sum_{j=1}^{a} S^2(\mathbf{A}_j)\} = V[y(\mathbf{x})]$.

By construction, the ith column of $\mathbf{B}_r$ is $r \times \mathbf{1}$, and every other column is a permutation of $\mathbf{u}$. For $i' \neq i$, the elements of the i'th columns of $\mathbf{B}_r$ correspond to a random sample of $f_{i'}(x_{i'})$, and so the rows of $\mathbf{B}_r$, excluding the ith element, correspond to a random sample of $f_{-i}(\mathbf{x}_{-i})$; therefore,

$$E\{S^2(\mathbf{B}_r)|x_i = x_{i,r}\} = V_{-i}[y(\mathbf{x}_{-i})|x_i = x_{i,r}],$$

where $x_{i,r}$ is the random value drawn for x_i corresponding to the coded value r. Because $\{x_{i,r}, r = 1, 2, 3, \ldots, n\}$ is a random sample from $f_i(x_i)$,

$$E\left\{\frac{1}{n}\sum_{r=1}^{n} S^2(\mathbf{B}_r)\right\} = E_i\big[V_{-i}\{y(\mathbf{x})\}\big].$$

Finally, combining these two results directly implies unbiasedness of $\hat{\theta}_i$.

Before presenting proofs of the main results, we note that these are most easily demonstrated by writing the UPCS in a "standardized" form. Without loss of generality, suppose that the UPCS is arranged so that

$$\mathbf{a}_1^1 = \mathbf{a}_2^1 = \mathbf{a}_3^1 = \cdots = \mathbf{a}_k^1 = \mathbf{a}_1^2 = \mathbf{a}_1^3 = \mathbf{a}_1^4 = \cdots = \mathbf{a}_1^a = \mathbf{u};$$

that is, the permutations used in all columns of $\mathbf{A}_1$ and the first column of each of arrays $\mathbf{A}_2$–$\mathbf{A}_a$ leave the elements of $\mathbf{u}$ in unchanged order.

Proof of Lemma 1

Consider purposeful selection of column permutations in a standardized PCS to satisfy Definitions 1 and 2 (Sec. 3). One implication of this standardization is that 1 cannot appear as the first entry of the second column of $\mathbf{A}_2$, because this would pair it with the 1 in the first column (and that pair is already represented in $\mathbf{A}_1$); say that, without loss of generality, we place it in the second row of $\mathbf{A}_2$ instead. Given this choice, a 1 cannot appear as the first or second entry in the third column of $\mathbf{A}_2$, because this would pair it with a 1 in either the first or second column; suppose that we choose to place it in the third column instead. We may continue this pattern through as many as n columns, but if $k > n$, then this leaves us with the following partially characterized array:

$$\begin{array}{c} \qquad\qquad\qquad\qquad (\text{column } n) \\ \mathbf{A}_2 = \begin{pmatrix} 1 & \text{--} & \text{--} & \cdots & \text{--} & \cdots \\ 2 & 1 & \text{--} & \cdots & \text{--} & \cdots \\ 3 & \text{--} & 1 & \cdots & \text{--} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ n & \text{--} & \text{--} & \cdots & 1 & \cdots \end{pmatrix}. \end{array}$$

But this presents a situation in which *any* ordering of 1–n in subsequent columns will produce a row containing at least two 1's, so that at least some pairs of runs, one from each of $\mathbf{A}_1$ and $\mathbf{A}_2$, will have more than one input value in common. Thus avoiding this problem requires a minimum of a k runs in each array.

Proof of Lemma 2

Consider the first two elements in the first row of each standardized array, for example,

$$\begin{array}{ccccc} \mathbf{A}_1 & \mathbf{A}_2 & \mathbf{A}_3 & \mathbf{A}_4 & \cdots \\ (1,1) & (1,\text{-}) & (1,\text{-}) & (1,\text{-}) & \cdots \end{array}$$

Note that each blank must be filled in with a different number from 2 through k; otherwise, at least two arrays would have the same entries in the first two positions of row one, and the condition would not be met. Thus the number of arrays cannot be greater than k.

Proof of Theorem 2

Sufficiency. By Definition 1, each column of $\mathbf{A}_j, j = 1, 2, 3, \dots, n$, contains each of the n symbols once, so the first column and any other columns of $\mathbf{C}$ contain each possible pair of symbols exactly once. Furthermore, if $\mathbf{A}_j, j = 1, 2, 3, \dots, n$, is a UPCS, then the requirement of Definition 2 implies that each pair of the last k columns of $\mathbf{C}$ must contain each possible pair of symbols *at most* once, and this also must be *exactly* once, because $\mathbf{C}$ contains n^2 rows. Thus sufficiency is proven.

Necessity. If $\mathbf{C}$ is an orthogonal array of strength 2 in n symbols, then each column of $\mathbf{A}_j, j = 1, 2, 3, \dots, n$, must be a permutation of $\mathbf{u}$, satisfying the requirement of Definition 1, to satisfy the balance condition with the first column of $\mathbf{C}$. Furthermore, orthogonality implies that any two of the last k columns of $\mathbf{C}$ contain each ordered pair of values exactly once, so the requirement of Definition 2 also is satisfied. Thus $\mathbf{A}_j, j = 1, 2, 3, \dots, n$, is a UPCS, and necessity is proven.

[Received December 2005. Revised February 2008.]

REFERENCES

Archer, G. E. B., Saltelli, A., and Sobol', I. M. (1997), "Sensitivity Measures, ANOVA–Like Techniques and the Use of Bootstrap," *Journal of Statistics and computer Simulation*, 58, 99–120.

Cukier, R. I., Fortuin, C. M., Shuler, K. E., Petschek, A. G., and Schaibly, J. H. (1973), "Study of the Sensitivity of Coupled Reaction Systems to Uncertainties in Rate Coefficients, I: Theory," *Journal of Chemical Physics*, 59, 3873–3878.

Hedayat, A. S., Sloane, N. J. A., and Stufken, J. (1999), *Orthogonal Arrays: Theory and Applications*, New York: Springer-Verlag.

Hoeffding, W. (1948), "A Class of Statistics With Asymptotically Normal Distribution," *The Annals of Mathematical Statistics*, 19, 293–325.

McKay, M. D. (1995), "Evaluating Prediction Uncertainty," Report NUREG/CR-6311, LA-12915-MS, Los Alamos National Laboratory.

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

Morris, M. D., Moore, L. M., and McKay, M. D. (2006), "Sampling Plans Based on Balanced Incomplete Block Designs for Evaluating the Importance of Computer Model Inputs," *Journal of Statistical Planning and Inference*, 136, 3203–3220.

Niederreiter, H., and Shiue, P. J.-S. (eds.) (1995), *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, New York: Springer-Verlag.

Oakley, J., and O'Hagan, A. (2004), "Probabilistic Sensitivity Analysis of Complex Models: A Bayesian Approach," *Journal of the Royal Statistical Society*, Ser. B, 66, 751–769.

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

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

Pearson, K. (1903), "Mathematical Contributions to the Theory of Evolution," *Proceedings of the Royal Society of London*, 71, 288–313.

Ratto, M., Pagano, A., and Young, P. (2007), "State-Dependent Parameter Metamodelling and Sensitivity Analysis," *Computer Physics Communications*, 177, 863–876.

Saltelli, A. (2002), "Making Best Use of Model Evaluations to Compute Sensitivity Indices," *Computer Physics Communications*, 145, 280–297.

Saltelli, A., and Sobol', I. M. (1995), "About the Use of Rank Transformation in Sensitivity Analysis of Model Output," *Reliability Engineering and System Safety*, 50, 225–239.

Saltelli, A., Chan, K., and Scott, M. (eds.) (2000), *Sensitivity Analysis*, New York: Wiley.

Satterthwaite, F. D. (1959), "Random Balance Experimentation," *Technometrics*, 1, 111–137.

Sobol', I. M. (1993), "Sensitivity Analysis for Nonlinear Mathematical Models," *Mathematical Modelling and Computational Experiments*, 1, 407–414.

Stein, M. L. (1987), "Large-Sample Properties of Simulations Using Latin Hypercube Sampling," *Technometrics*, 29, 143–151.

Stuart, A., and Ord, J. K. (1991), *Kendall's Advanced Theory of Statistics*, Vol. 2 (5th ed.), London: Oxford University Press.

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

Tarantola, S., Gatelli, D., and Mara, T. A. (2006), "Random Balance Designs for the Estimation of First Order Global Sensitivity Indices," *Reliability Engineering and System Safety*, 91, 717–727.