Proceedings of the 1992 Winter Simulation Conference
ed. J. J. Swain, D. Goldsman, R. C. Crain, and J. R. Wilson

DESIGNING EFFICIENT SIMULATION EXPERIMENTS

Barry L. Nelson

Department of Industrial & Systems Engineering
The Ohio State University
Columbus, Ohio 43210, U.S.A.

ABSTRACT

This tutorial describes basic principles for designing statistically efficient simulation experiments and controlling experiment error, including the use of exploratory experiments, assignment of random number seeds or streams, and analysis of the results. Simulation experiments that are performed to compare two or more systems are emphasized.

1 INTRODUCTION

Stochastic simulations are experiments that provide *estimates* of the performance parameters associated with simulated systems. The estimates are almost certainly wrong, in the sense that they do not equal the true, unknown performance parameters. For this reason every estimate should be accompanied by a measure of its potential error. The size of the error is seldom known in advance, but it could be so large that the estimates are not meaningful. Therefore, the experimenter should control or design for the error, as well as measure it.

This tutorial describes basic principles for designing statistically efficient simulation experiments and controlling experiment error. The focus is estimator error, as opposed to modeling error (the simulation model fails to represent the system of interest) or numerical error (the loss of accuracy due to the finite representation of numbers by computers). We concentrate on simulation experiments that are performed to compare the performance of two or more simulated systems (e.g., to select the best performer of k systems). More advanced references include Bratley, Fox and Schrage (1987) and Law and Kelton (1991); a more comprehensive reference at about the same level is Nelson (1992), which is the basis for part of this tutorial.

A certain amount of mathematical notation is necessary. §2 contains two examples to make the notation concrete.

Let Y be a random variable that generically represents the output (sample performance) of a simulation, and let μ be the unknown expected value of Y, $\mu \equiv \mathrm{E}[Y]$. In this tutorial μ is the performance parameter of interest. Suppose we want to compare k different systems. To avoid confusion between the real system and the model of the system represented by the simulation, and for compatibility with statistics textbooks, we call the k simulated systems *design points.*

Let

$$\mathbf{Y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_k \end{bmatrix}$$

be a $k \times 1$ vector of outputs across all k design points; that is, Y_i denotes the sample performance from design point i, and μ_i is the expected performance. We are primarily interested in comparisons of the form $\mu_i - \mu_\ell$, the expected difference in performance between design points i and ℓ. We assume that the relationship among the k design points can be approximated by the following general linear model (GLM):

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{1}$$

where $\mathbf{X}$ is $k \times p$ fixed *design matrix*, $\boldsymbol{\beta}$ is $p \times 1$ vector of unknown constants, and $\boldsymbol{\varepsilon}$ is a $k \times 1$ vector of random errors with expectation $\mathbf{0}$.

The portion of classical experiment design that addresses the specification of $\mathbf{X}$ is beyond the scope of this tutorial. We emphasize the special case when $\mathbf{X} = \mathbf{I}_{k\times k}$, the $k \times k$ identity matrix, and $\boldsymbol{\beta} = (\mu_1, \mu_2, \ldots, \mu_k)'$, which simplifies (1) to the one-way analysis of variance model

$$Y_i = \mu_i + \varepsilon_i \tag{2}$$

for $i = 1, 2, \ldots, k$. For the purpose of this tutorial *experiment design* includes the following:

1. Specifying the number of replications at each design point. We attach a second subscript, j, to Y_i or ε_i to indicate different replications; e.g., Y_{ij} denotes the output from jth replication at design point i. The number of replications at a design point is either m or n.

2. Specifying the length of each replication at each design point, when it is controllable.

3. Assigning the pseudorandom numbers to each design point.

2 EXAMPLES

There are two broad (not necessarily exhaustive) classes of system performance parameters of interest: those defined with respect to prespecified initial and final conditions for the system of interest, and those defined over a (conceptually) infinite time horizon. Simulation experiments that estimate the former are called *terminating* simulation experiments, while the latter are called *steady-state* simulation experiments.

The experiment design for terminating simulations always calls for multiple replications, and the length of each replication is determined by the prespecified initial and final conditions. The experiment design for steady-state simulation may call for one or more replications, and the length of each replications is a design decision. Consider the following examples (the first example is based on Nelson (1992)):

Example 1, terminating simulation: A small city allows auto owners to renew their license plates by mail, with each owner's renewal taking place during the month of their birth. The mail-in renewal applications are processed by a clerk. The rate of receipt of applications increases steadily throughout the month, but the load during all months is about the same. Mail-in renewals that are postmarked after the 27th day of the month are returned to the applicant without being processed. The city is interested in determining the performance level the clerk must attain, in terms of the expected time to process a renewal, to prevent excessive processing delays. Uncertainty in the system is due to the arrival dates of renewal applications and the actual processing time for each application. These input processes are described by probability distributions based on data from a similar city.

The simulation experiment will evaluate the expected average delay in processing applications during a month. Thus, μ_i is the expected average delay per month when the expected processing time per renewal is x_i minutes, each month initially having no applications to process and ending after processing the last application received on the 27th. Let Y_{ij} be the observed average delay in processing the applications received during the jth month when the expected processing time is x_i minutes. A functional relationship among the design points is assumed, namely

$$Y_{ij} = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \varepsilon_{ij}$$

implying that $\mu_i = \mathrm{E}[Y_{ij}] = \beta_0 + \beta_1 x_i + \beta_2 x_i^2$. The design matrix for a single replication from each system is

$$\mathbf{X} = \begin{bmatrix} 1 & 10 & 100 \\ 1 & 15 & 225 \\ 1 & 20 & 400 \end{bmatrix}$$

which covers expected processing times ranging from 10 to 20 minutes. The design decision is the number of replications at each of the $k = 3$ design points.

Example 2, steady-state simulation: A company that provides an on-line data-base service wants to evaluate five proposed computer architectures in terms of their effect on the expected response time to user requests for information (a shorter response time is better). The five architectures constitute $k = 5$ design points. Uncertainty in the system is due to user behavior: the actual log-on times of users, the lengths of user's sessions, and the types of queries users make. These input processes are described by probability distributions derived from data on the present computer system.

The simulation experiment will evaluate the expected response time over a prolonged period of peak load. Thus, μ_i is the steady-state expected response time for design point i under peak-load conditions. No functional relationship among the design points is assumed, so the one-way model (2) is appropriate to describe the simulation output data, Y_{ij}, the observed average response time for design point i on replication j. Design decisions include the length of each replication (how long the system is simulated at peak load) and the number of replications. Given some constraint on computing budget or time available for experiments, the trade off is between many short replications versus a few long ones.

3 RANDOMNESS IN SIMULATION

Uncertainty ("randomness") in a simulation experiment is derived from the pseudorandom numbers, typically numbers in the interval $(0, 1)$, that are difficult to distinguish from independent and identically distributed (i.i.d.), uniformly distributed random numbers. A useful way to think about the random numbers is as a large, ordered table, where the number of entries in the table is often around $2^{31} \approx 2 \times 10^9$ (small tables such as this appear in the back of some statistics textbooks). Given a starting point in the table, a simulation uses the pseudorandom numbers in order until the experiment is completed. If the end of the table is encountered, then numbers starting from the beginning of the table are used. Most simulation languages do not actually store the pseudorandom numbers in a table (they are generated by a recursive function as needed), but the table of random numbers is a good physical representation of how the simulation language works.

Although the (conceptual) table of pseudorandom numbers is ordered, the order does not matter. As long as the numbers are used without replacement, they can be taken in any manner or starting from any position in the table and still appear to be a sample of i.i.d. random numbers. An important feature of most simulation languages is that they permit control of the pseudorandom numbers through *seeds* or *streams*. The seeds or streams are nothing more than different starting points in the table, typically spaced far apart. For example, stream 1 might correspond to entering the table at the 121,567th random number.

The assignment of random number seeds or streams is part of the design of a simulation experiment. All subsequences within the (conceptual) table appear to be i.i.d. random numbers, so assigning a different seed or stream to different design points guarantees that the outputs from different design points will be statistically independent. Similarly, assigning the same seed or stream to different design points induces dependence among the corresponding outputs, since they all have the same source of randomness. Controlling the dependence between design points is the primary reason for the existence of seeds or streams.

Many textbook experiment designs (specifications of $\mathbf{X}$ in (1)) assume independent results across design points. To conform to the assumptions of these designs we must assign different seeds or streams to each design point. Later we argue that, in many cases, we do not want independence across design points. In fact, we want the outputs across design points to be as strongly positively dependent as possible.

Two closing comments about seeds and streams: Assigning different seeds or streams to different design points does not guarantee independence if, say, we assign stream 1 to a design point, but use so many random numbers that we begin taking numbers from stream 2, which is assigned to another design point (remember that the streams are just starting points in an ordered table). If independence is critical, it is worthwhile to know the spacing between seeds or streams in a simulation language, and to make a rough estimate of the number of pseudorandom numbers needed at each design point.

On the other hand, it is typically not necessary to assign different seeds or streams to different replications at a single design point in order to obtain independent replications. Nearly all simulation languages begin subsequent replications using random numbers from where the previous replication finished, implying that different replications use different random numbers and are therefore independent.

4 EXPERIMENT PLANNING

Designing a simulation experiment that controls estimator error requires obtaining some estimate of that error. In most cases this means that a simulation experiment must be performed in (at least) two stages: An *exploratory experiment* and a *designed experiment*. The purpose of the exploratory experiment is to provide information for planning the designed experiment, which in turn provides the comparisons of interest. There is a vast literature on formal methods for multiple-stage experiments; the ideas presented here are statistically sound, but informal.

The primary reason to perform the exploratory experiment is to assess the potential error in the simulation estimates, and to use that assessment to determine the number, and possibly the length, of the replications in the designed experiment to reduce the error below an acceptable level; this topic is discussed in §4.1 and §4.3. The exploratory experiment may also be used as a preliminary check on approximations—such as normality of the output data and equality of the output variance across design points—that are common in statistical-analysis procedures; this topic is discussed in §4.2.

4.1 Number of Replications

We focus on determining the number of replications needed, assuming that a multiple-replication design will be used, and discuss the problem of determining the length of the replications in §4.3.

Let m denote the number of replications in an exploratory experiment at a design point. The value of m should strike a balance between obtaining useful information and not expending a significant portion of the available time or money for experiments; it must be at least 2, and 10 or more if possible. At least two design points, say i and ℓ, should be considered to insure that system behavior is not markedly different at different design points.

From the m replications, compute a point estimate of the expected difference in performance between the two design points, $\mu_i - \mu_\ell$. Let $\widehat{\mu_i - \mu_\ell}$ denote the estimator. Also compute an estimate of the *absolute error* of $\widehat{\mu_i - \mu_\ell}$; the absolute error estimator, denoted $\widehat{a}(m)$, is typically of the form

$$\widehat{a}(m) \equiv c(m)\frac{\widehat{\sigma}(m)}{\sqrt{m}}$$

where $c(m)$ is a constant that may depend on m and $\widehat{\sigma}(m)$ is an estimator of the *standard deviation* of the difference between design points i and ℓ, denoted σ. An example is given below. Notice that σ is a property of the simulation and not a function of m; we exploit this fact to control error.

In some contexts it is desirable to control the *relative error* of the estimates rather than the absolute error. The relative error of $\widehat{\mu_i - \mu_\ell}$ is the absolute error as a fraction of $\mu_i - \mu_\ell$; it can be estimated by $\widehat{r}(m) \equiv \widehat{a}(m)/|\widehat{\mu_i - \mu_\ell}|$.

The error estimator $\widehat{a}(m)$ or $\widehat{r}(m)$ can be used to approximate the error that will be obtained at $n > m$ replications by assuming that the estimates of σ and $\mu_i - \mu_\ell$ will not change much, and solving for the n required to obtain a prespecified level of error. To obtain an absolute error less than a, set

$$n = \left\lfloor \left(\frac{c(m)\widehat{\sigma}(m)}{a}\right)^2 \right\rfloor + 1.$$

To obtain a relative error less than r, set

$$n = \left\lfloor \left(\frac{c(m)\widehat{\sigma}(m)}{r \cdot |\widehat{\mu_i - \mu_\ell}|}\right)^2 \right\rfloor + 1.$$

The designed experiment then specifies $n - m$ additional replications for each design point considered in the exploratory experiment, and n replications for design points not considered in the exploratory experiment or for all design points if it is more convenient to generate new outputs rather than reuse outputs from the exploratory experiment. If the error actually realized in the designed experiment is too far from the target, then a new design can be constructed in the same manner.

Consider example 2, and suppose that the expected response time for computer architecture i, μ_i, is estimated by the sample mean $\bar{Y}_i = \sum_{j=1}^{m} Y_{ij}$, so that $\widehat{\mu_i - \mu_\ell} = \bar{Y}_i - \bar{Y}_\ell$. One might want to estimate the expected difference in response time to within an absolute error of, say, 40 milliseconds, or within a relative error of, say, 0.05 (5% of the true difference).

Let

$$S_i^2(m) = \frac{1}{m-1}\sum_{j=1}^{m}(Y_{ij} - \bar{Y}_i)^2 \qquad (3)$$

be the sample variance across replications at design point i. Then, if the design points are simulated independently (i.e., different random number seeds or streams for each architecture), then $\widehat{\sigma}(m) = \sqrt{S_i^2(m) + S_\ell^2(m)}$. A measure of the absolute error is the confidence-interval width $\widehat{a}(m) = t_{1-\alpha/2,2m-2}\,\widehat{\sigma}(m)/\sqrt{m}$, where $t_{1-\alpha/2,2m-2}$ is the $1 - \alpha/2$ quantile of the t distribution with $2m - 2$ degrees of freedom. Notice that $\widehat{\sigma}(m)$ is a measure of the standard deviation of the *difference* between design points i and ℓ, which is the deviation of interest when making comparisons. Later we describe how to modify this estimate if common seeds or streams are employed across design points.

4.2 Data Aggregation (Batching)

At design point i, suppose we have n replications providing outputs $Y_{i1}, Y_{i2}, \ldots, Y_{in}$. The following aggregation of the data is sometimes useful: Set

$$\bar{Y}_{ih} = \frac{1}{b}\sum_{j=1}^{b} Y_{(h-1)b+j}$$

for $h = 1, 2, \ldots, g$, so that $n = bg$. Simply stated, $\bar{Y}_{ih}$ is the sample mean of the "batch" of outputs $Y_{i,(h-1)b+1}, \ldots, Y_{i,hb}$. There are several reasons to consider basing statistical analysis on the batch means $\bar{Y}_{i1}, \ldots, \bar{Y}_{ig}$ rather than the original outputs $Y_{i1}, \ldots, Y_{in}$:

- The batch means tend to be more nearly normally distributed than the original outputs, since they are averages. This property is useful if the statistical analysis will be based on normal-distribution theory.

- By using different batch sizes, b, at different design points, the variances of the batch means across design points can be made more nearly equal than the variances of the original outputs; equal variances is a standard assumption behind many statistical procedures used for comparisons.

  For example, suppose that $S_i^2(n)$ and $S_\ell^2(n)$ are the sample variances at design points i and ℓ of the original outputs as defined in (3). If $S_i^2(n) > S_\ell^2(n)$, then a batch size of $b \approx S_i^2(n)/S_\ell^2(n)$ for design point i, and a batch size 1 for design point ℓ, will cause the batch means from i and ℓ to have approximately equal variance. Of course, the same will be true for batch sizes cb and c, respectively, for any positive integer c.

- Saving all of the batch means may be possible when it is impossible or inconvenient to save all of the original outputs. The original outputs are often aggregated into only a sample mean and variance. Yet it is sometimes useful to have the "raw" data available, not just summary statistics.

Surprisingly, it not necessary to maintain a large number of batch means, g, even if the number of replications, n, is large. One way to see this is to look at the critical values of the t distribution: $t_{0.975,30} = 2.04$, while $t_{0.975,\infty} = 1.96$, a very small difference. Therefore, batching a large number of replications into 30 batch means will have little effect on statistical analysis based on the t distribution, but it may be much easier to save 30 batch means as opposed to the all of the replication results.

If the number of replications in the exploratory experiment, m, is large enough, then the relative batch sizes can be determined during the exploratory experiment by examining the variances of the outputs. The outputs in the designed experiment can then be aggregated as they are collected. Even if inequality of variances is not an issue, the outputs from the designed experiment can be batched as long as the number of batches does not become too small.

4.3 Length of the Replications

Replication length is an issue in steady-state simulation due to the presence of initial-condition bias. The detection and elimination of initial-condition bias is outside the scope of this tutorial; see Law and Kelton (1991). However, the decision to perform a single or a multiple-replication experiment is based on the following judgement: If the initial-condition bias at each design point is well understood, and the simulation effort required to eliminate it is very small relative to the time or money available for experiments, then a multiple-replication design is preferred. Otherwise a single-replication should be used.

If a multiple-replication design is employed, then the replication length should be much longer than the initial-transient period (say 20 times longer to be concrete). The number of replications can be determined as described in §4.1.

If a single-replication design is employed, then the length of the replication is used to control the estimator error. The tutorial by Goldsman (1992) elsewhere in this volume describes methods for error estimation in steady-state simulation. One of those methods is batching, as described in §4.2, except that the batch means are formed from outputs *within* a single replication, rather than across multiple replications. The central idea is to make the batch size large enough so that the batch means are nearly independent, and can therefore be treated as independent replications. All of the guidelines in §4.1–§4.2 then apply to the batch means.

5 COMMON RANDOM NUMBERS

In this section we argue that it is often useful to assign the same random number seeds or streams to all of the design points; this technique is called *common random numbers* (CRN). The presentation will be in terms of a multiple-replication experiment, but the ideas apply to a single-replication experiment when batch means play the role of replications.

The intuition behind CRN is that a fairer comparison among design points is achieved if the design points are subjected to the same experimental conditions, specifically the same source of randomness. For instance, in example 2 it seems fair to compare the five computer architectures under the same user load (log-on times, length of sessions, and types of queries). CRN can insure this.

The mathematical justification for CRN is as follows: Suppose, in example 2, that the sample mean response time, $\bar{Y}_i$, is used to estimate the unknown expected response time, μ_i. Then for design points i and ℓ, the (unknown) standard deviation σ is

$$\begin{aligned}\sigma &= \sqrt{\mathrm{Var}[\bar{Y}_i - \bar{Y}_\ell]} \\ &= \sqrt{\mathrm{Var}[\bar{Y}_i] + \mathrm{Var}[\bar{Y}_\ell] - 2\mathrm{Cov}[\bar{Y}_i, \bar{Y}_\ell]}\end{aligned}$$

where Var denotes variance and Cov denotes covariance. If different seeds or streams are assigned to design points i and ℓ, then $\mathrm{Cov}[\bar{Y}_i, \bar{Y}_\ell] = 0$; if common seeds or streams are assigned then frequently $\mathrm{Cov}[\bar{Y}_i, \bar{Y}_\ell] > 0$, reducing σ.

Perhaps it is less obvious how example 1—in which we fit a GLM (1) rather than directly estimating expected performance—will benefit from CRN. For ease of explanation, suppose we fit the simplier model $Y_{ij} = \beta_0 + \beta_1 x_i + \varepsilon_{ij}$ and obtain estimates $\widehat{\beta}_0$ and $\widehat{\beta}_1$. Then an estimator of the expected difference in performance between design points i and ℓ is

$$\widehat{\mu_i - \mu_\ell} = \widehat{\beta}_0 + \widehat{\beta}_1 x_i - (\widehat{\beta}_0 + \widehat{\beta}_1 x_\ell) = \widehat{\beta}_1(x_i - x_\ell).$$

Since x_i and x_ℓ are fixed design points, $\widehat{\beta}_1$ determines the estimated difference between design points i and ℓ, or for that matter any other two values of x. Therefore, CRN can be expected to reduce the variance of $\widehat{\beta}_1$, and more generally reduce the variance of all of the slope terms in the GLM; it does not typically reduce the variance of the intercept term, $\widehat{\beta}_0$.

The effect of CRN can be enhanced by *synchronizing* the random numbers, which means forcing the random numbers to be used for the same purpose at each design point. The primary technique for achieving synchronization is to assign a different seed or stream to each random input process, and then to use the same collection of seeds or streams across all design points. In example 1, this means assigning a stream to the application arrival process and a different stream to the application processing process. When common streams are used across design points, the same random numbers will generate arrivals and processing at each one.

Sometimes synchronization is facilitated by generating entity/transaction attributes at the time the entity/transaction is created. For instance, in example 2 the entire sequence of queries that a user makes could be generated when the user logs on, rather than generating each new query as the previous query is completed.

One must also take care to synchronize the random numbers across replications. To be specific, replication 2 of design points i and ℓ should both begin with the same random numbers. This may not happen automatically, since replication 1 of design point i may require a different quantity of random numbers than replication 1 of design point ℓ. The best way to insure that all replications across all design points begin with the same random numbers is simulation language dependent. If a large number of seeds or streams can be created, then one approach is to assign different seeds or streams to each replication.

The exploratory experiment can be used to verify that CRN is having the desired effect by estimating the covariance between design points using

$$C_{i\ell}(m) \equiv \frac{1}{m-1}\sum_{j=1}^{m}(Y_{ij}-\bar{Y}_i)(Y_{\ell j}-\bar{Y}_\ell). \qquad (4)$$

The covariance terms should be positive; if they are negative then CRN may inflate variance and should not be used. The estimate of σ used for planning the designed experiment, $\widehat{\sigma}(m)$, should also reflect the use of CRN; see §6 below.

6 OUTPUT ANALYSIS

Analysis of the exploratory experiment provides planning information for the designed experiment. Analysis of the designed experiment provides the comparisons of interest. In this section we concentrate on output analysis under CRN. When the design points are simulated independently, standard statistical methods apply. See, for example, Law and Kelton (1991), and the tutorial by Goldsman (1992) elsewhere in this volume. The presentation is in terms of the designed experiment, but all of the output-analysis procedures can also be used with the exploratory experiment.

Suppose we make n replications at each design point. If $\mathbf{X}$ is the design matrix for one replication across all k design points, then the design matrix for all n replications is the $kn \times p$ matrix

$$\vec{\mathbf{X}} = \begin{bmatrix} \mathbf{X} \\ \mathbf{X} \\ \vdots \\ \mathbf{X} \end{bmatrix}.$$

Organize the outputs across all n replications into the $kn \times 1$ vector

$$\vec{\mathbf{Y}} = (Y_{11}, Y_{21}, \ldots, Y_{k1}, \ldots, Y_{1n}, Y_{2n}, \ldots, Y_{kn})'.$$

Then a point estimator for β is the ordinary-least-squares (OLS) estimator

$$\widehat{\beta} = \left(\vec{\mathbf{X}}'\vec{\mathbf{X}}\right)^{-1}\vec{\mathbf{X}}'\vec{\mathbf{Y}}$$

which reduces to $\widehat{\beta} = (\bar{Y}_1, \bar{Y}_2, \ldots, \bar{Y}_k)'$ when the one-way model (2) applies ($\mathbf{X} = \mathbf{I}_{k\times k}$).

The OLS point estimator is appropriate whether or not we use CRN, but the associated statistical analysis is affected by CRN. Unfortunately, statistical analysis under CRN is still an open problem. In the next two subsection we discuss exact and approximate methods.

6.1 Analysis for the One-Way Model

When there is no functional relationship among the design points, as in example 2, the one-way model (2) is appropriate, and the natural estimator for the expected difference in response time between design points i and ℓ, $\mu_i - \mu_\ell$, is $\bar{Y}_i - \bar{Y}_\ell$. Based on n replications, the standard deviation of the difference is estimated by

$$\widehat{\sigma}(n) = \sqrt{S_i^2(n) + S_\ell^2(n) - 2C_{i\ell}(n)}$$

where $S_i^2(n)$ is the sample variance from system i as defined in (3) and $C_{i\ell}(n)$ is the sample covariance as defined in (4). If the output data are nearly normally distributed, then a $(1-\alpha)100\%$ confidence interval for $\mu_i - \mu_\ell$ is

$$\bar{Y}_i - \bar{Y}_\ell \pm t_{1-\alpha/2,n-1}\frac{\widehat{\sigma}(n)}{\sqrt{n}}.$$

This confidence interval can also be used for experiment planning in the presence of CRN.

Difficulties arise in extending the analysis above to all $d = k(k-1)/2$ differences $\mu_i - \mu_\ell$, for all $i \neq \ell$, simultaneously. A standard approach is to form each confidence interval at level $1-\alpha/d$, rather than $1-\alpha$, which guarantees that the overall confidence level for all d intervals is at least $1-\alpha$ by the Bonferroni inequality. Unfortunately, the benefit we hope to obtain from CRN is shorter confidence intervals, and this procedure is so conservative that it may overwhelm the benefits of CRN, especially if k is large.

An approximate procedure, that the author has shown to be robust, is to form the following set of confidence intervals

$$\bar{Y}_i - \bar{Y}_\ell \pm q^{1-\alpha}_{k,(k-1)(n-1)} \frac{\tilde{\sigma}(n)}{\sqrt{n}}$$

for all $i \neq \ell$, where $q^{1-\alpha}_{k,(k-1)(n-1)}$ is the $1-\alpha$ quantile of the Studentized range distribution with parameter k and $(k-1)(n-1)$ degrees of freedom (Hochberg and Tamhane 1987), and $\tilde{\sigma}(n)$ is the modified standard deviation defined by

$$\tilde{\sigma}^2(n) = \sum_{i=1}^{k}\sum_{j=1}^{n} \frac{\left(Y_{ij} - \bar{Y}_i - \widehat{Y}_j + \bar{\bar{Y}}\right)^2}{(k-1)(n-1)}.$$

Here $\widehat{Y}_j = \sum_{i=1}^{k} Y_{ij}/k$, the sample mean across all design points on replication j, and $\bar{\bar{Y}} = \sum_{i=1}^{k}\sum_{j=1}^{n} Y_{ij}/(kn)$, the sample mean of all the outputs. These confidence intervals can also be used for experiment planning.

6.2 Analysis for the GLM

Consider estimating the GLM proposed in example 1. Many procedures exist for analysis of a GLM under CRN, but they are complex and depend on a host of conditions. Here we present a simple procedure proposed by Kleijnen (1988). This subsection assumes that the reader is familiar with standard analysis techniques for the GLM when the design points are independent.

Suppose we make n replications at each design point. Let $\widehat{\Sigma}_Y(n)$ be the $k \times k$ matrix with i,jth element $S_i^2(n)$, for $i = j$, and $C_{ij}(n)$, for $i \neq j$. In other words, $\widehat{\Sigma}_Y(n)$ is an estimtor of the variance-covariance matrix of $\mathbf{Y}$ based on n replications. Kleijnen proposes estimating the variance-covariance matrix of $\widehat{\beta}$ by

$$\widehat{\Sigma}_{\widehat{\beta}}(n) = \frac{1}{n}\left(\vec{\mathbf{X}}'\vec{\mathbf{X}}\right)^{-1}\vec{\mathbf{X}}'\left(\widehat{\Sigma}_Y(n)\right)\vec{\mathbf{X}}\left(\vec{\mathbf{X}}'\vec{\mathbf{X}}\right)^{-1}.$$

Planning and analysis are based on the elements of $\widehat{\Sigma}_{\widehat{\beta}}(n)$. For example, an approximate $(1-\alpha)100\%$ confidence interval for β_ℓ is

$$\widehat{\beta}_\ell \pm t_{1-\alpha/2,n-1}\,\widehat{\sigma}_\ell(n)$$

where $\widehat{\sigma}_\ell(n)$ is the square root of the ℓth diagonal element of $\widehat{\Sigma}_{\widehat{\beta}}(n)$.

ACKNOWLEDGEMENT

This work was partially supported by NSF Grant DDM-8922721.

REFERENCES

Bratley, P., B. L. Fox and L. E. Schrage. 1987. *A guide to simulation*. Second edition. New York: Springer-Verlag

Goldsman, D. 1992. Simulation output analysis. In *Proceedings of the 1992 Winter Simulation Conference*, ed. J. J. Swain, D. Goldsman R. C. Crain, and J. R. Wilson, in press. Institute of Electrical and Electronics Engineers, Washington, D.C.

Hochberg, Y. and A. C. Tamhane. 1987. *Multiple comparison procedures*. New York: John Wiley.

Kleijnen, J. P. C. 1988. Analyzing simulation experiments with common random numbers. *Management Science* 34: 65–74.

Law, A. M. and W. D. Kelton. 1991. *Simulation modeling & analysis*. New York: McGraw-Hill.

Nelson, B. L. 1992. Statistical analysis of simulation results. In *Handbook of industrial engineering* (G. Salvendy, ed.). New York: John Wiley.

AUTHOR BIOGRAPHY

BARRY L. NELSON is an Associate Professor in the Department of Industrial and Systems Engineering at The Ohio State University. His research interests are experiment design and analysis of stochastic simulations. He is President of the TIMS College on Simulation and an Associate Editor for *Operations Research*.