Comput. Opns. Res. Vol. 15, No. 5, pp. 447–456, 1988
Printed in Great Britain.

0305-0548/88 $3.00 + 0.00

AUTOMATED ESTIMATION AND VARIANCE REDUCTION VIA CONTROL VARIATES FOR INFINITE-HORIZON SIMULATIONS

ROWENA AÑONUEVO* and BARRY L. NELSON†
Department of Industrial and Systems Engineering, The Ohio State University, Columbus, OH 43210, U.S.A.

(*Received August* 1987; *revised December* 1987)

Scope and Purpose—Computer simulation is a general-purpose method for analyzing systems that are subject to uncertainty. The price of generality is that simulation-based estimators are just that: estimators of system performance that are subject to sampling variability. Variance reduction techniques (VRTs) have long been advocated for mitigating the effects of sampling variability, but the difficulty inherent in selecting and applying an appropriate VRT has limited their use to statistically knowledgeable practitioners. This paper describes an automated variance reduction and output analysis procedure that interfaces with a well-known simulation language; thus, the procedure opens the way for widespread application of VRTs. The procedure is specifically designed for estimating long-run performance measures, an area that has received little attention in the simulation literature.

Abstract—We present an automated procedure that interfaces with SIMSCRIPT II.5 simulation experiments to derive point and interval estimators for infinite-horizon parameters of stochastic simulations. The procedure combines the nonoverlapping batch means method of output analysis and the control variates variance reduction technique. Batch size and control variates are selected automatically. The paper emphasizes methodology issues and experimental evaluation rather than the specific software developed.

INTRODUCTION

Variance reduction techniques (VRTs) are used to reduce the population variance of estimators from stochastic simulation experiments; see Nelson and Schmeiser [11] for a review of well-known VRTs. VRTs improve the efficiency of a simulation experiment by reducing the cost of achieving an estimator with a specified level of precision, or by increasing the precision of an estimator for a fixed cost. This paper addresses two practical obstacles to the routine application of variance reduction in general simulation experiments: the difficulty of selecting and implementing a VRT, and the lack of VRTs for infinite-horizon simulation experiments.

Nelson [1] gives an algorithm for selecting VRTs for general simulation experiments. However, implementing the chosen technique in a particular experiment may still be a formidable task. Detailed knowledge of variance reduction is frequently required to ensure that the VRT is applied appropriately. Unfortunately, a VRT applied inappropriately may even increase variance. Automation of variance reduction in simulation languages is probably the only hope for widespread application of VRTs. Ideally an automated procedure would select and implement the most effective VRT for the particular experiment at hand; our approach here is more modest: we automate the use of a versatile VRT called control variates that is applicable in all stochastic simulation experiments (see [2] for a general characterization of control variate estimators). While this approach may not yield the greatest possible variance reduction, the computational savings over multiple simulation projects should be significant. This is different from the approach typically taken in variance reduction research in which dramatic variance reductions are achieved by restricting attention to a narrow class of simulation problems.

Most VRTs, including control variates, are designed for finite-horizon (sometimes called

*Rowena Añonuevo obtained her Master's degree in Industrial and Systems Engineering at The Ohio State University. She has a B.S. in mathematics from the University of the Philippines, and was employed by the Operations Research Department of the San Miguel Corporation from 1980 to 1984. Her graduate studies emphasized simulation of stochastic systems. She is currently employed by the Operations Research Group at American Airlines in Arlington, Texas.

†Barry L. Nelson is an Assistant Professor in the Department of Industrial and Systems Engineering at The Ohio State University. His Ph.D. is from the School of Industrial Engineering at Purdue University, and his research interests center on the design and analysis of computer simulation experiments, particularly methods for statistically efficient simulation. Dr Nelson is an active member of the TIMS College on Simulation, and is editor of its newsletter.

"transient" or "terminating") simulation experiments. In such experiments the natural experimental design is to generate independent and identically distributed (i.i.d.) realizations of the process of interest. However, for infinite-horizon (sometimes called "steady-state") simulations it may be more practical to use a single long realization because of the need to delete an initial-transient period on each realization. The initial-transient problem has lead to the development of point and interval estimation methods for the steady-state mean of a process from a single realization. The primary advantage of these methods is that more of the simulation budget can be allocated to generating usable outputs, since only one initial transient period must be deleted. Nonoverlapping batch means (see, for instance, Schmeiser [3]) is one of these methods.

In this paper we present an automated procedure called BMCV that combines the batch means method of output analysis and the control variates VRT for infinite-horizon simulation experiments. The procedure's design is based on results in Nelson [4]; some preliminary work on the procedure is reported in Añonuevo and Nelson [5]. We emphasize the issues involved in designing and implementing such a procedure and the procedure's performance in experiments, rather than the particular software developed.

The paper is organized as follows: the next section states the infinite-horizon simulation problem and reviews the batch means method and the control variates VRT. The two sections that follow present the issues addressed in the design of BMCV and an outline of procedure BMCV, respectively. We end the paper by reporting the experimental evaluation of BMCV and by discussing possible extensions of the procedure.

BACKGROUND

One statement of the infinite-horizon simulation problem is as follows: obtain a point and interval estimate of a scalar parameter $\theta = \lim_{i\to\infty} E[Y_i]$ from a finite-length output sequence $\{Y_1, Y_2, \ldots, Y_n\}$, where $\{Y_i\}$ is a sequence of random variables that may be neither independent nor identically distributed. An example of an infinite-horizon parameter is the limiting expected customer delay in a queueing system. The usual reason that the outputs $\{Y_i\}$ are not identically distributed is that the simulation is initialized in a fixed state rather than in a state sampled from the (unknown) limiting state distribution. A common approach is to delete or discard a portion of the beginning of the output series, say $\{Y_1, \ldots, Y_d\}$, where $d < n$. We do not address specific solutions to the initial-transient problem in this paper.

The remainder of this section, which is based on [4], reviews the theoretical basis for the batch means method and the control variates VRT assuming that the effects of the initial transient have somehow been removed. Let the relevant output of the simulation experiment after the initial transient be represented by a sequence of identically distributed random (column) vectors $Z_i' = [Y_i, X_{1i}, X_{2i}, \ldots, X_{qi}]$, $i = 1, 2, \ldots, n$, where $'$ denotes transpose. Let $E[Z_i'] = [\theta, \mu_1, \mu_2, \ldots, \mu_q]$ and $\mathrm{Cov}[Z_i] = \Sigma$ where

$$\Sigma = \begin{bmatrix} \sigma_y^2 & \sigma_{yx}' \\ \sigma_{yx} & \Sigma_x \end{bmatrix}$$

so that X_{ji} is the scalar $\mathrm{Var}[Y_i]$, Σ_x is the $q \times q$ matrix of $\mathrm{Cov}[X_{ji}, X_{mi}]$, $j, m = 1, 2, \ldots, q$, and σ_{yx} is the $q \times 1$ vector of $\mathrm{Cov}[Y_i, X_{ji}]$, $j = 1, 2, \ldots, q$. Thus, the square of the multiple correlation coefficient of Y_i on $[X_{1i}, X_{2i}, \ldots, X_{qi}]$ is

$$R_{yx}^2 = \frac{\sigma_{yx}' \Sigma_x^{-1} \sigma_{yx}}{\sigma_y^2}.$$

For our purposes, θ is the unknown parameter of interest and $X_{1i}, X_{2i}, \ldots, X_{qi}$ are the q control variates To be useful as a control variate, X_{ji} must be correlated with Y_i and $\mu_j = E[X_{ji}]$ must be known. For later convenience, we define the column vector

$$(X_i - \mu)' = [X_{1i} - \mu_1, X_{2i} - \mu_2, \ldots, X_{qi} - \mu_q],$$

which has expectation $[0, 0, \ldots, 0]'$ and covariance matrix Σ_x. Note that our convention for random variables is to use single subscripts to denote column vectors and double subscripts to denote scalar elements, with the exception of Y_i which is a scalar random variable.

The idea behind batch means is to transform the n dependent vectors $Z_1, Z_2, \ldots, Z_n$ into fewer (almost) independent and (almost) normally distributed batch vectors

$$\bar{Z}_j = b^{-1} \sum_{i=(j-1)b+1}^{jb} Z_i$$

for $j = 1, 2, \ldots, k$; $b = n/k$ is called the batch size, and k the number of batches. Here vector addition is component-by-component. We use the convention that any random variable with a bar and argument k is a batch mean of $b = n/k$ observations; e.g. $\bar{Y}_j(k)$ is the jth batch mean of the Y_i with batch size $b = n/k$. We assume for now that the total sampling budget n is fixed.

Given k batch means, the control variate estimator of θ is

$$\hat{\theta}(k, q) = \bar{Y} - \hat{\beta}(k, q)'(\bar{X} - \mu) \tag{1}$$

where

$$\bar{Y} = k^{-1} \sum_{j=1}^{k} \bar{Y}_j(k) = n^{-1} \sum_{i=1}^{n} Y_i$$

$$(\bar{X} - \mu) = k^{-1} \sum_{j=1}^{k} (\bar{X}_j(k) - \mu) = n^{-1} \sum_{i=1}^{n} (X_i - \mu)$$

$$\hat{\beta}(k, q) = \hat{\Sigma}_x(k, q)^{-1} \hat{\sigma}_{yx}(k, q). \tag{2}$$

The quantities on the right-hand side of (2) are the sample versions of $\Sigma_x(k, q) = \mathrm{Cov}[\bar{X}_j(k)]$ and $\sigma_{yx}(k, q) = \mathrm{Cov}[\bar{Y}_j(k), \bar{X}_j(k)]$. The estimator $\hat{\beta}(k, q)$ is equal to the estimator of the slope coefficient of a least-squares regression of $\bar{Y}_j(k)$ on $\bar{X}_j(k) - \mu$. Confidence intervals for θ are given in [4].

Lavenberg and Welch [6] considered the case when $k = n$ ($b = 1$, meaning no batching), and the Z_i are i.i.d. $q+1$-variate normal vectors. They showed that $\mathrm{Var}[\hat{\theta}(n, q)] = (1 - R_{yx}^2)(\sigma_y^2/n)(n-2)/(n-q-2)$. This compares to $\mathrm{Var}[\bar{Y}] = \sigma_y^2/n$, showing that the control variate estimator has smaller variance than the sample mean if $R_{yx}^2 > q/(n-2)$. Schmeiser [3] considered the case when $q = 0$ (no control variates) and there exists a number of batches $2 \leqslant k^* \leqslant n$ such that for $k \leqslant k^*$ the dependency and nonnormality of the k batch means $\bar{Y}_j(k), j = 1, 2, \ldots, k$, is negligible. He showed that there is little additional benefit in terms of point and interval estimator performance from $k > 30$ batches, provided $k^* > 30$.

Nelson [4] examined the joint effect on variance reduction and confidence interval performance of simultaneously applying batching and control variates, so that the results of Lavenberg and Welch and Schmeiser are special cases. He found that as the number of control variates increases from $q = 1$ to 5, $30 \leqslant k \leqslant 60$ batches assure good point and interval estimator performance, provided $k^* > 60$. Also, as the number of batches approaches 60, the penalty, relative to the sample mean, of using even an ineffective control variate is slight, while the improvement from using an effective control variate is large. This is an important result for the design of BMCV, since BMCV automatically selects control variates using statistical procedures rather from *a priori* knowledge about which potential control variates are strongly correlated with the output variable Y_i. If Z_i, $i = 1, 2, \ldots, n$ cannot be partitioned into at least 30 acceptable batches, then Nelson recommends increasing n.

We have been implicitly assuming that the simulation output process can be represented by Z_i, $i = 1, 2, \ldots, n$, as defined above. However, a simulation output process may have a continuous-time index. In that case batching by time, rather than by count, is necessary to obtain an output process of the form considered here. For example, if we have a continuous-time process $Z(t), 0 \leqslant t \leqslant \tau$, then

$$\bar{Z}_j(k) = b^{-1} \int_{(j-1)b}^{jb} Z(t)\mathrm{d}t \tag{3}$$

where $b = \tau/k$ and τ is fixed, rather than n. Since both discrete and continuous-time output processes are possible in general simulation experiments, procedure BMCV batches all output variables by time. While this makes the number of outputs per batch a random variable for discrete-time outputs, the expected number of outputs per batch is the same for all batches and the expected values of the batch means are identical provided no batch is empty.

BMCV METHODOLOGY

There were several reasons for choosing to combine the batch means method of output analysis and the control variates VRT for automated variance reduction in infinite-horizon simulations. We begin this section with an overview of these reasons, and then discuss in the subsections the specific methods developed to implement the approach. It is important to notice that there are not obvious solutions to many of these implementation issues; thus, we carefully justify the approach we chose.

The batch means method is designed for a single realization, so it has the advantage of having to eliminate initial-transient effects only once. Also, the batch means method has been successfully automated; see for example Fishman [7], Law and Carson [8], Mechanic and McKay [9], and Schriber and Andrews [10]. When applied effectively the batch means method yields (almost) i.i.d. observations, which is a requirement for many VRTs including control variates.

The most difficult problem with using the batch means method is determining the batch size that yields nearly i.i.d. batch means. The additional complication of multivariate batching arises when batching to facilitate control variates. The solution of these two problems is discussed in a subsection below.

One key advantage of the control variates VRT is that it does not alter the simulation output process. Some VRTs, such as antithetic variates and stratified sampling (see for instance [11]), do alter the output of the simulation experiment to achieve a variance reduction. This may facilitate the estimation of some parameters of interest at the expense of making it more difficult to estimate others. The control variates VRT can be applied automatically to multiple parameters without conflict, since it only changes the estimator.

The particular type of control variates BMCV employs are called internal or concomitant control variates. Internal control variates are input random variables. Inputs are random variables with experimenter specified probability distributions that describe the randomness in a stochastic model; e.g. the interarrival times and service times in a queueing simulation. Thus, internal control variates are available in any computer simulation of a stochastic system. BMCV further restricts attention to mutually independent sequences that are each composed of i.i.d. input random variables for reasons discussed later.

Unfortunately, it is sometimes difficult to predict which input random variables, if any, will be strongly correlated with the output random variables of interest. Thus, control variate selection is an important problem for an automated procedure. The selection procedure BMCV uses is discussed in a subsection below.

Batching

The control variates VRT requires equal numbers of batches for the output variable of interest Y and the potential control variates X. To ensure the same number of batches, BMCV batches by time rather than count. Thus, the batches represent intervals of time over which variables are observed. Batching for all variables takes place during the execution of the simulation run as opposed to storing all of the simulation inputs and outputs and batching after termination of the run. This approach keeps storage requirements to a minimum, which is important since we may want to consider a large number of parameters of interest and potential control variates.

Let $k_{\max}$ be the maximum number of batches permitted (the default value is $k_{\max} = 60$) and t_0 the simulation clock time when data collection begins; t_0 may not equal 0 if an initial-transient period has been deleted. The algorithm below details how multivariate batching is accomplished for discrete-time index variables. The algorithm shows how batching is done for a single variable, generically denoted W, but the same procedure is applied independently to each output variable and control variate.

Algorithm: Batching for Discrete-time Variables

1. Begin with a batch size of $b \leftarrow 1$ time unit.
2. Take note of the time, denoted t, when an observation W_i is generated. Determine which batch sum, denoted p, the observation belongs to as follows: $p \leftarrow [(t - t_0)/b] + 1$, where [ ] is the largest integer function.

3. If $p > k_{\max}$ then $b \leftarrow 2b$ and $k \leftarrow k/2$, where k is the current number of batches. Recompute the batch sums and number of observations per batch corresponding to the new batch size. Recompute p. If $p > k_{\max}$, then repeat this step; otherwise continue.
4. Add the observation to the pth batch sum, and add 1 to the number of observations in the pth batch.

An important property of this batching procedure is that it is applicable even if the length of the realization is not specified by total simulation time, but rather by a count or some other condition leading to a random stopping time. If a fixed batch size b were specified when the stopping time is random, then it would be possible to end up with only a small number of nonempty batches. Using the default value of $k_{\max} = 60$, BMCV keeps the number of batches in the range of 30–60 as recommended in [4].

Another property of the procedure is that it matches batches of Y and X over the same time interval in order to achieve a strong correlation between the output of interest and the control variates. If each variable was batched separately by count then, for example, the ith batch means $\bar{Y}_i(k)$ and $\bar{X}_i(k)$ might not be strongly correlated if they were formed from batches of observations in different time intervals.

The batching procedure for continuous-time index variables is similar to the above procedure except that the observed value of the variable multiplied by the length of time it stayed at that value is added to the appropriate batch sum each time the value of the variable changes. As shown in (3), batching by time is the natural approach for continuous-time variables.

Whenever point and interval estimators are required, usually after the termination of the realization, BMCV performs the "clean-up" procedure below to ensure a common batch size and number of batches for all output Y and control variates X. The batch sizes at any particular time could be different; this is because the batch sizes of all variables are updated individually so that some variables might have had their batch size recently doubled, while that event might not yet have occurred for other variables. Also, step 3 below ensures that we do not use the last batch if it is empty for any variable.

Algorithm: Common Batch Size

1. Let b_0 be the maximum of the current batch sizes for all the outputs and the control variates.
2. For all variables with batch size $b < b_0$, combine the batches until all variables have batch size b_0. (Comment: The batching procedure ensures that b_0 is always a multiple of any other current batch size b.)
3. Compare the new number of nonempty batches for all variables and let k_0 be the minimum. This is the common number of batches.

The batching and common batch size procedure yield a common batch size b_0 and corresponding number of batches k_0. However, neither independence nor multivariate normality of $\bar{Z}_j(k_0), j = 1, 2, \ldots, k_0$ is assured. BMCV determines a number of batches $k \leqslant k_0$ that appears to give i.i.d. multivariate normal batch mean vectors as follows: A test of independence is applied sequentially to a smaller and smaller number of batches (larger and larger batch size) until the test is passed. For the number of batches that pass the test of independence, a test of multivariate normality is applied to a smaller and smaller number of batches (larger and larger batch size) until the test is passed. The number of batches k that results from this procedure is used to determine point and interval estimates.

Rather than performing a test of multivariate independence on the batch vectors $\{\bar{Z}_j(k_0)\}$, BMCV performs a test of univariate independence on $\{\bar{Y}_j(k_0)\}$. Since BMCV uses mutually independent sequences of i.i.d. input random variables as control variates, the batch means of the control variates should be nearly independent (the only dependence results from the dependence between the random number of observations in successive batches for discrete-time index variables). While it is not necessarily the case that, say, $\bar{Y}_j(k)$ and $\bar{X}_{j-1}(k)$ are independent, it is assumed that the stronger dependence is within the output sequence $\{\bar{Y}_j(k)\}$. Thus, determining a batch size that approximates independence for this sequence will also approximate independence for the $\{\bar{Z}_j(k)\}$ sequence. This approach permits the use of the more powerful univariate test of independence in Fishman [7], which is based on the lag-1 autocorrelation and is applicable for $k \geqslant 8$ batches.

The test of multivariate normality is performed on the (largest) number of batches $k' \leqslant k_0$ that passes the test of independence. The test implemented in BMCV, which is an extension of the univariate Shapiro–Wilk test due to Malkovich and Afifi [12], assumes independent vectors $\bar{Z}_j(k')$, $j = 1, 2, \ldots, k'$, which is why the test of independence is performed first. Unfortunately, at the time the experiments reported below were performed, the applicability of the test was limited by the availability of critical values. Critical values were only available for $k' = 10$, 25 or 50 batches and $q = 1$, 2 or 4 control variates at type I error levels of 0.05 and 0.10. Because of this limitation, BMCV permits the normality test to be disabled. In the experiments performed, employing or not employing a normality test did not seem to have much impact on the performance of the BMCV, meaning that independence of the batch means seems to be the more critical issue. A program to determine critical values for any number of batches or control variates is now available and has been used to complete the test in BMCV [13].

It is important to recall that passing tests of independence and normality does not guarantee i.i.d. multivariate normal batch vectors. In addition, the overall significance level when sequentially applying these tests to the same data is unknown. This emphasizes the importance of the results in [4], because they show that little is gained in point and interval estimator performance from achieving $k > 60$ batches for a fixed total sample size n provided 5 or fewer control variates are used. Since independence and normality are more likely with fewer, larger batches, keeping the number of batches between 30 and 60 makes the errors inherent in such tests less critical.

Selecting control variates

BMCV uses stepwise regression (see for instance Neter and Wasserman [14]) to select a subset of control variates from a set of potential control variates specified by the experimenter. The stepwise method fits a sequence of regression models by adding or deleting a control variate at each step based on a partial F-value for each control variate. Two significance levels, α_e and α_s, determine the difficulty for a control variate to enter and stay in the regression, respectively. The choice of α_e is quite important. As shown in the Background section, there is a loss in potential variance reduction associated with larger values of q because of the loss ratio $(k-q)/(k-q-2)$. This suggests using a small significance level α_e to make entry difficult. However, entry into the regression must not be made too difficult because R^2_{yx} increases as controls are added, leading to greater variance reduction. We set $\alpha_e = 0.10$, which seemed to perform well in experiments. While selection mistakes are possible, Nelson [4] showed that even selection of an ineffective control variate will not seriously degrade estimator performance provided $k > 30$.

PROCEDURE BMCV

In this section we briefly outline how procedure BMCV works; complete details are given in Añonuevo [15]. BMCV interfaces with SIMSCRIPT II.5 (C.A.C.I. Inc.), which is a general purpose programming language containing features that support discrete-event and process-interaction simulation models. See, for instance, Russell [16]. Procedure BMCV is written in SIMSCRIPT II.5, and calls routines from the IMSL Library (IMSL, Inc.). It is not the purpose of this paper to present a software product, but the authors will provide a listing of the source code and a user's guide on request.

The following declarations are required by BMCV: the output variable(s) of interest (Y_i) and the potential control variates ($X_{1i}, X_{2i}, \ldots, X_{qi}$), the expected values of the potential control variates ($\mu_1, \mu_2, \ldots, \mu_q$), and the maximum number of batches ($k_{\max}$).

A statement of the BMCV algorithm follows. The algorithm is presented as if θ is a scalar, but multiple parameters of interest are permitted. BMCV contains a routine for clearing statistics after an initial-transient period which is not shown below.

Algorithm: Procedure BMCV

1. Declarations: Y_i; $X_{1i}, \ldots, X_{qi}$; $\mu_1, \ldots, \mu_q$; $k_{\max}$ (default 60). Begin with a batch size of $b \leftarrow 1$ time unit.
2. Collect and batch $\bar{Y}_j(k), \bar{X}_{1j}(k), \ldots, \bar{X}_{qj}(k)$ by time as discussed in the Batching section above.
3. If the BMCV output routine is called, then perform the common batch size procedure to ensure that Y and X have common batch size, b_0, and corresponding number of batches, k_0.

4. Perform the test of independence on $\bar{Y}_j(k_0)$, $j = 1, 2, \ldots, k_0$. If the test fails, then reduce the number of batches, k', as follows: $k' \leftarrow n/mb_0$, $m = 2, 3, \ldots$, where $m-1$ is the number of times the test is performed. Recompute the batch means and repeat the test. Report if $10 \leqslant k' \leqslant 30$ but continue. If $k' < 10$ then there are insufficient batches, so print $\bar{Y}$; 90 and 95% confidence intervals for θ; estimated Var$[\bar{Y}]$; and stop.
5. Perform the test of multivariate normality on $\bar{Y}_j(k')$, $\bar{X}_{1j}(k')$, $\ldots$, $\bar{X}_{qj}(k')$, $j = 1, 2, \ldots, k'$, where k' is initially the number of batches that passed the test of independence. Use the same procedure as step 4 if the test fails. The resulting number of batches is k.
6. Perform stepwise regression of $\bar{Y}_j(k)$ on $\bar{X}_{1j}(k) - \mu_1, \ldots, \bar{X}_{qj}(k) - \mu_q$ to select $q' \leqslant 5$ control variates from the q potential control variates.
7. Compute and report the results: point estimates $\hat{\theta}(k, q')$, $\bar{Y}$; 90 and 95% confidence intervals for θ; estimated Var$[\hat{\theta}(k, q')]$, Var$[\bar{Y}]$; and estimated percentage variance reduction.

If the test of independence or normality fails BMCV does not halve the batch size as is done in some batching algorithms, since the goal is to accept the largest number of batches less than $k_{\max}$ possible. Also, if the number of acceptable batches is less than 10, the current version of BMCV stops after printing point and interval estimates without control variates and reporting that the independence or normality test failed. No further reduction in the number of batches is made because the independence test is only considered valid for $k \geqslant 8$. Control variates are not employed because of the requirement that $R^2_{yx} > q/(k-2)$ to achieve a variance reduction, which is difficult to meet if k is too small. Another possibility (which was not implemented) would be to restart the simulation and increase the total sample size.

EXPERIMENTAL EVALUATION

In this section we report the results of experiments applying BMCV to simulations of two systems: the M/M/1 queue, and the closed machine-repair system that Wilson and Pritsker [17, 18] used to test their standardized control variates. These experiments were chosen for three reasons: first, the parameters of interest for both systems can be determined analytically, so the performance of the confidence interval procedure can be evaluated. Second, the M/M/1 queue is a standard example used to test many proposed output analysis procedures, so it provides a point of common comparison for future research. Finally, Wilson and Pritsker's experimental design used multiple independent realizations, deleting an initial-transient period on each one, which is an alternative to the single realization approach of BMCV.

In all the experiments the performance of the control variate estimator is relative to the sample mean based on the same number of batches, which we call the crude estimator. The performance measures are: (1) the estimated percentage variance reduction, (2) the estimated percentage reduction in the half width of the 90 and 95% confidence intervals, and (3) the estimated actual probabilities of coverage of the nominal 90 and 95% confidence intervals.

Results for the M/M/1 queue

The M/M/1 queue is a single-channel, single-server queueing system. Arrivals to the system are represented by a stationary Poisson process with rate λ_1 customers per unit time. Service is rendered to one customer at a time on a first-come–first-served basis. Service times are i.i.d. random variables with an exponential distribution having mean $1/\lambda_2$. The parameters of interest are the long-run expected queue length and customer delay in queue.

The basic experiment was to simulate the M/M/1 queue for 8400 time units, deleting outputs from the first 2000 time units. The entire experiment was replicated 50 times to estimate the average performance of the estimators and the actual probability the confidence intervals cover the parameters of interest. The potential control variates were the customer interarrival times, X_{1i}, and customer service times, X_{2i}. The index i represents the ith customer, and for fixed j, $\{X_{ji}, i = 1, 2, \ldots\}$ is a sequence of i.i.d. random variables. The expected values of these random variables are $1/\lambda_1$ and $1/\lambda_2$, respectively. The value of λ_2 was fixed at 1, and experiments were performed with $\lambda_1 = 0.2$, 0.5, and 0.9 to achieve different traffic intensities. The traffic intensity is a measure of the customer load on the queue.

The experiment results are contained in Table 1. It should be noted that in these runs the test

Table 1. Performance of BMCV estimators for M/M/1 queue

	% variance reduction	% half width reduction		Coverage probability	
		90% c.i.	95% c.i.	90% c.i.	95% c.i.
Traffic intensity 0.2					
queue length	25.62	13.65	13.62	0.88	0.98
waiting time	23.50	12.39	12.37	0.90	0.94
Traffic intensity 0.5					
queue length	41.70	23.86	23.84	0.92	0.94
waiting time	34.78	19.35	19.33	0.86	0.92
Traffic intensity 0.9					
queue length	24.02	13.51	13.36	0.68	0.78
waiting time	21.37	11.93	11.80	0.62	0.66

of multivariate normality was not applied because critical values were not available for the number of batches that passed the test of independence. The tables show variance reductions of 20 to 40%, and reductions in confidence interval half widths of 10 to 20%. The estimated confidence interval coverages were close to the nominal coverage except for the high traffic intensity case. The M/M/1 queue with high traffic intensity is a notoriously difficult case for output analysis methods using a single realization. The control variate confidence intervals and the crude confidence intervals (not reported in Table 1) performed almost identically in terms of coverage for all traffic intensities.

Results for the closed machine-repair system

The machine-repair system operates as follows: there are initially 5 machines in operation and 2 idle spares. The time to failure for an operating machine is exponentially distributed with mean $\mu_1 = 10.0$ time units. When a machine fails it needs a major overhaul with probability 0.25, in which case it waits in a FCFS queue for a single repairman. The time required to do a major overhaul is exponentially distributed with mean $\mu_2 = 1.5$ time units. Those failed machines not requiring a major overhaul receive minor repair on a FCFS basis from a different repairman whose repair time is exponentially distributed with mean $\mu_3 = 1.0$ time unit. Finally, all repaired machines are inspected by a single inspector. Those machines that pass inspection (probability 0.9) return to the queue of spares if 5 machines are currently operating, or directly into service if less than 5 are operating. Machines that fail inspection are returned to the minor repair facility. The time required for inspection is exponentially distributed with mean $\mu_4 = 0.5$ time units. We are interested in estimating the long-run average number of operating machines, the average utilization of the repairmen and the inspector, the expected waiting time for repair and inspection, the expected queue length at each station, the expected time a machine spends in the pool of spares, the expected number of idle spares, and the expected response time for a failed unit to again become operational. The actual values of these parameters are given in Wilson and Pritsker [18].

The basic experiment was to simulate the machine-repair system for 7400 time units, deleting outputs from the first 1000 time units. This leaves 6400 time units of usable output; 6400 is a convenient total time for BMCV because it gives exactly 50 batches of 128 time units. Wilson and Pritsker generated 30 independent realizations of 250 time units each, deleting outputs from the first 50 time units of each realization. Thus, they expended a total of 7500 time units, of which 6000 were usable. This illustrates the advantage of a single realization method: out of a smaller total budget (7400 vs 7500 time units) BMCV was able to allocate more effort to removing the initial transient period (1000 vs 50 time units per realization) while still obtaining more usable output (6400 vs 6000 time units). Had we allocated only 50 time units to the initial transient period then the increase in usable output would have been even greater.

As in [8], we replicated the entire experiment 50 times to estimate the average performance of the estimators and the actual probability the confidence interval covers the parameter of interest. The potential control variates were the time to machine failure, X_{1i}, the time to do a major overhaul, X_{2i}, the time to do a minor repair, X_{3i}, and the time to inspect a machine, X_{4i}. The index i represents the ith realization of each random variable, and for fixed j, $\{X_{ji}, i = 1, 2, \ldots\}$ is a sequence of i.i.d. random variables. The expected values of these random variables were given above. Note that, unlike Wilson and Pritsker who used all four control variates on all 50 replications, the particular

Table 2. Performance of BMCV estimators for closed machine-repair system variables

	% variance reduction	% half width reduction		Coverage probability	
		90% c.i.	95% c.i.	90% c.i.	95% c.i.
Number of operating machines	43.03	24.62	24.59	0.90	0.98
Number of idle spares	73.70	48.80	48.78	0.94	1.00
Time in pool of spares	75.59	50.69	50.67	0.98	0.98
Response time	53.83	32.12	32.08	0.90	0.94

Table 3. Performance of BMCV estimators for closed machine-repair station variables

	% variance reduction	% half width reduction		Coverage probability	
		90% c.i.	95% c.i.	90% c.i.	95% c.i.
Utilization at					
major repair	48.46	28.05	27.98	0.88	0.94
minor repair	73.17	48.20	48.18	0.86	0.94
inspection	83.84	59.87	59.85	0.74	0.78
Queue length at					
major repair	13.40	6.65	6.63	0.90	0.92
minor repair	42.61	24.23	24.20	0.88	0.94
inspection	39.94	22.58	22.54	0.88	0.92
Waiting time at					
major repair	16.90	8.26	8.17	0.82	0.86
minor repair	40.53	22.82	22.79	0.84	0.96
inspection	35.21	19.48	19.45	0.76	0.90

control variates selected by the stepwise procedure in BMCV can and did differ from replication to replication.

The experiment results are contained in Tables 2 and 3. The variance reductions range from about 45 to 85%, and the percentage reductions in confidence interval half widths from 25 to 60%. These reductions compare favorably to Wilson and Pritsker [18], obtaining almost identical variance reductions, confidence interval half width reductions, and confidence interval coverage probabilities.

Since Wilson and Pritsker used standardized control variates rather than the sample mean control variates employed by BMCV, we performed additional experiments using standardized batch mean control variates to complete the comparison. The results were essentially unchanged. Standardized control variates make use of the known variance of the control variate and are asymptotically stable.

Computational burden of BMCV

To estimate the computational burden added by BMCV, two simulation experiments were conducted: the first employed only batching together with the test of independence, while the second used procedure BMCV. Both runs estimated the average response time for the closed machine-repair system. Fifty replications of 7400 time units were performed as outlined above. Procedure BMCV added an average of 1.65 cpu seconds per replication on an IBM 3081-D computer. This is a significant part of the average of 4.24 cpu second total time per replication. Since BMCV was developed for research purposes and is not a production code, improvements in efficiency are certainly possible. More importantly, however, the primary factors that affect the computational burden added by BMCV are the number of parameters of interest and potential control variates, so the size of the simulation experiment in terms of number of lines of code or speed of execution does not affect BMCV. Thus, a simulation with much longer total execution time but the same number of parameters of interest and control variates should experience about the same *absolute* increase in cpu time. An increase of 1.65 cpu seconds in an experiment requiring several hundred cpu seconds will be worth the effort even if the variance reductions are modest. This is important because it is in large, expensive simulation experiments that variance reduction is most important. Since BMCV is largely automated, the additional burden on the experimenter is negligible.

DISCUSSION

The experiment results in the previous section show that BMCV implements a competitive method for estimating infinite-horizon parameters. However, to be more general BMCV could

include several other features, for example, a run-length-control procedure to determine the initial-deletion amount and the total run length based on the available budget. BMCV only contains a routine that can be called at any time to clear statistical variables. Since it is possible that no batch size passes the test of multivariate normality, a procedure to form Jackknife point and interval estimators (Bratley *et al.* [19]) could also be included. Nonnormality causes $\hat{\theta}(k, q)$ to be a biased estimator, and Jackknifing is a procedure to reduce this bias. Currently, if θ is multivariate, BMCV constructs a separate confidence interval for each scalar component. Thus, the overall confidence level is not the 90 or 95% reported for the individual intervals. Some recent results for control variate estimation of multivariate parameters could be used to construct a joint confidence region; see for instance Venkatraman and Wilson [20]. Finally, BMCV is designed only for single realization experiments, but could be modified to use the method of independent replications.

Acknowledgements—This research was partially supported by a Seed Grant from the Office of Research and Graduate Studies, The Ohio State University. The authors benefitted from discussions with Gordon Clark, Jane Fraser, Marc Posner and Charles Reilly of The Ohio State University.

REFERENCES

1. B. L. Nelson, A decomposition approach to variance reduction. *Proc. 1985 Winter Simulation Conf.*, pp. 23–32 (1985).
2. B. L. Nelson, On control variate estimators. *Comput. Opns Res.* **14**, 219–225 (1987).
3. B. Schmeiser, Batch size effects in the analysis of simulation output. *Opns Res.* **30**, 556–568 (1982).
4. B. L. Nelson, Batch size effects on the efficiency of control variates in simulation. Working Paper Series No. 1986-001, The Ohio State University (1986).
5. R. Añonuevo and B. L. Nelson, Automated estimation and variance reduction for steady-state simulations. *Proc. 1986 Winter Simulation Conf.*, pp. 871–875 (1986).
6. S. S. Lavenberg and P. D. Welch, A perspective on the use of control variables to increase the efficiency of Monte Carlo simulations. *Mgmt Sci.* **27**, 322–335 (1981).
7. G. S. Fishman, Grouping observations in digital simulation. *Mgmt Sci.* **24**, 510–521 (1978).
8. A. M. Law and J. S. Carson, A sequential procedure for determining the length of a steady-state simulation. *Opns Res.* **27**, 1011–1025 (1979).
9. H. Mechanic and W. McKay, Confidence intervals for averages of dependent data in simulations II, Tech. Rep. ASDD 17-202. IBM, Yorktown Heights, New York (1966).
10. T. J. Schriber and R. W. Andrews, Interactive analysis of simulation output by the method of batch means. *Proc. 1979 Winter Simulation Conf.*, pp. 513–525 (1979).
11. B. L. Nelson and B. W. Schmeiser, Decomposition of some well-known variance reduction techniques. *J. Stat. Comput. Simul.* **23**, 183–209 (1986).
12. J. F. Malkovich and A. A. Afifi, On tests for multivariate normality. *J. Am. Stat. Assoc.* **68**, 176–179 (1973).
13. J. R. Wilson, J. D. Tew and S. Venkatraman, private communication.
14. J. Neter and W. Wasserman, *Applied Linear Statistical Models*. R. D. Irwin, Illinois (1974).
15. R. Añonuevo, Automated estimation and variance reduction via control variates for SIMSCRIPT II.5 simulations. Unpublished M.S. thesis, The Ohio State Univ. (1986).
16. E. C. Russell, *Building Simulation Models with SIMSCRIPT II.5*. C.A.C.I., Los Angeles (1983).
17. J. R. Wilson and A. A. B. Pritsker, Variance reduction in queueing simulation using generalized concomitant variables. *J. Stat. Comput. Simul.* **19**, 129–153 (1984).
18. J. R. Wilson and A. A. B. Pritsker, Experimental evaluation of variance reduction techniques for queueing simulation using generalized concomitant variables. *Mgmt Sci.* **30**, 1459–1472 (1984).
19. P. Bratley, B. L. Fox and L. E. Schrage, *A Guide to Simulation*. Springer, Berlin (1983).
20. S. Venkatraman and J. R. Wilson, The efficiency of control variates in multiresponse simulation. *Opns Res. Lett.* **5**, 37–42 (1986).