Proceedings of the 2000 Winter Simulation Conference
J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, eds.

A STOPPING PROCEDURE BASED ON PHI-MIXING CONDITIONS

E. Jack Chen
W. David Kelton

Department of Quantitative Analysis and Operations Management
University of Cincinnati
Cincinnati, OH 45221, U.S.A.

ABSTRACT

The use of batch means is a well-known technique for estimating the variance of mean point estimators computed from a simulation experiment. This paper discusses implementation of a sequential procedure to determine the batch size for constructing confidence intervals for a simulation estimator of the steady-state mean of a stochastic process. Our *quasi-independent* (QI) procedure increases the batch size progressively until a certain number of essentially i.i.d. samples are obtained. We show that our sequential procedure gives valid confidence intervals. The only (mild) assumption is that the correlations of the stochastic process output sequence die off. An experimental performance evaluation demonstrates the validity of the QI procedure.

1 INTRODUCTION

When estimating the steady-state mean μ of some discrete-time stochastic output process $\{X_i : i \geq 1\}$ with simulation, we would like an algorithm to determine the simulation run length N so that the mean estimator (sample mean $\bar{X}(N) = \frac{1}{N}\sum_{i=1}^{N} X_i$) is asymptotically unbiased (i.e. the asymptotic approximation is valid), the confidence interval (CI) is of a pre-specified width, and the actual coverage probability of the CI is close to the nominal coverage probability $1-\alpha$. Because we assume the underlying distribution is stationary, i.e., the joint distribution of the X_i's is insensitive to time shifts, the mean estimator will be unbiased. But a workable sample size must be determined dynamically to attain the precision required of a point estimator from a simulation.

The usual method of CI construction from classical statistics, which assumes independent and identically distributed (i.i.d.) observations, is not directly applicable since simulation output data are generally correlated. Several methods of determining the simulation run length and estimating the variance of the sample mean have been proposed. In particular, Crane and Iglehart (1975) use regenerative processes; Fishman (1971) and Schriber and Andrews (1984) use time series; Priestley (1981) uses classical spectral analysis; Heidelberger and Welch (1981) use a spectral-based regression; Schmeiser (1982), Meketon and Schmeiser (1984) and Steiger and Wilson (1999) use batch means; Schruben (1983) and Nakayama (1994) use standardized time series. For an overview of existing methods of determining the simulation run length and estimating the variance of the sample mean, see Law and Kelton (2000 pp. 527-537), or Sargent, Kang and Goldsman (1992). However, many existing techniques are either too restrictive to be applied to general cases or too complicated to be implemented for practical use. Therefore, there is a need for robust procedures to determine appropriate simulation run lengths and to estimate the variance of the sample mean that can be applied to generic processes, and is easy to implement.

In the non overlapping batch-means method, the simulation output sequence $\{X_i : i = 1, 2, \ldots, N\}$ is divided into R adjacent non overlapping batches, each of size m. For simplicity, we assume that N is a multiple of m so that $N = Rm$. The sample mean, $\hat{\mu}_j$, for the j^{th} batch is calculated by

$$\hat{\mu}_j = \frac{1}{m}\sum_{i=m(j-1)+1}^{mj} X_i \quad \text{for } j = 1, 2, \ldots, R. \qquad (1)$$

Then the grand mean $\bar{\hat{\mu}}$ of the individual batch means, given by

$$\bar{\hat{\mu}} = \frac{1}{R}\sum_{j=1}^{R} \hat{\mu}_j, \qquad (2)$$

is used as a point estimator for μ. Here $\bar{\hat{\mu}} = \bar{X}(N)$, the sample mean of all N individual X_i's, and we seek to construct a CI based on the point estimator obtained by equation (2).

The use of batch means is a well-known technique for estimating the variance of point estimators computed from simulation experiments. The batch-means variance estimator is simply the sample variance of the estimator

computed on means of subsets of consecutive subsamples. The asymptotic validity of batch means depends on both the assumption of approximate independence of the batch means and the assumption of the batch means being approximately normally distributed. For the method to be practical, a good choice of batch size is necessary. We propose a method to estimate the batch size using only observed data. In contrast to results that model the unknown underlying dependence structure in terms of a few unknown parameters, the method is completely nonparametric. We do not consider the effect of any initial-transient period and require that the observations obtained are already in steady-state.

We propose a *quasi-independent* (QI) procedure (see Section 2) for constructing a CI for the mean μ of a stationary process, $X_1, X_2, \ldots$. The proposed procedure will sequentially determine the length of a simulation run so that the mean estimate satisfies a pre-specified precision requirement, and produces asymptotically valid confidence intervals. The asymptotic validity of our quasi-independent procedures occurs as the subsequence appears to be independent.

The only (mild) assumption is that the stochastic process output sequence satisfies the ϕ-mixing conditions. This assumption is satisfied in virtually all practical settings. Roughly speaking, a stochastic process is ϕ-mixing if its distant future behavior is essentially independent of its present and past behavior (Billingsley 1999). A broad class of dependent stochastic processes possess this ϕ-mixing property. Many simulation output-analysis methods have used this property to establish their validity, for example Schruben (1983), Heidelberger and Lewis (1984), and Chen and Kelton (1999).

The main advantage of our approach is that by using a subsequence of quasi-independent samples to determine the batch size, we do not require complicated computation or advanced theory. Moreover, the batch size is determined automatically without any user intervention. Furthermore, the subsequence of quasi-independent samples can be used to represent the underlying distribution, and we can apply classical statistical techniques to those samples directly; for details see Chen and Kelton (2000a).

Although our quasi-independent procedures have many desirable properties, there are also some drawbacks. First, our methods suffer from one of the problems that afflicts any sequential stopping rule: the run length of the simulation may be inappropriately short or long. The simulation may terminate inappropriately early due to statistical variability, which may cause difficulties such as the asymptotic approximation not yet being valid. However, by specifying an appropriate stopping rule, we can reduce the chance that the simulation terminates early. To avoid the simulation's running longer than necessary, though, is somewhat difficult, which arises from the fact that we double the simulation run length every two iterations.

In Section 2 we present our methodologies and proposed procedure for mean estimation. In Section 3, we show our empirical-experiment results. In Section 4, we give concluding remarks.

2 METHODOLOGIES

In this section we introduce the methodologies we used in our procedures: ϕ-mixing, systematic sampling, and the *runs-up* test, and present our quasi-independent procedure.

2.1 ϕ-Mixing

To define ϕ-mixing, let $\{X_i; -\infty < i < \infty\}$ be a stationary sequence of random variables defined on a probability space $(\Omega, \mathcal{A}, \mathrm{P})$. Thus, if $\mathcal{M}_{-\infty}^{k}$ and $\mathcal{M}_{k+j}^{\infty}$ are respectively the sequences generated by $\{X_i; i \le k\}$ and $\{X_i; i \ge k+j\}$, and if $E_1 \in \mathcal{M}_{-\infty}^{k}$ and $E_2 \in \mathcal{M}_{k+j}^{\infty}$, then for all k $(-\infty < k < \infty)$ and j $(j \ge 1)$, if there exists a sequence $\phi(1), \phi(2), \ldots$ such that

$$|P(E_2|E_1) - P(E_2)| \le \phi(j), \quad \phi(j) \ge 0,$$

where $1 \ge \phi(1) \ge \phi(2) \ge \cdots$, and $\lim_{j\to\infty} \phi(j) = 0$, then $\{X_i; -\infty < i < \infty\}$ is called *ϕ-mixing*. That is, in ϕ-mixing sequences, the lag-i covariance $\gamma_i = \mathrm{Cov}(X_k, X_{k+i}) \to 0$ as i increases. Intuitively $X_1, X_2, \cdots, X_n$ is ϕ-mixing if X_i and X_{i+j} become essentially independent as j becomes large. For example, the waiting-time W_i of an M/M/1 delay-in-queue is ϕ-mixing, because W_i and W_{i+j} become essentially independent as j becomes large; see Daley (1968). The ϕ-mixing conditions implies that the correlations of the stochastic process output sequence die off, which is the assumption that the QI algorithm really depends on.

These weakly dependent processes typically obey a Central Limit Theorem (CLT) for dependent process of the form

$$\frac{\sqrt{N}[\bar{X}(N) - \mu]}{\sigma} \xrightarrow{\mathcal{D}} \mathcal{N}(0,1) \quad \text{as} \quad N \to \infty,$$

where

$$\sigma^2 \equiv \lim_{N\to\infty} N\mathrm{Var}[\bar{X}(N)] = \sum_{i=-\infty}^{\infty} \gamma_i$$

is the steady-state variance constant (SSVC). If the sequence is independent, then the SSVC is equal to the process variance σ_x^2, i.e., $\mathrm{Var}(X_i)$. For a finite sample m, let

$$\sigma^2(m) = \gamma_0 + 2\sum_{i=1}^{m-1}(1 - i/m)\gamma_i.$$

It follows that $\lim_{m\to\infty}\sigma^2(m) = \sigma^2$ and $\text{Var}[\hat{\mu}_j] = \sigma^2(m)/m \approx \sigma^2/m$, provided that m is sufficiently large.

Although asymptotic results are often applicable when the amount of data is "large enough," the point at which the asymptotic results become valid generally depends on unknown factors. An important practical decision must be made regarding the sample size N required to achieve the desired precision. Therefore, both asymptotic theory and workable finite-sample approaches are needed by the practitioner. We propose our quasi-independent method; the procedure uses a subsequence of n samples (taken from the original output sequence of N observations) that appears to be independent to determine the batch size. We accomplish this by *systematic sampling*, i.e., select a number l, choose that observation and then every l^{th} observation thereafter. Here l will be chosen sufficiently large so that samples are essentially independent. This is possible because we assume the underlying output sequence satisfies the ϕ-mixing conditions.

We compute the size l for our systematic sampling based on the runs-up test (see Section (2.2)) and choose the number of required independent observations $n = 4000$, the minimum recommended sample size for the runs-up test (Knuth, 1998). The simulation run length is then $N = nl$. Let R be the number of batches (see Section (2.3) for how R is determined in our procedure); then the batch size is $m = N/R$. Here R will be chosen so that m will be an integer.

2.2 Test of Independence

Because the required sample sizes are drastically different between i.i.d. and correlated sequences, it is beneficial to check whether the input data appear to be independent. We use a *runs-up* test for this purpose, see Knuth (1998). Briefly, a run up is a monotonically increasing subsequence and we consider the length of a run up. The observation immediately following a run is discarded so that subsequent runs are independent. Therefore, a straightforward chi-square test can be used on the proportion of different run length to check whether the sequence appears to be independent. The runs-up tests looks solely for independence and has been shown to be very powerful (Knuth, 1998). If the output data sequence appear to be dependent, then a sequential procedure will be used. The runs-up test is used in our procedure to determine the lag length l so that observations that are at least l units apart will be essentially independent.

However, in some cases, even when the subsequence should have been independent, it continues to fail the runs-up test with lag up to 2^{14}. This is because some assumptions of the runs-up test are violated. For example, the distribution function of the waiting-time of an M/M/1 delay-in-queue is $F(x) \to 1 - \frac{\lambda}{\nu}e^{-(\nu-\lambda)x}$, $x \geq 0$ (where λ is the arrival rate and ν is the service rate); this distribution is discontinuous at $x = 0$ and has a jump from 0 to $1-\rho$ ($\rho = \lambda/\nu$ is the traffic intensity). If the traffic intensity $\rho = 0.5$, then $F(0) = 0.5$, i.e., about fifty percent of the waiting times will be zero. Therefore, the probability of a run of length of 1 will be unusually high. This "over mixing" will cause the subsequence to fail the runs-up test. To correct this problem, we will increase the run length with probability $1/(r+1)$ when two elements are equal, where r is the current run length. If the distribution is continuous, the probability of run length r is $r/(r+1)!$, under the null hypothesis that the sequence consists of i.i.d. random variables. Moreover, if there are $r+1$ observations, the possibility of the $(r+1)^{th}$ observation being the largest is $1/(r+1)$. For example, if $X_i = X_{i+1}$ and the run length r up to X_i is 2, we will generate a uniform (0,1) random variable u; if $u < 1/3$, we will let $X_i \leq X_{i+1}$ and increase the run length by one, otherwise we will let $X_i \geq X_{i+1}$ and start a new run-up sequence. Our preliminary experimental results show that this adjustment works well even for discrete distributions.

2.3 Quasi-Independent Procedure

One of the most important aspects of the design of experiments is the determination of the appropriate sample size of the basic experiment. Because almost all simulation output processes are autocorrelated and nonstationary, no fixed-sample-size procedure can be relied upon to produce a CI that covers μ with the desired probability level, if the fixed sample size is too small for the system being simulated. In addition to this problem of coverage, a simulator might want to determine a sample size large enough to produce a CI with a small absolute precision ϵ. It will seldom be possible to know in advance even the order of the magnitude of the sample size needed to meet these goals in a given simulation problem, so some sort of procedure to increase iteratively this sample size would be needed. Consequently, sequential procedures have been developed (Law and Kelton 2000).

We propose a simple *quasi-independent* (QI) algorithm to determine the simulation run length. The aim of the QI method is to continue the simulation run until we have obtained a pre-specified number of essentially independent random samples by skipping highly correlated observations. In this paper, we focus on computing the variance of the mean directly from several mean estimators, i.e., the $\hat{\mu}_j$'s. The runs-up test is somewhat sensitive to the sample size used. The sample size of 4000 is the most consistent one among three sample sizes 4000, 6000 and 8000 we tried. Let l be the minimum lag length so that the subsequence will appear to be independent based on the runs-up test. The algorithm will sequentially increase the simulation run length by increasing the lag length l.

The simulator will generate $n = 4000$ (the minimum recommended sample size for the runs-up test) observations initially. We allocate a buffer A with size $t = 12,000$ $(3n)$ to store our quasi-independent sequence. We then carry out a runs-up test with these 4000 samples $y_1, y_2, y_3, \ldots, y_{4000}$, as our initial (0^{th}) iteration. If the sequence appears to be independent, then the simulation run length should be large enough to obtain an unbiased variance estimator. If the sequence appears to be dependent, we generate another 4000 observations and store their values in buffer A after the ones already there. We then carry out a runs-up test with first $4,000$ odd samples in buffer A, namely samples $y_1, y_3, y_5, \ldots, y_{2i+1}, \ldots, y_{7999}$, $i = 0, 1, 2, \ldots, 3999$ as our 1_A^{th} iteration. If the subsequence in buffer A appears to be independent, the algorithm will stop. Otherwise, we generate another 4000 observations and store their values in buffer A after the ones already there. We will then carry out another runs-up test as our 1_B^{th} iteration, in the following fashion: $y_1, y_4, y_7, \ldots, y_{3i+1}, \ldots, y_{11998}$, $i = 0, 1, 2, \ldots, 3999$. If this subsequence appears to be independent, the algorithm will stop. At iterations 0, 1_A, and 1_B, the simulation run lengths are 4000, 8000 and 12000 respectively. So far, all observations are stored in buffer A, therefore, we will be able to chose any batch size R such that $N = Rm$. We chose $R = 4$ for these 3 iterations.

We alternate the iteration numbers between A and B. For example, the 1_B^{th} iteration is followed by the 2_A^{th} iteration and the 2_A^{th} iteration is followed by the 2_B^{th} iteration and so forth. The reason for this notation is because we double the batch size every two iterations. If the subsequence appears to be dependent at the k_B^{th} iteration, samples $y_2, y_4, y_6, \ldots, y_{12000}$ will be discarded. The remaining samples $y_1, y_3, y_5, \ldots, y_{11999}$ will be rearranged in the buffer as samples $y_1, y_2, y_3, \ldots, y_{6000}$. There will be 3 batches with batch size m after the k_B^{th} iteration. We will then generate another 2000 samples and store their values in buffer A, namely $y_{6001}, y_{6002}, \ldots, y_{8000}$, as our k_A^{th} $(k \geq 2)$ iteration. However, we only store 2^{k-1} lag observations (from all observations generated) in the buffer at the k^{th} iteration $(k \geq 2)$. So the buffer will store samples with each sample taken $2^{k-1} - 1$ observations apart in the k^{th} iteration. In effect, we generated $m = 2000l$ $(l = 2^{k-1})$ observations at k_A iterations. We can then compute the fourth batch mean from these m observations. We will then carry out a runs-up test with samples y_{2i+1}, $i = 0, 1, 2, \ldots, 3999$. If the subsequence appears to be independent, the algorithm stops. Otherwise, we generate another 4000 samples that are 2^{k-1} lag observations and store their values in buffer A as $y_{8001}, y_{8002}, \ldots, y_{12000}$ as our k_B^{th} iteration. Moreover, at the beginning of the k_B^{th} $(k \geq 2)$ iteration, we will reduce the number of batches from 4 to 2, by taking the average of adjacent batch means. Therefore, we increased the batch size from m to $m' = 2m$. In effect, we generated $m' = 4000l$ $(l = 2^{k-1})$ observations at k_B iterations. We compute the third batch mean from these m' observations. We will then carry out a runs-up test with samples y_{3i+1}, $i = 0, 1, 2, \ldots, 3999$. This process will be done iteratively until the subsequence appears to be independent. The sample size N will be equal to $(4000)2^k$ at the k_A^{th} $(k \geq 1)$ iterations and $(1.5)(4000)(2^k)$ at the k_B^{th} iterations.

In the simulation run length determination stage of our algorithm, the batch size will be $(4000)2^{k-2}$, $k \geq 2$ at the k_A^{th} iterations, and be $(4000)2^{k-1}$, $k \geq 2$ at the k_B^{th} iterations. The corresponding number of batches R will be 4 and 3 respectively. The following shows the incremental sample sizes and the batch sizes (for iterations after 2_A) at each iteration:

0	1_A	1_B	2_A
4000	4000	4000	$4000(2 \times 2000)$
$0 : 1_A$	$1_B : 2_A$	2_B	3_A
8000	8000	$8000(2 \times 4000)$	$8000(4 \times 2000)$
$0 : 2_A$	$2_B : 3_A$	3_B	4_A
16000	16000	$16000(4 \times 4000)$	$16000(8 \times 2000)$
...			

The equation inside the parenthesis shows how the batch size is determined. For example, the batch size is 8000 (4×2000) at the 3_A^{th} iteration. We obtain 2000 samples with each sample is the lag 4 observation, therefore, we generated 8000 observations. Note that the entire observations are used to compute the batch means. The batch size m is dynamically determined at the beginning of each iteration, so a new batch mean can be computed when additional m observations are generated. We discard samples in the QI sequence so that the size of the QI sequence will be no more than t. Samples in the QI sequence are used by the runs-up test to determine batch sizes and are not used to compute the batch means. We assume that the batch size determined by our algorithm is sufficiently large so that the batch means $\{\hat{\mu}_j : 1 \leq j \leq R\}$ are i.i.d. normal.

This method of estimating the variance of $\hat{\mu}$ is obtained directly by obtaining multiple mean estimators. Because $\hat{\mu}_j$ has a limiting normal distribution, by the CLT a CI for μ using the i.i.d. $\hat{\mu}_j$'s can be approximated using standard statistical procedures. That is, if there are R batches, the point estimate is $\bar{\hat{\mu}} = \frac{1}{R}\sum_{j=1}^{R} \hat{\mu}_j$, and the ratio

$$T = \frac{\bar{\hat{\mu}} - \mu}{S/\sqrt{R}}$$

would have an approximate t distribution with $R - 1$ d.f. (degrees of freedom), where

$$S^2 = \frac{1}{(R-1)} \sum_{j=1}^{R} (\hat{\mu}_j - \bar{\hat{\mu}})^2$$

is the usual unbiased estimator of the variance of μ. This would then lead to the $100(1-\alpha)\%$ CI for μ,

$$\bar{\hat{\mu}} \pm t_{R-1,1-\alpha/2}\frac{S}{\sqrt{R}}, \tag{3}$$

where $t_{R-1,1-\alpha/2}$ is the upper $1-\alpha/2$ quantile for the t distribution with $R-1$ d.f. ($R \geq 2$).

This CI is approximately valid when the batch size m becomes large because the batch means $\hat{\mu}_1, \hat{\mu}_2, \ldots, \hat{\mu}_R$ become almost independent (since the underlying process satisfies the ϕ-mixing conditions) and almost normally distributed (from an appropriate CLT for ϕ-mixing sequences).

The Quasi-Independent Means Algorithm:

1. Remark: $itmax$ is the maximum iteration the algorithm will try, t is the size of buffer A, which are used to store QI samples, and k is the number of iterations. Each iteration k contains two sub-iterations k_A and k_B.
2. Set the required number of independent samples $n = 4000$, buffer size $t = 12,000$ and iteration number $k = 0$. Generate $N = n$ observations.
3. Carry out a runs-up test to determine whether the sequence appears to be independent. Our runs-up test uses $n = 4000$ samples. If this the initial iteration, use lag 1 samples in the QI sequence. If this is a K_A iteration, use lag 2 samples in the QI sequence. If this is a K_B iteration, use lag 3 samples in the QI sequence.
4. If the subsequence appears to be independent, go to step 10.
5. If the current iteration is k_A iterations, start a k_B iteration. If the current iteration is the initial or k_B iterations, set $k = k+1$. If $k > itmax$, go to step 10; otherwise start a k_A iteration.
6. If this is the 1_A^{th} or 1_B^{th} iteration, generate 4000 observations, store those values in buffer A after the ones already there and go to step 3.
7. If this is a k_A iteration ($k \geq 2$), then discard every other sample in the buffer, rearrange the rest of n samples in the first half of the buffer. Generate 2000 samples that are lag 2^{k-1} observations and store them in the later portion of the buffer. The fourth batch mean is the average of $m = 2000 \times 2^{k-1}$ observations.
8. If this is a k_B iteration ($k \geq 2$), reduce the number of batches from 4 to 2 by taking average of adjacent batch means and double the batch size. Generate 4000 samples that are lag 2^{k-1} observations and store them in the later portion of the buffer. The third batch mean is the average of $m = 4000 \times 2^{k-1}$ observations.
9. Go to step 3.
10. If the simulation run length is less than or equal to $12,000$, divide the sequence into $R = 4$ batches and compute the corresponding batch size $m = N/4$.
11. Compute the point mean estimator according to equation (2).
12. Compute the confidence interval of the mean estimator according to equation (3).
13. Let ϵ be the desired absolute half-width and $r|\bar{\hat{\mu}}|$ be the desired relative half-width. If the half-width of the CI is less than ϵ or $r|\bar{\hat{\mu}}|$, terminate the algorithm.
14. Run one more batch with batch size m, set $R = R+1$. If $R = 30$, double the batch size by taking the average of adjacent batch means and reduce the batch number R to 15. Go to step 11.

Our QI procedure addresses the problem of determining the batch size m, and the number of batches R, that are required to satisfy the assumptions of independence and normality. Theoretically, if these assumptions are satisfied, then the actual coverage of the CI's should be close to the pre-specified level. The QI procedure must store the whole quasi-independent sequence, which is needed for the runs-up test.

Let the half-width

$$H = t_{R-1,1-\alpha/2}\frac{S}{\sqrt{R}}.$$

The final step in the QI procedure is to determine whether the CI meets the user's requirement for precision, a maximum absolute half-width ϵ or a maximum relative fraction r of the magnitude of the final grand mean $\bar{\hat{\mu}}$. If the relevant requirement,

$$H \leq \epsilon \tag{4}$$

or

$$H \leq r|\bar{\hat{\mu}}| \tag{5}$$

for the precision of the confidence interval is satisfied, then the QI procedure terminates: return the sample mean $\bar{\hat{\mu}}$ and the CI with half-width H. If the precision requirement (4) or (5) is not satisfied with R batches, then the QI procedure will increase the number of batches by one. This step can be repeated iteratively until the pre-specified precision is achieved.

Schmeiser (1982) studied batch-size effects in the analysis of simulation output. He recommends the number of batches, in general, should be between 20 and 30 for any fixed sample size. In order to keep the number of batches to a reasonable size, we will set the maximum number of batches to 30 in our half-width-reduction phase. At the

beginning of the half-width-reduction phase, the number of batches is 3 or 4 in our algorithm. If the number of batches has been increased to 30 and the half-width is still larger than desired, we will reduce the number of batches to 15 and double the batch size. This is accomplished by taking the average of adjacent batch means.

2.4 Properties of Estimators

Goldsman and Schmeiser (1997) list some properties that a good estimator should posses. We use these properties to assess the desirability of our algorithm. The followings describe the performance of our algorithm under each property.

- Statistical performance. The batch mean estimator is asymptotically unbiased. Based on our experiments, the asymptotic approximation is valid when the batch size is determined by our QI algorithm.
- Ease of computation. Our algorithm involves only a little more than $O(N)$ operations. The runs-up test is relatively computationally inexpensive.
- Parsimonious storage requirements. Our data storage is $12,000$ for the quasi-independent sequence and is $R = 30$ for batch means. We only need to process each observation once and do not require storing the entire N observations.
- Ease of understanding. Our algorithm requires only simple statistics: ϕ-mixing, systematic sampling, the runs-up test, and the CLT.
- Numerical stability. The limits of machine precision is the limit of our algorithm precision.
- User-specified parameters. We only require two parameters: the confidence level α and the relative precision r. We do not require the user to enter the number of batches or the batch size.
- Amenability for use in algorithms. Our algorithm can be incorporated with other procedures easily.

3 EMPIRICAL EXPERIMENTS

In this section we present some empirical results obtained from simulations using the quasi-independent procedure proposed in this paper. The purpose of the experiments was not so much to test the methods thoroughly, but rather to demonstrate the interdependence between the correlation of simulation output sequences and simulation run lengths, and the validity of our methods.

A stochastic model that has such a covariance structure and admits an exact analysis of performance criteria is the *first-order auto-regressive* (AR(1)) process, generated by the recurrence relation

$$X_i = \mu + \varphi(X_{i-1} - \mu) + \epsilon_i \text{ for } i = 1, 2, \ldots,$$

where

$$E(\epsilon_i) = 0, \quad E(\epsilon_i \epsilon_j) = \begin{cases} \sigma^2 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases},$$

$$0 < \varphi < 1,$$

and X_0 is specified to some random variate x_0 drawn from the steady-state distribution. The ϵ_i's are commonly called *error terms*.

The AR(1) process shares many characteristics observed in simulation output processes, including first- and second-order nonstationarity and autocorrelations that decline exponentially with increasing lag (so AR(1) sequences are ϕ-mixing sequences). If we make the additional assumption that the ϵ_i's are normally distributed, since we have already assumed that they are uncorrelated, they will now be independent as well, i.e., the ϵ_i's are i.i.d. $\mathcal{N}(0, 1)$. It can be shown that X has a $\mathcal{N}(0, \frac{1}{1-\varphi^2})$ distribution, and the SSVC of the AR(1) process is $1/(1-\varphi)^2$.

To get a sense of the quality of these algorithms, all experiments were executed with the initial sample size N set to $n = 4000$ and steps 13 and 14 in the algorithm were skipped, which are used to reduce the half-width of the confidence intervals. The correlation coefficient (φ) is set to 0.50, 0.75 and 0.95, respectively. Each design point is based on 100 independent replications. The results of our mean estimations of the AR(1) process are summarized in Table 1. The *cover* column lists the percentage of the CIs that cover the true mean. The μ column lists the true mean value. The *avg. s.s.* column lists the average of the sample size determined by the runs-up test. The *avg. r.p.* column lists the average of the observed relative precision of the estimator, i.e., the average of $(\bar{\hat{\mu}} - \mu)/\mu$. The *avg. hw* column lists the average of the confidence interval half-width of the mean estimators. Note that the coverages are all close to the nominal 90% confidence level. The average relative precisions are ∞ for all three AR(1) models, because the true mean $\mu = 0$.

Table 1: Coverages for CIs for the AR(1) Process with $1 - \alpha = 0.90$

φ	cover	μ	avg. s.s.	avg. r.p.	avg. hw
0.50	92%	0	12320	∞	0.049474
0.75	89%	0	29600	∞	0.056243
0.95	88%	0	171840	∞	0.127643

Another stochastic model that allows exact analysis of performance criteria is the waiting-time of the M/M/1 delay

in queue. Queuing systems are usually positively correlated and often strongly so. Furthermore, the skewness of the exponential distribution causes exponential-servers queuing models to yield output data that might be considered "worst case." Some feel that if an output-analytic method works well for an exponential-servers model, it is likely to work well in practice. We tested three M/M/1 queuing models. The service rate (ν) is set to 1.0 per period for all models. The arrival rate (λ) is set to 0.50, 0.75 and 0.95 per period, respectively.

Let $\{A_n\}$ denote the interarrival-time i.i.d. sequence and $\{S_n\}$ denote the service-time i.i.d. sequence. Then the waiting-time sequence $\{W_n\}$ is defined by

$$W_{n+1} = (W_n + S_n - A_{n+1})^+ \text{ for } n \geq 1,$$

where $w^+ = \max(w, 0)$. In order to eliminate the initial bias, w_1 is set to a random variate drawn from the steady-state distribution.

The results of our mean estimations of the waiting-time of the M/M/1 delay-in-queue are summarized in Table 2. It is deceiving that the simulation run length for $\rho = 0.50$ (of the M/M/1 delay in queue) determined by the runs-up test is larger the simulation run length for $\rho = 0.75$, because for $\rho = 0.50$ the algorithm was terminated by the exceptional rule for discrete distributions (see Section 2.2). For the M/M/1 queuing process with $\rho = 0.50$, about half of the incoming customers do not have to wait, i.e., waiting-time is 0. Therefore, there will be many customers with the same waiting times. In general, the batch size determined by the algorithm grows with the traffic intensity ρ.

Table 2: Coverages for CIs for the Waiting-time of the M/M/1 Delay in Queue with $1 - \alpha = 0.90$

ρ	cover	μ	avg. s.s.	avg. r.p.	avg. hw
0.50	93%	1.0	138240	0.012658	0.033024
0.75	86%	3.0	59520	0.032987	0.259181
0.95	90%	19.0	3097600	0.018606	1.047964

The actual coverage is close to the specified $1-\alpha$ confidence level, indicating the effectiveness of the algorithm. Moreover, if the half-width is longer than desired, we increase the number of batches R until the required precision is achieved. As the simulation run length increases, we will reduce the number of batches and increase the batch size.

Steiger and Wilson (1999) proposed the ASAP (Automated Simulation Analysis Procedure) for batch-means procedures. We take the performance measure of the ASAP from their paper and tested the QI algorithm under the same conditions for comparison. We would like to point out the these two algorithms are executed with different random numbers. The first test case they presented is a process defined by a real-valued function on a simple 2-state Discrete Time Markov Chain (DTMC) whose one-step probability transition matrix and cost vector associated with the states are respectively given by

$$\mathbf{P} = \begin{array}{c} \\ 0 \\ 1 \end{array} \begin{array}{c} \begin{array}{cc} 0 & 1 \end{array} \\ \begin{pmatrix} 0.99 & 0.01 \\ 0.01 & 0.99 \end{pmatrix} \end{array} \quad \text{and} \quad \mathbf{h} = \begin{array}{c} \begin{array}{cc} 0 & 1 \end{array} \\ \begin{pmatrix} 5 & 10 \end{pmatrix} \end{array}. \tag{6}$$

The second test case they presented is the waiting-time of the M/M/1 delay in queue with $\rho = 0.90$.

We made 100 independent simulations of the DTMC and the waiting-time of the M/M/1 delay in queue and attempted to construct nominal 90% confidence intervals for three cases:

1. no precision requirement, i.e., we terminated the procedure when a CI was constructed based on the default sample size.
2. 15% relative precision so that $r = 0.15$ in (5); and
3. 7.5% relative precision so that $r = 0.075$ in (5).

This enabled us to get a relative idea of the quality of our CIs. Tables 3 and 4 display in detail the results of our tests.

Table 3: Performance of Batch-Means Procedures for the 2-State DTMC Defined by (8) Based on 100 Independent Replications Seeking Nominal 90% Confidence Intervals of the True Mean $\mu = 7.5$

Precision Requirement	Procedure ASAP	QI
$\pm$7.5% PRECISION		
avg. sample size	22711	128000
coverage	99%	89%
avg. rel. precision	0.059	0.007370
avg. CI half-width	0.438	0.155900
var. CI half-width	0.006	0.059750

ASAP adjusts the half-width when the batch means are dependent and normally distributed, but the QI procedure attempts to obtain batch means that are independent and normally distributed. Therefore, the average sample size of the DTMC with no precision was specified by the user to be 128, 000 in the QI procedure, much larger than the 22, 711 in the ASAP with a 7.5% relative precision requirement, i.e., the achieved half-width should be smaller than 0.5775 ($7.5 \times 7.5\%$). Even though the coverages of QI CIs are less than that of the ASAP, we would like to point that the average half-width is smaller in QI CIs and the achieved relative precision is better from the QI procedure.

The average sample size of the M/M/1 queuing model when no precision was specified by the user is 636, 160 in the QI procedure, much larger than 7, 719 in the ASAP. That is, the QI procedure will obtain high precision by default.

Table 4: Performance of Batch-Means Procedures for the Waiting-Time of the M/M/1 Delay in Queue with $\rho = 0.90$ Based on 100 Independent Replications Seeking Nominal 90% Confidence Intervals of the True Mean $\mu = 9.0$

Precision Requirement	Procedure ASAP	QI
NO PRECISION		
avg. sample size	7719	636160
coverage	83%	92%
avg. rel. precision	1.088	0.022371
avg. CI half-width	11.8	0.621437
var. CI half-width	523.0	0.339490
±15% PRECISION		
avg. sample size	298950	636160
coverage	88%	87%
avg. rel. precision	0.089	0.023748
avg. CI half-width	0.783	0.567189
var. CI half-width	0.082	0.290066
±7.5% PRECISION		
avg. sample size	815755	718720
coverage	94%	93%
avg. rel. precision	0.046	0.020140
avg. CI half-width	0.413	0.464453
var. CI half-width	0.018	0.125578

So the half-width reduction phase of the QI procedure was not activated until the relative precision requirement is less than 15% (roughly), i.e., the achieved half-width should be less than 1.35 ($9.0 \times 15\%$). We do not think this is a major practical drawback for the QI procedure, because large relative half-width (larger than 15%) may provide little useful information. In the 15% and 7.5% relative precision cases, the QI procedure shows a little less coverage than the ASAP, but very close to the desired level. In the 7.5% relative precision case (i.e. the achieved half-width should be smaller than 0.675 ($9.0 \times 7.5\%$)), the sample sizes determined by the QI procedure are smaller than from ASAP. The average CI half-width of these two procedures are about the same; however, the variance of the CI half-width is larger in the QI procedure. On the other hand, the average relative precision is better in the QI procedure than in the ASAP; the average relative fraction r in the QI procedure is about 50% of that achieved in the ASAP. This indicates that the point estimator $\tilde{\hat{\mu}}$ of the QI procedure is better than that from ASAP despite the small sample sizes are used. This favorable average relative precision also indicates that those QI CIs that do not cover the true mean must miss the true mean by only a very small amount, most probably caused by the half-width being too small.

Performance data of 5.0% 2.5%, and 1.0% relative precisions for the ASAP are not available. Table 5 lists the performance of the QI procedure under these precision requirements. The average relative precision is reduced

Table 5: Performance of Batch-Means Procedures for the Waiting-Time of the M/M/1 Delay in Queue with $\rho = 0.90$ Based on 100 Independent Replications Seeking Nominal 90% Confidence Intervals of the True Mean $\mu = 9.0$

Precision Requirement	Precision 5.0%	2.5%	1.0%
avg. sample size	870400	1978880	12446720
coverage	87%	87%	87%
avg. rel. precision	0.019167	0.012639	0.004826
avg. CI half-width	0.355147	0.207948	0.086311
var. CI half-width	0.088165	0.022658	0.007058
duration hh:mm	0:49	1:54	11:36

from 0.022371 to 0.004826 when the relative precision of 1.0% is specified. Coverages from our experiments are all close to the specified 90% level; this indicates the efficacy of the QI procedure. To get an idea of the speed of the QI implementation, we also list the execution times to get these 100 independent replications. This was done on a 32-bit SUN Ultrasparc 1/140 workstation under Solaris 2.5.1. The timings (in hours and minutes) are in Table 5. One should keep in mind that these timings are strongly dependent on the machine and its state.

4 CONCLUDING REMARKS

We have presented a confidence interval for the mean μ of a stationary process. Some CIs require more observations than others before the asymptotics necessary for CIs become valid. Our proposed quasi-independent algorithm works well in determining the required simulation run length and the batch size for the asymptotic approximation to be valid. Although it is heuristic, the QI procedure has a strong theoretical basis. The results from our empirical experiments show that the QI procedure is excellent in achieving the intended coverage, not only for slightly correlated processes but also for highly correlated processes. The QI procedure does not require any extensive computation; therefore, it is able to estimate highly correlated processes with good precision in a reasonable amount of run time. For example, to obtain one CI of the waiting time of the M/M/1 delay in queue with $\rho = 0.90$ and relative precision of 5.0%, the execution takes less than 20 seconds. However, the variance of the simulation run length from our sequential procedure is large. This is not only because of randomness of the output sequence but also because we double the lag length l every two iterations. Further research is then to develop new algorithms so that the simulation run length does not need to be doubled every two iterations.

Our proposed quasi-independent algorithm requires storing a sequence of quasi-independent observations that most likely represent the underlying distribution. This sequence is used by the runs-up test to check for independence. Our procedure is completely automatic and has the desir-

able properties that it is a sequential procedure and does not require the user to have *a priori* knowledge of values that the data might assume. This allows the user to apply this method without having to make a pilot run to determine the range of values to be expected or guess and risk having to re-run the simulation, either of which represents potentially large costs because many realistic simulations are time-consuming to run. The main advantage of our approach is that by using a straightforward runs-up test to determine the simulation run length and the batch size, we do not require more advanced statistical theory, thus making it easy to understand, simple to implement, and fast to run. The simplicity of this method should make it attractive to simulation practitioners. Moreover, the QI simulation run length determination mechanism can also be used when estimating other parameters of the simulation output sequence. Chen and Kelton (2000b) use this QI algorithm to estimate quantiles.

REFERENCES

Billingsley, P. 1999. *Convergence of probability measures*. 2nd ed. New York: John Wiley & Sons, Inc.

Chen, E. J., and W. D. Kelton. 1999. Simulation-based estimation of quantiles. In *Proceedings of the 1999 Winter Simulation Conference*, ed. P.A. Farrington, H.B. Nembhard, D.T. Sturrock, and G.W. Evans, 428–434. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Chen, E. J., and W. D. Kelton. 2000a. Mean estimation based on ϕ-mixing sequences. In *Proceedings of the 33rd Annual Simulation Symposium*.

Chen, E. J., and W. D. Kelton. 2000b. Quantile estimation based on histogram. Technical report, Department of Quantitative Analysis and Operations Management, University of Cincinnati, Cincinnati, Ohio.

Crane, M. A. and D. L. Iglehart. 1975. Simulating stable stochastic systems, III: Regenerative processes and discrete-event simulations. *Operations Research* 23:33–45.

Daley, D. J. 1968. The serial correlation coefficients of waiting times in a stationary single server queue. *The Journal of the Australian Mathematical Society* 8 (4): 683–699.

Fishman, G. S. 1971. Estimating sample size in computer simulation experiments. *Management Science* 18 (1): 21–38.

Goldsman D., and B. W. Schmeiser. 1997. Computational efficiency of batching methods. In *Proceedings of the 1997 Winter Simulation Conference*, ed. S. Andradóttir, K.J. Healy, D.H. Withers, and B.L. Nelson, 202–207. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Heidelberger, P. and P. A. W. Lewis. 1984. Quantile estimation in dependent sequences. *Operations Research* 32:185–209.

Heidelberger, P. and P. D. Welch. 1981. A spectral method for confidence interval generation and run length control in simulation. *Communications of the ACM* 24:233–245.

Knuth, D. E. 1998. *The art of computer programming*. Vol. 2. 3rd ed. Reading, Mass.: Addison-Wesley.

Law, A. M., and W. D. Kelton. 2000. *Simulation modeling and analysis*. 3rd ed. New York: McGraw-Hill.

Meketon, M. S. and B. Schmeiser. 1984. Overlapping batch means: Something for nothing? In *Proceedings of the 1984 Winter Simulation Conference*, ed. S. Sheppard, U. Pooch, and D. Pegden, 227–230. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Nakayama, M. K. 1994. Two-stage stopping procedures based on standardized time series. *Management Science* 40:1189–1206.

Priestley, M. B. 1981. *Spectral analysis and time series*. New York: Academic Press:

Sargent, R.G., K. Kang, and D. Goldsman. 1992. An investigation of finite-sample behavior of confidence interval estimators. *Operations Research* 40:898–913.

Schmeiser, B. 1982. Batch-size effects in the analysis of simulation output. *Operations Research* 30:556–568.

Schriber, T. J. and R. W. Andrews. 1984. ARMA-based confidence interval procedures for simulation output analysis. *Amer. J. Math. and Management Science* 4:345–373.

Schruben, L. W. 1983. Confidence interval estimation using standardized times series. *Operations Research* 31:1090–1108.

Steiger, N. M., and J. R. Wilson. 1999. Improved batching for confidence interval construction in steady-state simulation. In *Proceedings of the 1999 Winter Simulation Conference*, ed. P.A. Farrington, H.B. Nembhard, D.T. Sturrock, and G.W. Evans, 442–451. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

AUTHOR BIOGRAPHIES

E. JACK CHEN is a Senior Staff Specialist with BASF Corporation. He received an M.S. in computer science from Syracuse University, an M.B.A. from Northern Kentucky University, and Ph.D. degrees in Quantitative Analysis from the University of Cincinnati. His research interests are in the area of computer simulation, statistical data analysis and stochastic processes. His email and web addresses are <chenj@econqa.cba.uc.edu> and <http://www.econqa.cba.uc.edu/~chenj>.

W. DAVID KELTON is a Professor in the Department of Quantitative Analysis and Operations Management

at the University of Cincinnati. He received a B.A. in mathematics from the University of Wisconsin-Madison, an M.S. in mathematics from Ohio University, and M.S. and Ph.D. degrees in industrial engineering from Wisconsin. His research interests and publications are in the probabilistic and statistical aspects of simulation, applications of simulation, and stochastic models. He is co-author of Simulation Modeling and Analysis (3d ed., 2000, with Averill M. Law), and Simulation With Arena (1998, with Randall P. Sadowski and Deborah A. Sadowski), both published by McGraw-Hill. Currently, he serves as Editor-in-Chief of the INFORMS Journal on Computing, and has been Simulation Area Editor for Operations Research, the INFORMS Journal on Computing, and IIE Transactions, as well as Associate Editor for Operations Research, the Journal of Manufacturing Systems, and Simulation. From 1991 to 1999 he was the INFORMS co-representative to the Winter Simulation Conference Board of Directors and was Board Chair for 1998. In 1987 he was Program Chair for the WSC, and in 1991 was General Chair. His email and web addresses are <david.kelton@uc.edu> and <http://www.econqa.cba.uc.edu/~keltond>.