Generation of Simulation Input Scenarios Using Bootstrap Methods
Author(s): O. F. Demirel and T. R. Willemain

Source: *The Journal of the Operational Research Society*, Vol. 53, No. 1 (Jan., 2002), pp. 69-78
Published by: Palgrave Macmillan Journals on behalf of the Operational Research Society
Stable URL: http://www.jstor.org/stable/822880
Accessed: 25-08-2016 12:32 UTC

Journal of the Operational Research Society (2002) 53, 69–78 ©2002 Operational Research Society Ltd. All rights reserved. 0160-5682/02 $15.00
www.palgrave-journals.com/jors

Generation of simulation input scenarios using bootstrap methods

OF Demirel and TR Willemain*
Rensselaer Polytechnic Institute, Troy, NY, USA

Simulation modellers frequently face a choice between fidelity and variety in their input scenarios. Using an historical trace provides only one realistic scenario. Using the input modelling facilities in commercial simulation software may provide any number of unrealistic scenarios. We ease this dilemma by developing a way to use the moving blocks bootstrap to convert a single trace into an unlimited number of realistic input scenarios. We do this by setting the bootstrap block size to make the bootstrap samples mimic independent realizations in terms of the distribution of distance between pairs of inputs. We measure distance using a new statistic computed from zero crossings. We estimate the best block size by scaling up an estimate computed by analysing subseries of the trace.

Journal of the Operational Research Society (2002) **53,** 69–78. DOI: 10.1057/palgrave/jors/2601251

Keywords: simulation; statistics

Introduction

Simulation is the preferred operational research technique for complex problems that do not lend themselves to mathematical analysis. We refer to simulation inputs as scenarios. We focus on trace-driven simulations, ie those derived from a single historical record of the input to the system being simulated. We limit ourselves here to traces that are univariate and stationary. We exclude data with trends, seasonality or long memory, though the first two features can often be removed by differencing and seasonal differencing, respectively.

Trace-driven simulations are realistic because they are driven by a record of actual events. However, they cannot explore the full range of system responses because they do not contain all possible contingencies. Pritsker[1] suggested the expedient of dividing the single trace into blocks, but this is unsatisfactory because each scenario is then shorter than the trace; this would be a problem, for instance, if one wanted to simulate a year's operation of a factory based on a year of data. Commercial simulation software provides parametric models for independent and identically distributed (iid) inputs. Since inputs are commonly autocorrelated, this gives users an unlimited number of unrealistic scenarios. Rather than accept this tradeoff of fidelity for variety, we propose using a nonparametric time series bootstrap to convert the trace into an unlimited number of realistic inputs.

A new task to the bootstrap—generating input scenarios

The bootstrap is a method of computational inference based on resampling a data set. It was developed to provide valid nonparametric inference for iid data in nonstandard situations.[2,3] More recently, the bootstrap has been applied to stationary time series data with short-range dependence, where capturing autocorrelation is critical.[4–7]

There has been a limited amount of prior research relating the bootstrap to simulation output analysis.[7–12] However, our focus is on simulation input analysis.

Performance evaluation for bootstrap samples

It is important to develop a criterion for assessing the performance of a bootstrap method in generating samples from a single trace. With a single exception, all the statistical literature on the bootstrap[2,13–15] focuses on the properties of the statistics computed from bootstrap samples (eg consistency, asymptotic normality) rather than on the properties of the bootstrap samples themselves. The one exception was Efron and Tibshirani's[3] informal visual assessment of a few bootstrap samples.

One basic test of a bootstrap method in the context of generating input scenarios is a Turing test:[16] Is it possible to distinguish independent sample realizations from bootstrap samples? The answer should be no. What is required is the right mixture of variety across samples but fidelity to the essential characteristics of the observed data. In principle, we want to recreate the ideal situation in which we have multiple independent realizations of the process. Operationally, this means that we want to construct bootstrap samples that have the same degree of similarity and difference vis à

**Correspondence: TR Willemain, Department of Decision Sciences and Engineering Systems, Rensselaer Polytechnic Institute, 110 8th Street, Troy, NY, 12180-3590, USA.*
E-mail: willet@rpi.edu

vis the original series as would be true of independent samples. Elsewhere, we have shown that passing this statistical version of a Turing test implies passing a visual Turing test as well.[16]

We have developed a new measure of the distance between two time series, which operationalizes the notion that independent replications have a characteristic distribution of distance from each other. By tuning the bootstrap properly, we can make the distribution of this distance for bootstrap samples closely match that of independent samples.

Selecting the size of the resampling unit

The conventional bootstrap method uses an individual observation as the resampling unit for iid data, ie it resamples with replacement from the original observations. From the perspective of time series analysis, this creates white noise. To preserve autocorrelation present in the original data, all nonparametric approaches to bootstrapping time series data resample chunks of consecutive observations. The moving blocks bootstrap divides the data into overlapping blocks of fixed size. It creates bootstrap samples by concatenating blocks chosen at random with replacement. This paper is about how to determine the block size for scenario generation.

Review of methods for generating autocorrelated simulation inputs

Input modelling is a vital part of simulation methodology.[17] Successful input models mimic the underlying probabilistic mechanism of the system drivers,[18] which commonly involves autocorrelation.[19] There is no known general-purpose method for generating dependent input processes. Guided by available software, practitioners usually ignore dependence in inputs when present, resulting in unrealistic simulation outputs.[20]

Several parametric methods have been developed for generating autocorrelated inputs to simulations.[19–26] The main difficulty with parametric methods is the need to identify a reasonable model for the data.[27] Classical autoregressive-moving average (ARMA) models are not able to mimic basic features of many real time series, and fitting models more complex than ARMA can be difficult.[4] The only previous proposal we know of for a nonparametric approach is Cheng's[12] suggested use of the bootstrap, which unfortunately assumed iid data.

There are two broad approaches to bootstrapping dependent data: model-based (parametric) and model-free (nonparametric). A parametric model will yield superior results when correct but inferior results when incorrect. A nonparametric approach avoids this model risk with little loss of efficiency. We use a nonparametric method.

The parametric bootstrap first fits a model to the dependent data, then resamples residuals instead of the original data. Unlike classical ARMA models, the parametric bootstrap does not assume normality of the residuals. A bootstrap sample is generated using the resampled residuals and the estimated parameters.[4,28–31]

There are several alternative nonparametric approaches to bootstrapping autocorrelated data.[6,7,32–34] In this paper, we use the most prominent, the moving blocks bootstrap (MBB).[4,5] Let $X_1, X_2, \ldots, X_n$ be a stationary time series and B_i be the block of b consecutive observations starting from the ith. The $B_i = (X_i, \ldots, X_{i+b-1})$, $i = 1, 2, \ldots, n - b + 1$, form $n - b + 1$ overlapping blocks from the original sample. Resampling n/b blocks with replacement from the set $\{B_1, B_2, \ldots, B_{n-b+1}\}$ produces the bootstrap sample. Using blocks as resampling units preserves autocorrelation within blocks. The assumption is that, for block size b, observations more than b time units apart are essentially independent. The MBB becomes the conventional bootstrap when $b = 1$.

One must carefully select the block size: too big loses variety, too small loses fidelity. Hall *et al*[35] established that the best block size depends on three factors: the autocorrelation structure, the series length n, and the reason for bootstrapping. The optimal block size is proportional to $n^{1/3}$ for estimating the bias or variance of a sample mean, $n^{1/4}$ for estimating one-tail probabilities, and $n^{1/5}$ for estimating two-tail probabilities. We found the optimal block size to be roughly proportional to $n^{1/2}$ for scenario generation.

Index of difference Δ

The Δ statistic is a new measure of the distance between two sample time series. It is based on the characterization of series by higher order crossing (HOC) counts.[36] The Δ statistic reduces the problem of comparing two series with n observations to a simple one-dimensional problem. Kedem[36] showed that matching HOC was theoretically equivalent to matching autocorrelation functions or spectra, but HOC have some practical advantages, including greater resistance to outliers and easier selection of parameters.

Higher order crossings (HOC)

Higher order crossings (HOC) are counts of the zero crossings made by the mean-centred series and its first, second, and higher differences. Any stationary process can be characterized by its HOC. Figure 1 illustrates the computations through the second difference. Kedem found that using nine HOC was sufficient to characterize most short memory time series for some purposes; we found ten to be adequate for scenario generation.

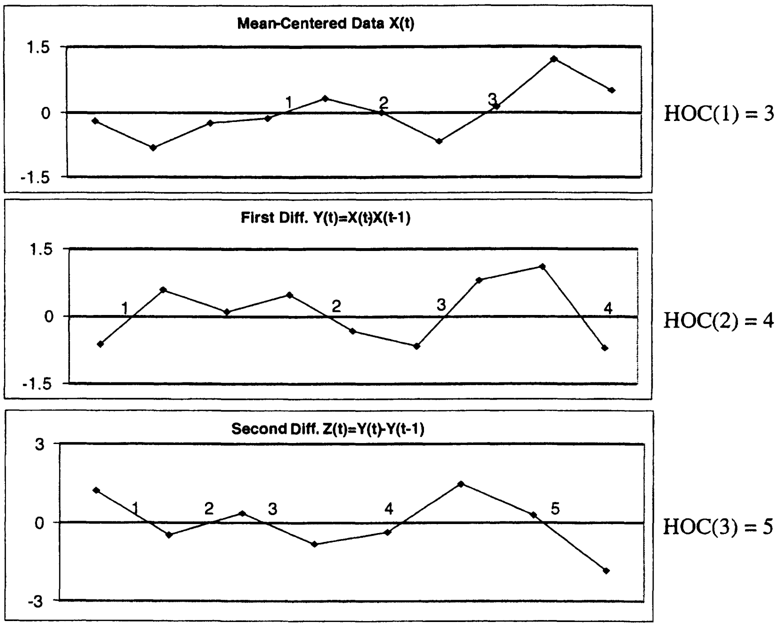

Figure 1 Illustration of the calculation of higher order crossing counts.

The Δ statistic

Given a series of length n and several HOC D_k, $k = 1, \ldots, K$, Kedem defined the differences between successive counts δ_k as D_1 for $k = 1$, $D_k - D_{k-1}$ for $k = 2, \ldots, K - 1$, and $(n - 1) - D_{k-1}$ for $k = K$. Defining the expected values $E[\delta_k] = m_k$, he formed the difference measure

$$\psi^2 = \sum(\delta_k - m_k)^2/m_k$$

for testing the hypothesis that a given series arises from a process with specified expected values $\{m_k\}$. When we attempted to use the ψ^2 statistic, we found that the denominator terms m_k were often unknown and sometimes estimated as zero in shorter series, creating a divide-by-zero problem.

We elaborated on Kedem's ψ^2 statistic to develop an alternative measure, Δ. The statistic Δ is the sum of squared standardized differences between the coordinates of the two series in normalized HOC space. Given a series of length n and maximum order of difference $K - 1$, the largest possible number of sign changes is $n - K$. Let D_{ki} and D_{kj} be the kth HOC for the ith and jth series. To standardize across series lengths, define the normalized HOC

$$d_k = D_k/n - K$$

Regarding the d_k's as proportions, we worked by analogy to the classical Z test for differences to define Δ:

$$\Delta = \sum Z_k^2, \qquad k = 1, \ldots, K$$

where

$$Z_k = (d_{ki} - d_{kj})/\mathrm{SE}\{d_{ki} - d_{kj}\}$$

and

$$\mathrm{SE}\{d_{ki} - d_{kj}\} = \{[d_{ki}(1 - d_{ki}) + d_{kj}(1 - d_{kj})]/n - K\}^{1/2}$$

Distribution of Δ

If successive normalized differences d_k and d_{k+1} were independent and the HOC reasonably large, the distribution of Δ would be approximately chi-square, since it would be the sum of squares of variables (the Z_k) that are approximately standard normal. However, successive normalized differences are correlated, so the sampling distribution of Δ must be examined empirically.

Design of experiment to determine the distribution of Δ

All experiments reported here were programmed using the FORTRAN 77 programming language and NAG subroutines[37] for random number generation.

We generated 1000 pairs of series, each with $n = 1000$ observations. From each pair, we computed a value of Δ using the first $K = 10$ HOC, producing 1000 values of Δ. We used these 1000 values to empirically characterize the distribution of Δ.

We simulated autoregressive/moving-average (ARMA) processes of order (p,q):

$$X_t = \phi_1 X_{t-1} + \cdots + \phi_p X_{t-p} + Z_t + \theta_1 Z_{t-1} + \cdots \theta_q Z_{t-q}$$

where the Z_t's are iid random variables with a standard normal distribution. The five ARMA (p,q) models used were:

$$\text{ARMA}(1,0)\text{: } X_t = 0.9X_{t-1} + Z_t$$
$$\text{ARMA}(1,0)\text{: } X_t = -0.9X_{t-1} + Z_t$$
$$\text{ARMA}(5,0)\text{: } X_t = 0.4X_{t-1} + 0.2X_{t-2} + 0.1X_{t-3} + 0.1X_{t-4} + 0.1X_{t-5} + Z_t$$
$$\text{ARMA}(0,4)\text{: } X_t = Z_t + Z_{t-1} + Z_{t-2} + Z_{t-3} + Z_{t-4}$$
$$\text{ARMA}(1,1)\text{: } X_t = 0.9X_{t-1} + Z_t + 0.9Z_{t-1}$$

Null distribution of Δ for two series from the same processes

The null distribution is the distribution when both members of the pair are realizations from the same process. The experimental results indicated that the null distribution of Δ is well approximated by a lognormal with parameters that depend on the type of ARMA process and, to a lesser extent, on the parameter values of the process.

We used the Kolmogorov–Smirnov (K–S) one sample test to make a formal test of the lognormality of Δ. The results are shown in Table 1. All the P-values were greater than 0.50, which confirmed that the lognormal distribution is a good fit.

Alternative distribution of Δ from mismatched pairs of AR(1) processes

If Δ is to be a good differentiator between processes, it should be sensitive to small parameter differences even within the same series type. We studied pairs of AR(1) series to see whether the distribution of Δ changed significantly from its null distribution when the two series had different values of the parameter ϕ. Figure 2 shows boxplots of 300 Δ values from pairs of AR(1) series, where one member of the pair had $\phi = 0.9$ and the other value of ϕ ranged from $+0.9$ to -0.9. The distributions of Δ from mismatched pairs were also well approximated by the lognormal but with means and standard deviations quite different from those in the null case. These differences increased with the difference in parameter values, as one would hope.

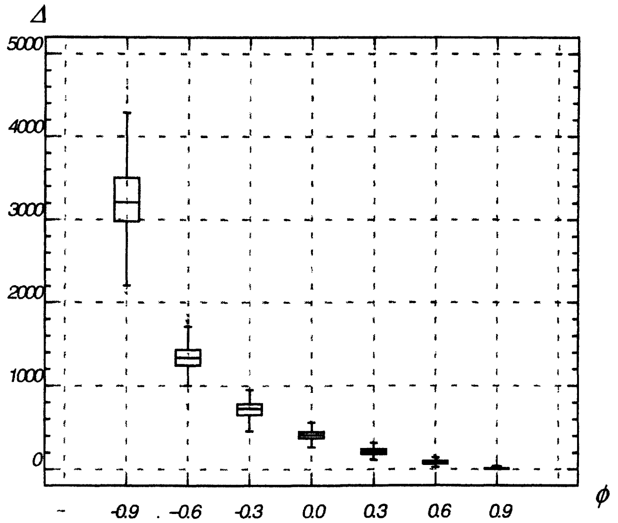

Figure 2 Boxplots of Δ computed from 300 pairs of AR(1) series with $\phi = 0.9$ *vs* $\phi = 0.9, 0.6, 0.3, 0.0, -0.3, -0.6, -0.9$ $(n = 1000, K = 10)$.

Determining bootstrap block size

As shown above, one can compare two data sources by their distribution of Δ. Let a and i be two sample realizations generated by the same stationary process, and let a^* be a bootstrap sample of series a. Let Δ be the distance between series a and i and Δ^* the distance between series a and its bootstrap sample, a^*. From the perspective of a Turing test, using the MBB to generate scenarios requires selecting the block size to minimize the difference between the distributions of Δ and Δ^*.

Hall *et al*[35] provided a practical approach to selecting the best block size in conventional applications of the MBB. Their method relied on breaking the data into subseries. We adapted this approach, as explained below.

Table 1 Kolmogorov-Smirnov one-sample test to lognormal distribution for Δ from 1000 pairs of 5 ARMA models $(n = 1000, K = 10)$

	Fitted lognormal distribution		*Kolmogorov-Smirnov test*	
Series type	*Mean*	*Stdev.*	*K-S statistic*	*Significance level*
ARMA(1,0) ($\phi = 0.9$)	8.8694	7.3979	0.0259	0.5087
ARMA(1,0) ($\phi = -0.9$)	13.8677	20.2336	0.0195	0.8433
ARMA(5,0) ($\phi_1 = 0.4,\ \phi_2 = 0.2,\ \phi_3 = 0.1,\ \phi_4 = 0.1,\ \phi_5 = 0.1$)	10.8266	10.3553	0.0251	0.5549
ARMA(0,4) ($\theta_1 = -1.0,\ \theta_2 = -1.0,\ \theta_3 = -1.0,\ \theta_4 = -1.0$)	12.3394	11.0763	0.0201	0.8122
ARMA(1,1) ($\phi = 0.9,\ \theta = -0.9$)	6.1827	4.4172	0.0210	0.7695

Experiment to determine best block size

We conducted a Monte Carlo experiment to see how the performance of the MBB depended on the choice of block size. We again used data from the five ARMA models.

We created 1000 sets of three series, each with $n = 1000$ observations. The first two series in each set were independent samples of the same ARMA process (called series *a* and *i* above). The third series was an MBB sample created from the second series (series a^*). Each triple produced two matched pairs of difference statistics, Δ and Δ^*. The distribution of the 1000 values of Δ established the 'gold standard' for independent samples. The distribution of the 1000 values of Δ^* was intended to mimic the gold standard as closely as possible. The best block size was the one that led to the best match, as measured by the K–S two-sample statistic. As it was impossible to find the optimal block size analytically, we used numerical search to estimate the block size empirically.

Figure 3 illustrates that there is indeed an optimal block size, using data for the AR(1) process with $\phi = 0.9$. It plots the relationship between block size and the K–S statistic *D*, defined as the maximum absolute discrepancy between the empirical distribution functions of Δ and Δ^*. Similar plots were observed for the other ARMA processes. Though there are small random variations, *D* decreased to a certain block size, after which it increased. The block size that gives the minimum *D* is the optimal block size, eg 12 in this example. Table 2 shows that the optimal block sizes ranged from 8–54 observations, depending on the type of data. These differences are consistent with the fact that Hall *et al*[35] found the optimal block size for conventional bootstrap analyses to depend on the correlation structure of the series.

We were interested not only in where the best match occurred, but also in how close a match was made. Table 2 also shows that, for AR(1) and ARMA(1,1) data, the K–S statistics were small, with corresponding large *P*-values, indicating an excellent match. The results of the AR(5) and MA(4) series were not as good. With these more complex data series, Δ^* has less variety than Δ; that is, the means of the lognormal distribution of Δ^* were well matched but the standard deviations were too small. Note, however, that an imperfect match does not equate to failure, since most of the variability is captured by the bootstrap, and that is much preferable to capturing only what is available in the trace.

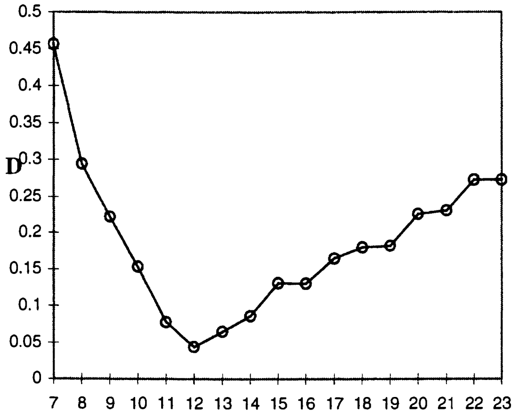

Figure 3 *D*, the difference between the CDFs of Δ and Δ^*, is represented as a function of block size. The optimal block size is 12. The underlying process is AR(1) $\phi = 0.9$. 1000 values of Δ and Δ^* are computed respectively. ($n = 1000, K = 10$).

Table 2 Performance of MBB in matching the distribution of Δ ($n = 1000$)

Series type	*Optimal block size*	*K–S statistic*	*P-value*
AR(1), $\phi = 0.9$	12	0.044	0.270 34
AR(1), $\phi = -0.9$	26	0.047	0.219 33
AR(5), $\phi = 0.4,\ 0.2,\ 0.1,\ 0.1,\ 0.1$	8	0.088	0.000 76
MA(4), $\theta = -1.0,\ -1.0,\ -1.0,\ -1.0$	9	0.106	0.000 02
ARMA(1,1) $\phi = 0.9, \theta = -0.9$	54	0.037	0.475 77

Experiment to relate optimal block size to series length

To explore the relationship between block size and series length, we conducted another experiment. We used the same five ARMA models with series lengths *n* varying from 200 to 6400. As in the previous experiment, 1000 values of Δ and Δ^* were generated for each experimental combination of series type and series length.

The results are shown in Table 3. The optimal block size is clearly a function of both series type and series length. We can represent the optimal block size quite well as a power function of the series length. Letting *b* be the block size and *n* the series length; the power function relationship is

$$b = cn^p$$

where *c* depends on the series type, but *p* is constant. This model implies that a log–log plot of the data in Table 3 will form parallel straight lines with common slope and series-specific intercepts.

Figure 4 confirms this expectation: the plotted lines are nearly straight and parallel. Table 4 summarizes least squares regression fits, all with *R*-square above 99%. The estimated slopes were all in the vicinity of 0.5. For convenience, we assume $p = 0.5$. This means that the optimal block size is roughly proportional to the square root of the series length. We use this square root relationship in a subseries technique to scale up the block size estimated from a portion of the data series to an optimal block size for the entire series.

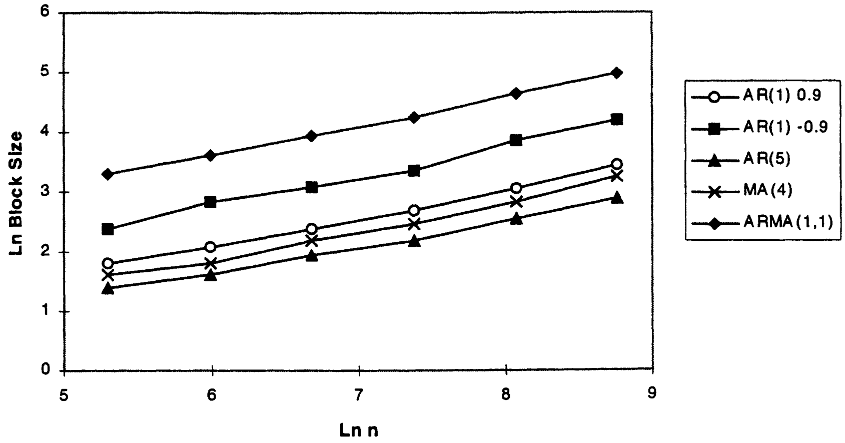

Figure 4 Plot of logarithm of optimal block sizes *vs* logarithm of lengths of series for 5 ARMA(p,q) series.

Subseries technique

Unlike in our Monte Carlo experiment, there is usually only a single trace in real-world problems, making it important to be able to tune the bootstrap using only one observed series. Our solution to this problem involves comparing the distribution of Δ computed from subseries of the original data against the corresponding distribution computed from bootstrapping the subseries. Lacking independent replications, the subseries provide the only available substitute. This approach is in the spirit of Carlstein[38] and Hall *et al.*[35]

We divide the observed series into s non-overlapping subseries, each consisting of m observations, such that $m = \lfloor n/s \rfloor$. Assume for simplicity that m is an integer. By treating these s subseries as independent samples, we can obtain $\binom{s}{2}$ values of Δ. These Δ values are reference values. We then fix a block size and form r bootstrap samples of each subseries and compute sr values of Δ^* between each of the subseries and its bootstrap samples. The best subseries block size $\hat{b}_m$ is that which best matches the distributions of Δ and Δ^*.

Scaling up from subseries

The final step is to scale up the estimate of the best block size for a subseries to match the length n of the entire series using the approximate square root relationship. The estimate of the optimal block size for the whole series with n observations is

$$\hat{b}_n = (n/m)^{1/2}\, \hat{b}_m = s^{1/2}\, \hat{b}_m$$

Table 3 Optimal block sizes for various lengths of 5 ARMA(p,q) series; results based on 1000 series

Length	*AR(1) 0.9*	*AR(1) − 0.9*	*AR(5)*	*MA(4)*	*ARMA(1,1)*
200	6	11	4	5	27
400	8	17	5	6	37
800	11	22	7	9	52
1600	15	29	9	12	71
3200	21	47	13	17	103
6400	31	66	18	26	146

Method to estimate best block size

We now summarize the subseries technique for finding the best block size for a given series.

1. Divide the original series into $s \geq 5$ subseries with m observations each. (Choosing $s = 5$ produces 10 reference values of Δ. In our experience, 10 is the minimum number of reference values needed to obtain trustworthy results. There is a tradeoff here: more subseries provide a smoother estimate of the reference distribution, but subseries that contain too few observations yield unstable estimates of Δ.)
2. Compute $\binom{s}{2}$ values of Δ using all pairs of subseries.
3. Create $r = 1000$ or more bootstrap samples of each subseries, beginning with a block size of one and incrementing by one. (More bootstrap samples give better results. It is important to create at least 1000 bootstrap samples, especially if the original series is shorter than 1000 observations.)
4. For each block size, compute the r values of Δ^* between each of the subseries and its bootstrap samples.
5. Compare the sample distributions of Δ and Δ^* using the K–S two-sample test.
6. The block size which gives the minimum K–S statistic value D is the best block size for subseries, $\hat{b}_m$.
7. Estimate the best block size for the whole series by using the square root rule, $\hat{b}_n = s^{1/2}\hat{b}_m$.

This rule appears to overestimate slightly, so we recommend rounding down to the nearest integer. This is consistent with the fact that the slope estimates in Table 4 are mostly slightly below 0.5.

Table 4 Fits to log of optimal block sizes versus log of series lengths for five ARMA(p,q) series; intercept, slope and R square values are computed with least squares regression. Values in parentheses are estimated standard errors

	AR(1) 0.9	*AR(1) − 0.9*	*AR(5)*	*MA(4)*	*ARMA(1,1)*
Intercept	−0.733 (0.082)	−0.273 (0.160)	−0.984 (0.105)	−1.016 (0.166)	0.697 (0.047)
Slope	0.471 (0.012)	0.506 (0.022)	0.439 (0.015)	0.480 (0.023)	0.487 (0.007)
R square	0.998	0.992	0.995	0.991	0.999

Experiment to evaluate the subseries method

We evaluated the subseries method in another experiment. For each of the five ARMA models, we generated ten sample series. Each model was assigned a particular series length, number of subsamples, and number of bootstrap samples.

The results are shown in Table 5. For ten sample series from each model, we obtained either the optimal value of the optimal block size or a value near the optimal. For example, consider Table 5(a). Of the ten series analysed, 5 had an estimated optimal block size of 3 for the subseries, which were of length $200/5 = 40$. Scaling up by the factor $\sqrt{s} = \sqrt{5}$ produced estimates of 6.71, which we round down to 6, which is the optimal value in this case. Three of the 10 series produced rounded estimates of 4, while the remaining two produced rounded estimates of 8. In every case in Table 5, the most frequently occurring estimated block size was the optimal block size.

Application to a real dataset

We developed the procedure for bootstrapping simulation inputs using artificial data from the ARMA family because it was easy to generate such data and control its characteristics. However, the data available in applications are unlikely to be so conveniently simple. Accordingly, we now show an application to a more realistic dataset. The data are transformations of daily New York Stock Exchange prices (adjusted for splits) of the common stock of the General Electric Corporation (GE). There are 1240 observations, corresponding to the period from 2 February 1996 to 29 December 2000. These transformed data are part of an

Table 5 Estimation of optimal block size by subseries technique

(a) AR(1) $\phi = 0.9$, length of series $n = 200$, optimal block size = 6, number of subseries $s = 5$, $r = 4000$ bootstrap samples formed from each subseries				
Best block size of subseries	2	3	4	
Frequency	0.30	0.50	0.20	
Estimated optimal block size of series	4	6	8	
(b) AR(1) $\phi = -0.9$, $n = 800$, optimal = 22, $s = 20$, $r = 1900$				
Best block size of subseries	4	5	6	7
Frequency	0.30	0.40	0.20	0.10
Estimated optimal block size of series	17	22	26	31
(c) AR(5), $n = 500$, optimal = 6, $s = 5$, $r = 4000$				
Best block size of subseries	2	3	4	
Frequency	0.40	0.50	0.10	
Estimated optimal block size of series	4	6	8	
(d) MA(4), $n = 400$, optimal = 6, $s = 5$, $r = 4000$				
Best block size of subseries	2	3	4	
Frequency	0.20	0.60	0.20	
Estimated optimal block size of series	4	6	8	
(e) ARMA(1,1), $n = 400$, optimal = 37, $s = 10$, $r = 1800$				
Best block size of subseries	11	12	13	
Frequency	0.20	0.70	0.10	
Estimated optimal block size of series	34	37	41	

(1) Estimated optimal block sizes are computed by square root rule and rounded down.

(2) True optimal block sizes are estimated from 2000 values of Δ and Δ^*.

(3) Ten series are generated from each ARMA model.

(4) When the series were shorter and the number of subseries were fewer, we generated more bootstrap samples (4000 instead of 1800 or 1900). The reason is that having fewer Δ reference values resulted in more unstable results, and we could reduce instability by generating more bootstrap samples.

ongoing study of stock price movement scenarios. The data were produced by a normalization process that removed nonstationarity in the mean by subtracting a 10-day moving average of 10 and then removed nonstationarity in the variance by dividing the differences by a 10-day rolling standard deviation. We used 10 subseries of 124 points each and estimated an optimal block size of 22.

Figures 5 and 6 show the success of the bootstrap at creating new time series that resemble the original data. The timeplots in Figure 5 show that the 'look' of the data was properly preserved. The autocorrelation and partial autocorrelation plots in Figure 6 show that the correlation structure of the bootstrap series matches that of the original data.

Conclusions

We developed a new method of converting a single historical trace into an unlimited number of simulation input scenarios. These series might be used as inputs to a discrete event simulation of a factory, as part of the risk assessment for a new financial instrument, or as scenarios for training human decision makers. Our method amounts to a procedure for choosing the block size in the moving blocks bootstrap.

Our goal was to match, among the bootstrap samples, the distribution of distances between pairs of independent samples from the same data generating process. We operationalized the notion of the distance between two sample

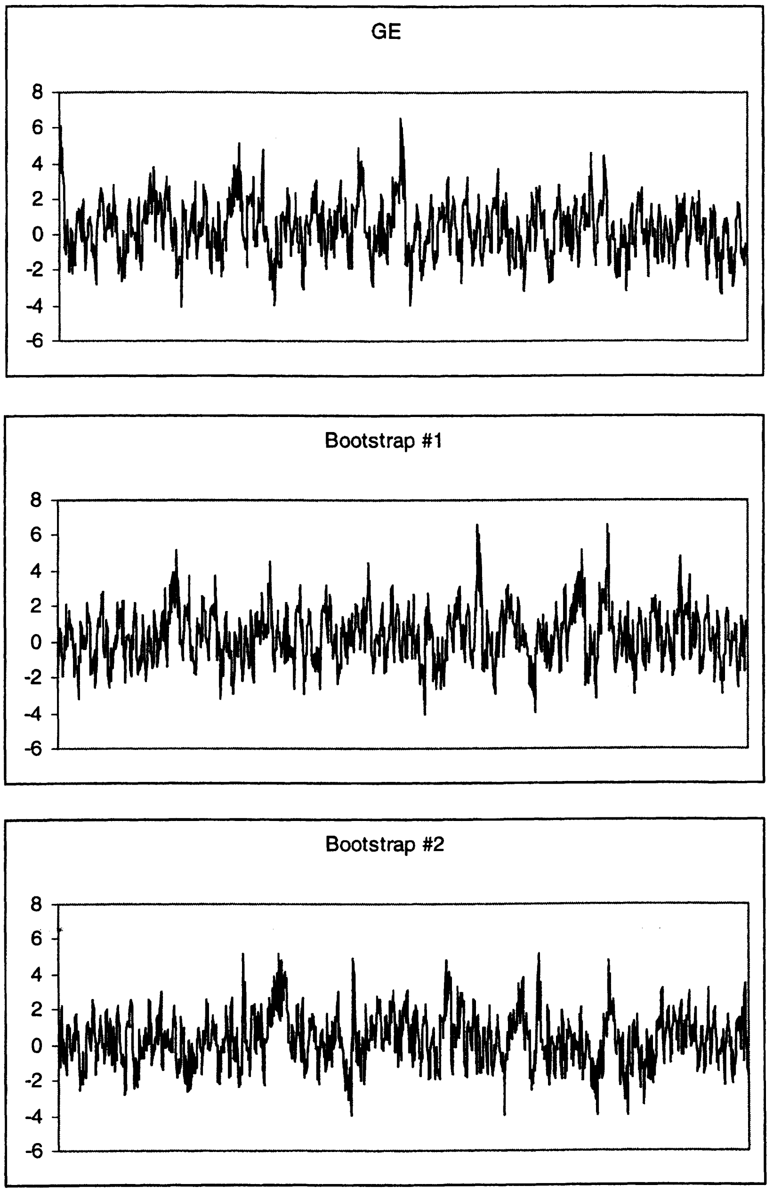

Figure 5 Timeplots of data derived from GE stock prices and two bootstrap replications.

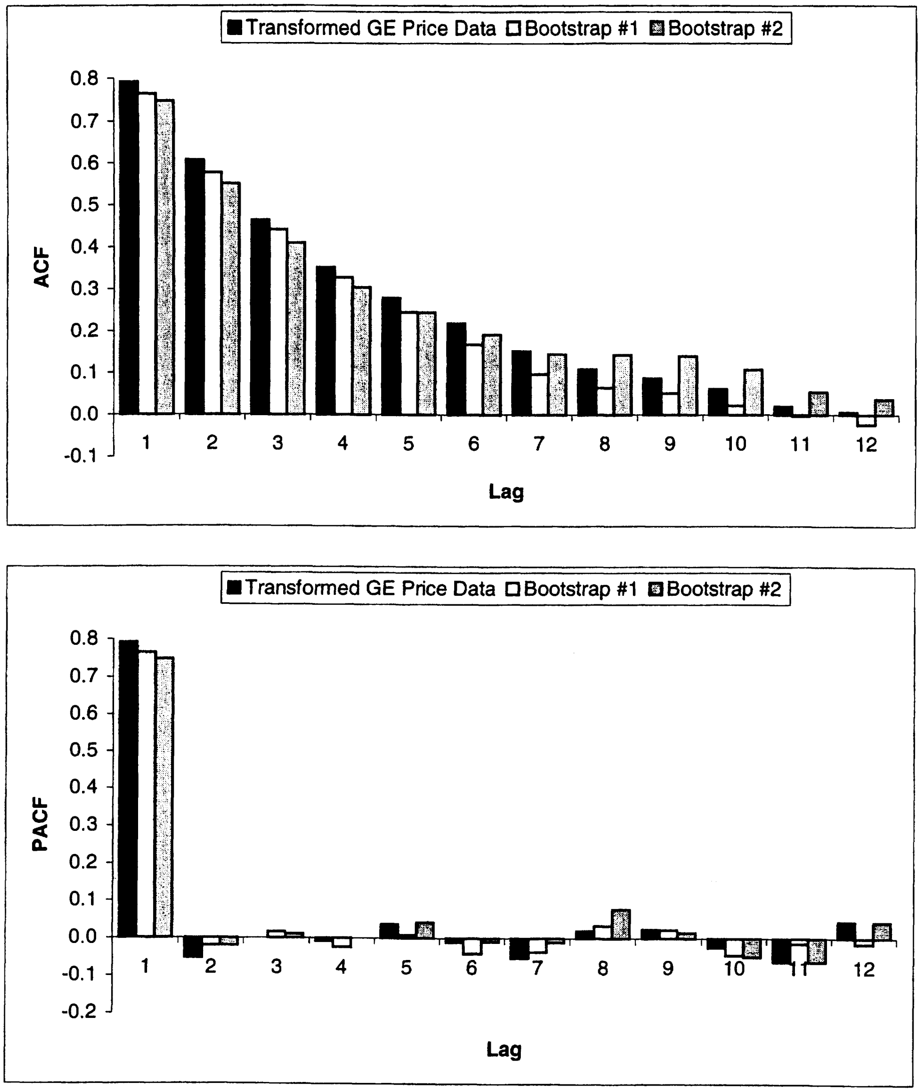

Figure 6 Autocorrelations and partial autocorrelations of GE price data and two bootstrap replications.

time series using the Δ statistic, which is based on the count of zero crossings by the mean-centred data and its differences.

Using Monte Carlo simulations of ARMA data, we discovered an approximate square root law relating the choice of block size to the length of the series. We exploited this relationship in a practical procedure to estimate the best block size from a single trace. The procedure divides the trace into subseries, determines the best block size for the subseries, then uses the square root relationship to scale up the estimate for series as long as the trace.

This work has significance for simulation analysis. Applications of statistical methods to simulation have concentrated more on output analysis than input analysis. In contrast, we focus on the relatively neglected input side of the simulation process. By providing a way to generate input scenarios that are both numerous and realistic, we allow the simulation analyst to better evoke a full range of output behaviours from the system being simulated.

This work also has significance for bootstrapping. Heretofore, the bootstrap has been judged by the properties of the statistics computed from the bootstrap samples. In contrast, we focus on the properties of the bootstrap samples themselves. We measure their quality in terms of matching the distribution of a measure of distance between pairs of independent samples from the same data generating process. The distance measure itself is a contribution. We also discovered that the optimal block size for indirect inference is proportional to the square root of the length of the data series, which is a larger power than has been found appropriate for other tasks.

Acknowledgements—The authors have benefited from conversations with L Baechle, R Bress, F Cai, J Haddock, Y Kim, D Park, M Raghavachari, G Runger, N Shang and P Sullo. We also benefited from suggestions provided by two referees. This research was supported by the US National Science Foundation through Grant DMI-9813097.

References

1 Pritsker AAB (1986). *Introduction to Simulation and SLAM II.* Haltsted Press: New York.

2 Efron B (1979). Bootstrap methods: another look at the jackknife. *Ann Stat* **7**: 1–26.

3 Efron B and Tibshirani R (1993). *An Introduction to the Bootstrap.* Chapman and Hall: New York.

4 Kunsch H (1989). The jackknife and the bootstrap for general stationary observations. *Ann Stat* **17**: 1217–1241.

5 Liu R and Singh K (1992). Moving blocks jackknife and bootstrap capture weak dependence. In: LePage R and Billard L (eds). *Exploring the Limits of Bootstrap.* Wiley: New York, pp 225–278.

6 Politis DN and Romano JP (1994). The stationary bootstrap. *J Am Stat Assoc* **89**: 1303–1313.

7 Park D and Willemain TR (1999). The threshold bootstrap and threshold jackknife. *Comput Stat Data Anal* **31**: 187–202.

8 Rea C, Shiue W and Xu C (1991). Applying bootstrap method to simulation output analysis. *Proceedings of the 23rd Symposium on the Interface. Computing Science and Statistics, Seattle, WA.* Interface FNA: VA, pp 82–85.

9 Shiue W, Xu C and Rea C (1992). Bootstrap confidence intervals for simulation outputs. *J Stat Comput Simul* **45**: 249–255.

10 Kim Y, Haddock J and Willemain TR (1993). The binary bootstrap: inference with autocorrelated binary data. *Commun Stat Simul Comput* **22**: 205–216.

11 Park D, Kim Y, Shin K and Willemain TR (2001). Simulation output analysis using the threshold bootstrap. *Eur J Opl Res* **134**: 17–28.

12 Cheng RCH (1995). Bootstrap methods in computer simulation experiments. In: Alexopoulos C, Kang K, Lilegdon W and Goldsman D (eds). *Proceedings of the 1995 Winter Simulation Conference.* IEEE: Piscataway, NJ, pp 171–177.

13 Bickel P and Freedman D (1981). Some asymptotic theory for the bootstrap. *Ann Stat* **9**: 1196–1217.

14 Efron B and Tibshirani R (1986). Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy. *Stat Sci* **1**: 54–77.

15 Hall P and Jing B (1996). On sample reuse methods for dependent data. *J R Stat Soc* B **58**: 727–737.

16 Demirel OF and Willemain TR (2000). Turing tests of bootstrap scenarios. Submitted for publication.

17 Law AM, McGomas MG and Vincent SG (1994). The crucial role of input modeling in successful simulation studies. *Indust Engng* **26**: 55–59.

18 Leemis L (1995). Input modeling for discrete events simulation. In: Alexopoulos C, Kang K, Lilegdon W and Goldsman D. (eds). *Proceedings of the 1995 Winter Simulation Conference.* IEEE: Piscataway, NJ, pp 16–23.

19 Melamed B, Hill JR and Goldsman D (1992). The TES methodology: modeling empirical stationary time series. In: Swain JJ, Goldsman D, Crain RC, and Wilson JR (eds). *Proceedings of the 1992 Winter Simulation Conference.* IEEE: Piscataway, NJ, pp 16–23.

20 Cario MC and Nelson BL (1996). Autoregressive to anything: time-series input processes for simulation. *Opns Res Lett* **19**: 51–58.

21 Gaver DP and Lewis PAW (1980). First-order autoregressive gamma sequences and point processes. *Adv Appl Prob* **12**: 727–745.

22 Lawrence AJ and Lewis PAW (1980). The exponential autoregressive-moving average EARMA (p,q) process. *J R Stat Soc* B **42**: 150–161.

23 Lawrence AJ and Lewis PA (1981). A new autoregressive time series model in exponential variables (NEAR(1)). *Adv Appl Prob* **13**: 826–845.

24 Melamed B (1991). TES: a class of methods for generating autocorrelated uniform variates. *ORSA J Comput* **3**: 317–329.

25 Nelson B *et al* (1995). Input modeling when simple models fail. In: Alexopoulos C, Kang K, Lilegdon W and Goldsman D (eds). *Proceedings of the 1995 Winter Simulation Conference.* IEEE: Piscataway, NJ, pp 93–100.

26 Cario MC (1996). Modeling and simulating time series input processes with ARTAFACTS and ARTAGEN. In: Charnes JM, Morrice DJ, Brunner DT and Swain JT (eds). *Proceedings of the 1996 Winter Simulation Conference.* IEEE: Piscataway, NJ, pp 207–213.

27 Carlstein E (1993). Resampling techniques for stationary timeseries: some recent developments. In: Brillinger D *et al* (eds). *New Directions in Time Series Analysis* Springer-Verlag: New York, pp 75–85.

28 Thombs LA and Schuacany WR (1990). Bootstrap prediction intervals for autoregression. *J Am Stat Accos* **85**: 786–792.

29 Thombs LA and Schucany WR (1990). Bootstrap prediction intervals for autoregression. *J Am Stat Assoc* **85**: 486–492.

30 Romano JP and Thombs LA (1996). Inference for autocorrelation under weak assumptions. *J Am Stat Assoc* **91**: 590–600.

31 Souza RC and Neto AC (1996). A bootstrap simulation study in ARMA(p,q) structures. *J Forecast* **15**: 343–353.

32 Lall U and Sharma A (1996). A nearest neighbor bootstrap for resampling hydrologic time series. *Water Res Res* **32**: 679–693.

33 Rajagopalan B, Lall U, Tarbotan G, and Bowles DS (1997). Multivariate nonparametric resampling scheme for generation of daily weather variables. *Stochastic Hydrol Hydraul* **11**: 65–93.

34 Carlstein E, Do KA, Hall P, Hesterberg T and Kunsch HR (1998). Matched-block bootstrap for dependent data. *Bernoulli* **4**: 305–328.

35 Hall P, Horowitz JL, and Jing B (1995). On blocking rules for the bootstrap with dependent data. *Biometrika* **82**: 561–574.

36 Kedem B (1993). *Time Series Analysis by Higher Order Crossings.* IEEE: Piscataway, NJ.

37 NAG Fortran Library (1995). The Numerical Algorithms Group Limited.

38 Carlstein E (1986). The use of subseries values for estimation the variance of a general statistic from a stationary sequence. *Ann Stat* **14**: 1171–1179.

Received July 2000;
accepted June 2001 after two revisions