Proceedings of the 1995 Winter Simulation Conference
ed. C. Alexopoulos, K. Kang, W. R. Lilegdon, and D. Goldsman

A TUTORIAL ON DESIGN AND ANALYSIS OF SIMULATION EXPERIMENTS

W. David Kelton

Department of Operations and Management Science, Carlson School of Management, and
Supercomputer Institute
University of Minnesota
Minneapolis, Minnesota 55455, U.S.A.

ABSTRACT

This paper describes, in general terms, methods to help design the runs for simulation models and interpreting their output. Statistical methods are described for several different purposes, and related problems like comparison, variance reduction, sensitivity estimation, metamodeling, and optimization are mentioned. The main point is to call attention to the challenges and opportunities in using simulation models carefully and effectively. This paper is an update of Kelton (1994).

1 INTRODUCTION AND SCOPE

Building a good simulation model can be a lot of work. You have to figure out how to model the system, express the model in whatever software you're using, collect data on the corresponding real system (if any) to set the simulation model's input parameters, verify that the simulation model, ıs expressed in the software, is working properly, and validate the simulation's output against the corresponding output from the real system (if any). After all that, you should feel pretty good.

But not *too* good. If you stop there you've wasted a lot of effort, since now it's the simulation model's turn to go to work for you. And I don't mean just running it (once) on your computer and looking at the results (which you doubtless did anyway just to get it to run). What you really have now is far more than just "a" simulation model—you have a great vehicle to test out a lot of different ideas without a lot more work on your part (although your computer will now get very busy, but that's good—and cheap, unlike your time invested to build the model), and to learn a lot about your model and the system it's simulating in terms of performance and possible improvement.

To do all this effectively, though, you have to think carefully about just how you're going to exercise your model. And, perhaps unfortunately, the most common kinds of simulation models can fool you (although not intentionally) if you're not circumspect about how you interpret their output.

My purpose in this tutorial is to call your attention to these issues and indicate in general terms how you can deal with them. I won't be going into great depth on a lot of technical details, but will refer you instead along the way to any of several texts on simulation that do, to introductory tutorials on this subject in the last few WSCs, and to more advanced and specialized WSC reviews.

Section 2 takes up the issue of randomness in simulation, Section 3 considers planning your runs, and Section 4 looks at the role of time in simulation. Analysis of a single variant is described in Section 5, and of alternative variants in Section 6. Sections 7–10 touch on variance reduction, sensitivity estimation, metamodels, and simulation optimization.

2 DIDO VS. RIRO

Some simulations take as input only fixed, nonrandom values, typically representing parameters that describe the model and the particular variant of it you're evaluating. If the system you're simulating is really like this, then you can get by with such a *deterministic* simulation model. The nicest thing about this is, since there's no randomness in the input, there's no randomness in the output either—if you repeat the simulation you'll get the same thing over again. Thus, your answers are exact, at least up to roundoff. Figure 1 illustrates the idea in a manufacturing example, where the inputs are the machine cycle times, the interarrival times between successively arriving batches of parts, and the sizes of the these batches; the outputs are the hourly production and the machine utilization. The big dots for the inputs represent their (deterministic) values, and the big dots for the outputs represent the (deterministic) output performance measures obtained by transforming the input via the simulation's logic into the output. To abuse the computer-science anti-maxim of GIGO (Garbage In, Garbage Out), this situation might be called DIDO (Deterministic In, Deterministic Out). You might still have to make a lot of different runs, but for the purpose of evaluat-

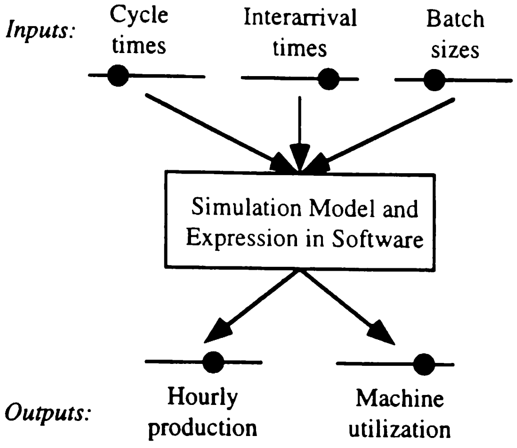

Figure 1: DIDO Simulation

ing a lot of different input-parameter combinations rather than to deal with uncertainty in the output.

But many (maybe most) systems involve some kind of uncertain, random input, so realistic simulation models ought to provide for such variable input as well; these are called *stochastic* simulation models. In fact, ignoring randomness in the input can make for dangerous errors in the simulation output. For instance, even in simple queueing models, which form the basic building blocks for a lot of simulations, the averages (expected values) of output performance measures like queue length and waiting time depend directly on the *variance* (as well as other things) of the service-time distributions. So ignoring randomness actually gets you the wrong answer, rather than just complicates your life. Besides, you might be interested in the output's randomness itself—like variability in hourly production. Of course, if you put random stuff into the simulation logic it's going to give you random stuff out—RIRO. Figure 2 illustrates the idea in the same manufacturing example, except now the inputs are probability distributions for the three quantities, regarded as random variables. The simulation proceeds by "drawing" realizations from the input probability distributions (indicated by the multiplicity of big dots from the input distributions) and transforms them into *an* observation on each of the (unknown) output distributions (indicated by the *single* big dot from each of the output distributions).

The whole point of this tutorial can be pretty much summed up by the fact that there's only a single big dot from the output distributions in Figure 2. The purpose of such a simulation is to learn (infer) something about these unknown output distributions, like maybe their expected values, variances, or probabilities on one side of some fixed tolerances. But all you get from one run of a stochastic simulation is a single observation on each of the output distributions, from which you obviously can't tell much about the governing output distribution (especially if, by unluck of the draw, you got an unusual value of the output under its distribution).

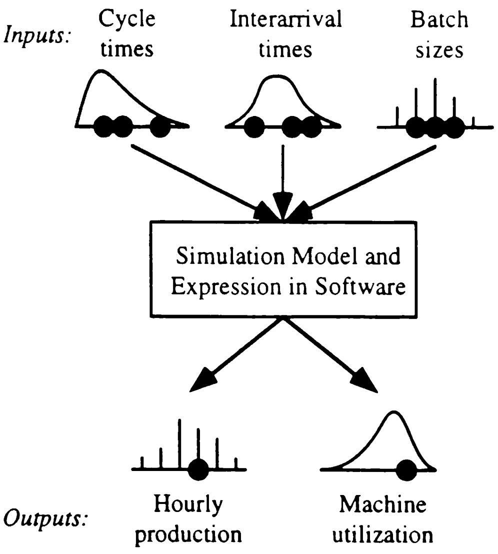

Figure 2: RIRO Simulation

So you have to think of *a* simulation run's output on some performance measure as *a* single observation on the output distribution; something very different could just as well have happened, just as something very different could just as well have happened on some other day in the actual manufacturing facility. Thus, you have to take care to perform the right kinds of simulation runs (design the simulation experiments) and do the right kinds of statistical analyses on the output data generated from the simulation. The rest of this tutorial will indicate some of the issues involved in these statistical questions as they apply to output data from a stochastic simulation.

3 EXPERIMENTAL DESIGN, OR THE SIMULATION AS TOMATO

Regardless of the type of simulation you have, you need to think ahead of time about exactly what scenarios you'll be asking your model to evaluate. Sometimes this is easy, having been specified by executive fiat or being just plain obvious. But most of the time it's not so clear, and you'll find yourself wondering what values the input parameters should take, and in what combinations with each other. You might also want to know what the effect is of changing some input parameter, and perhaps whether the effect of such a change might depend on (interact with) where the other parameters are set.

In these situations, formal experimental-design procedures can be of great help; you just need to think differently about them in the context of a simulation model. Traditionally, the "experiment" is some kind of physical situation; an agricultural experiment might be aimed at evaluating the effect of factors like different seed hybrids, fertilizers, and watering schedules on the yield of tomatoes, so a tomato-growing experiment with different factor-level combinations would be carried out and the results analyzed. The only things different about a simulation experiment are that you have a computer program rather than a tomato plant, the responses are the output measures like hourly production and machine utilization rather than yield of tomatoes, and the input factors are parameters like mean cycle times, variance of interarrival times, and maximum batch sizes, rather than seed hybrid, fertilizer, and watering (I guess another difference is that sometimes simulations run faster than tomatoes grow). So you can design the simulation experiment in the same way, and analyze it similarly as well in terms of measuring effects of factors and interactions among them.

Big simulations usually involve a lot of input factors, and you'll have to do some paring down of their numbers before you can do a workable analysis. For this purpose there are several factor-screening designs to help separate which factors matter (and should thus be retained as factors) and which ones don't (which should be frozen at some reasonable values and eliminated as factors).

For more on experimental design in the simulation context, see chapter 12 of Banks and Carson (1984), Cook (1992), Hood and Welch (1992), Chapter 12 of Law and Kelton (1991), Ramberg *et al.* (1991), Swain and Farrington (1994), and Sanchez (1994).

4 DOES TIME GO BY?

An issue that has as great an impact on what you do with your model as the deterministic/stochastic issue is whether time plays a role in the system and your model of it. Some simulations don't involve the passage of time, and are called *static*; examples include Monte Carlo evaluation of integrals and prediction with a cross-sectional regression model. The design-and-analysis approach is conceptually simple here (although may still be computationally scary): just repeat, or *replicate*, the model as many times as necessary to get the precision you need. Methods from classical statistical analysis can usually be used directly. For instance, in estimating an integral via a static Monte Carlo simulation, just get many independent estimates and then take the average, standard deviation, and maybe form a confidence interval in the elementary way.

But most simulations of industrial interest involve the passage of time as an important element; these are *dynamic* simulations, and the design-and-analysis approach can be a lot harder (as discussed in Section 5). From here on I'll assume that a dynamic simulation is what you've got.

5 EVALUATING A SINGLE CONFIGURATION

As a first step, you might want to evaluate the output from just a single configuration of the model. This section discusses issues involved in doing so, which will then be components of more ambitious goals like comparing alternative configurations or optimizing.

5.1 What to Watch?

In a stochastic simulation you'd really like to know all about the output distributions in Figure 2, but that's asking way too much in terms of the number and maybe length of the replications. So you usually have to settle for various summary measures of the output distributions. Traditionally, people have focused on estimating the expected value (mean) of the output distribution, and this can be of great interest. For instance, knowing something about the *average* hourly production is obviously important.

But things other than means might be interesting as well, like the *standard deviation* of hourly production, or the *probability* that the machine utilization for the period of the simulation will be above 0.80. In another example you might observe the *maximum* length of the queue of parts in a buffer somewhere to plan the floor space; in this connection it might be more reasonable to seek a value (called a *quantile*) below which the maximum queue length will fall with probability 0.95.

So think beforehand about what you'd like to get out of your simulation; it's easier to ignore things you have than go back and get things you forgot.

5.2 Multivariate Output

You'll probably want to get several different things out of your simulation; the stylized simulation of Figure 2 indicates two outputs, but dozens or scores might be more like it. Since these are all coming out of the same simulation runs, they're likely to be related in some way. For instance, high hourly production is probably associated with high utilization of the machine. So what you really have is a *vector* of output measures, and so multivariate statistical analyses can sometimes help you with estimating all the output parameters simultaneously, as well as with understanding how they might be related to each other.

For details in the simulation context and further references see Charnes (1991), section 9.7 of Law and Kelton (1991), and Seila (1991, 1992).

5.3 How Long?

A fundamental issue in your planning is whether you want performance measures over the long run (technically infinite, sometimes called *steady state*) or for a specific (finite, sometimes called *terminating*) fixed time period. The answer to this question is not a simulation issue, but rather one concerning the goals of your study. The answer also has obvious impact on how long you run your simulations; it also, perhaps less obviously, affects the kind of statistical analyses you can do on your output.

If a terminating simulation is appropriate for your goals, things are easy (at least in concept). Just run your model for whatever time period is called for, and get your output measures. That's one (replication, that is). Then repeat (replicate) this until you're happy with your results (described in more detail in Section 5.4 below). A complete run of the simulation constitutes a sample of size *one* (so isn't worth much), but standard statistical methods can be applied to the results *across* independent replications. The important thing is that you *do* multiple replications, *not* just one run, and use the output values from the entire replications as the fundamental "data points" for the basic ingredients in statistical analysis.

On the other hand, if you *really* want steady-state measures, the statistical-analysis problems become a lot harder (and, of course, your simulation runs become a lot longer). There are some things you can do, though, which are described in Section 5.5 below.

5.4 How to Express Things?

Traditionally, people have expressed statistical analyses of simulation output data in the form of confidence intervals (or confidence regions in the case of multivariate output). Compared to hypothesis tests, many people feel that confidence intervals are more informative and useful.

Increasingly, though, clever graphical displays are being used, which may not even involve formal inferential statistical analysis. For instance, histograms or dot plots of the output can indicate clear patterns that might not otherwise emerge from numerical measures. For more on graphical tools for describing simulation output, see Grier (1992).

Of course, animation has become very popular and, in some important ways, effective. But it's essential not to let yourself get swept along in the obvious visual appeal of animation to the exclusion of a proper statistical evaluation. For the fifteen simulated minutes that you had the patience to watch the animation, how do you know that the model was not in some weird state that's not representative of conditions as a whole?

5.5 Difficulties and Cautions

Alas, there are some pretty bad things you can do to yourself if you're not pretty careful about how your statistical analysis goes. Maybe the biggest mistake is to take as the basic "data" points for statistical analysis the *individual* observations coming out of a simulation over time. For instance, you dare not use the sequence of times in queue of successive parts in their raw form in standard statistical calculations (like "sample" variances). The problem is that they're not independent—if one part has a big delay in queue, the next one probably will too—which renders most of classical statistical theory invalid, sometimes with disastrous consequences. For instance, the "sample" variance of the individual part delays in queue will be biased low, perhaps severely, causing you to underestimate the variance and place more confidence in your results' precision than you ought to.

On the bright side, though, people have worked out some fairly simple and practical methods for dealing with simulation-generated data that usually work out pretty well. If you have a terminating simulation, for instance, you just make multiple replications and treat the summary statistics from each replication (averages, proportions, extremes, etc.) as the basic "data" points, which *can* be plugged into standard statistical formulas since the replications *are* identically distributed and independent of each other.

With steady-state simulations, though, things aren't quite so easy. Here are some ideas that people have come up with and tested out:

Replication. Even though you want (simulated) time to go to infinity, you can't. So just make runs as long as you can and then replicate them, pretending that the results give you a picture of being "in" steady state. You then have independent replications, just like in the terminating case, that you can plug into classical statistics. The problem with this is that the initial conditions you use to start the simulation (like everything's empty) are probably pretty atypical of steady state, which biases the run's output, at least for a while. You can make some plots and try to see where things stabilize, deleting the output data prior to that point, or maybe try to find better initial conditions that are more representative of steady state.

Batch means. Since you want to get as close as you can to steady state, just make one enormously long run. But then you really have only one replication (data point), so you can't estimate variances or do statistical analysis. To "manufacture" more observations out of this, split the run up into "batches" of observations, and treat the means of these batches as being independent unbiased observations of what's going on in steady state. While the initial-condition bias is less severe than with the Replication method, the batch means are not really independent; the key is to have big batches, and people have developed ways to help you decide how big the batches need to be for your situation.

Time-series models. The correlated, nonstationary simulation output series can be thought of as a time series, just like economic data such as stock prices or housing starts over time. Then a time-series model (like AR or ARMA) is fit to the data, and the fitted model is used for inference.

Standardized time series. A process version of the central limit theorem is applied to "standardize" the output series, and methods for statistical analysis have been worked out based on this.

Regeneration cycles. Some simulations return now and then to a state from which they "start over" probabilistically. For instance, if a queue empties out at some point it looks just like it did at the beginning (assuming it started empty). This creates independent *cycles* that are manipulated for statistical analysis.

Spectral analysis. Estimates of the correlation structure of the process are used to form a variance estimate for statistical analysis.

You get the idea that this is a hard problem, and that there is no completely satisfactory solution.

There's a very large literature on this subject, and the above list is a pretty thin tour of these methods. But they're all explained in detail elsewhere; see Alexopoulos (1993, 1994), chapter 11 of Banks and Carson (1984), chapter 3 of Bratley, Fox, and Schrage (1987), Charnes (1993), chapters 2, 3, and 5 of Fishman (1978), Goldsman (1992), chapter 7 of Khoshnevis (1994), Kleijnen (1987), chapter 9 of Law and Kelton (1991), Lewis and Orav (1989), chapter 6 of Ripley (1987), Seila (1991, 1992), and chapter 6 of Thesen and Travis (1992).

6 COMPARING ALTERNATIVES

Most of the time you'll be considering several different configurations of a simulation model, perhaps distinguished from each other by input-parameter values or by logical and structural differences. On the basis of some output performance measure, you might like to estimate the difference between various pairings of the configurations, perhaps expressed as a confidence interval for a difference or maybe a test of the null hypothesis that there is no difference. Most of the methods described in Section 5.5 can be adapted for these kinds of goals. For instance, in a terminating simulation you can use paired-sample confidence intervals for the difference, discussed in any elementary statistics book. The same difficulties and cautions apply, though, if you're interested in steady state.

Simulation is an ideal setting in which to apply any of several *selection and ranking* techniques. For instance, you can invoke statistical methods (basically telling you how much data you need to collect) that allow you to declare one of your alternatives as being the best on some criterion, and be highly confident that you're right about your choice. What makes simulation an attractive setting for this is that these techniques often require two-stage or sequential sampling (deciding on the sample size on the fly), which is a lot easier to do in simulation than in growing tomatoes.

For more depth on these subjects, see chapter 12 of Banks and Carson (1984), Goldsman, Nelson, and Schmeiser (1991), chapter 10 of Law and Kelton (1991), Nelson and Goldsman (1994), or chapter 7 of Thesen and Travis (1992).

7 VARIANCE REDUCTION

Section 5.5 dwelt on some of the difficulties and dangers in dealing with simulation data, but on the positive side there are some important opportunities not available when experimenting with tomatoes. Ease of sequential sampling, as mentioned in Section 6, was one example.

But a more important example is to reduce the variance of the output without doing any (well, hardly any) extra work. Such *variance-reduction* techniques often proceed by exploiting your ability to control the random-number generator driving the simulation, and re-use random numbers to induce helpful correlations that reduce noise in the output.

For instance, when comparing several alternative configurations of a manufacturing facility you could use the *same* random numbers, properly synchronized, to drive all configurations. This would result in the same jobs' arriving to the facilities at the same times, and with the same processing requirements. Whatever differences in performance you observe are thus attributable to configuration differences rather than to "environmental" differences in the arriving jobs (since there weren't any such differences). While intuitively appealing, there is actually firm statistical foundation for this, and the variance of the difference is usually reduced. This strategy, known as *common random numbers*, is often used unconsciously by just starting the runs for all alternatives with the same random-number streams and seeds.

There are many other sophisticated variance-reduction ideas; for details see chapter 2 of Bratley, Fox, and Schrage (1987), chapter 3 of Fishman (1978), chapter 11 of Law and Kelton (1991), Kleijnen (1987), L'Ecuyer (1994), Lewis and Orav (1989), chapter 7 of Morgan (1984), Nelson (1992), and chapter 5 of Ripley (1987).

8 WHAT IF YOU WANT SENSITIVITIES?

Related to the question of comparing alternatives is the more micro-level question of measuring the effect on the output due to a change in one or several of the inputs. For example, how much would hourly production increase if the mean cycle time on a critical machine were reduced by a small amount? Viewing the output as a (complicated) function of the input, this is a question about a partial derivative of the output with respect to one of the inputs.

A direct way to address this is to make two sets of runs—one at the original value and another at the changed value of the input parameter—and then look at the difference. There are other ways of doing this, though, that are more clever (and maybe more complicated), yet are also more economical from the point of view of the amount of simulating you have to do.

Details on these methods can be found in chapter 12 of Law and Kelton (1991), Glasserman and Glynn (1992), Ho (1992), L'Ecuyer (1991), and Strickland (1993).

9 METAMODELS

Thinking of the simulation logic and action as being a transformation of inputs into outputs, the notion arises that a simulation is just a function, albeit a pretty complicated one that you can't write down as some little formula. But it might be possible to *approximate* what the simulation does with some little formula, which could be particularly useful if a large number of input-factor combinations are of interest and it takes a long time to run the simulation.

So people sometimes fit a regression model to the simulation model, with the dependent variable's being the simulation output and the independent variables' being the input parameters to the simulation. All the usual techniques for building regression models come into play, like selecting important subsets of the independent variables, modeling nonlinearities, and considering interactions. Since this is a (regression) model of a (simulation) model, it's sometimes called a *metamodel.*

For more on metamodeling, see Barton (1992), Hood and Welch (1993), Kleijnen (1987), and chapter 12 of Law and Kelton (1991).

10 FINDING OPTIMAL CONFIGURATIONS

The ultimate, maybe, in using a simulation model is to find input-factor settings that optimize some performance measure. This could involve several of the above issues, including gradient estimation, metamodeling, and comparing alternatives. Now optimization of nonlinear functions is a hard enough problem in itself, but in a stochastic simulation you have uncertainty in terms of measuring the response, as well as the statistical difficulties described in Section 5.5. So this is truly a tall order.

People have made important advances in this, though. One idea is to estimate the partial derivatives at a point (the gradient), then move in the direction of steepest descent (if you're minimizing) or steepest ascent (if you're maximizing). You could also fit a regression metamodel as in Section 9 and then use simple calculus to optimize it in lieu of the simulation itself. There are, to be sure, many more techniques (like adaptation of stochastic-programming methods) that have been developed or are under investigation; for more details see Azadivar (1992), Fu (1994), or chapter 12 of Law and Kelton (1991).

11 CONCLUSIONS

While all the details, methods, and cautions of doing a good job at output analysis may seem bewildering, you really owe it to yourself to try to get as much honest, precise information out of your hard-built simulation model as you can. While there are dangers and difficulties at times, there are also reliable and robust methods available. Moreover, some simulation-software products now have output-analysis capabilities built in to facilitate things.

ACKNOWLEDGMENT

I'm grateful for support from the Minnesota Supercomputer Institute.

REFERENCES

Alexopoulos, C. 1993. Advanced simulation output analysis for a single system. In *Proceedings of the 1993 Winter Simulation Conference*, ed. G.W. Evans, M. Mollaghasemi, E.C. Russell, and W.E. Biles, 89–96. WSC Board of Directors.

Alexopoulos, C. 1994. A review of advanced methods for simulation output analysis. In *Proceedings of the 1994 Winter Simulation Conference*, ed. J.D. Tew, M.S. Manivannan, D.A. Sadowski, and A.F. Seila, 133–140. WSC Board of Directors.

Azadivar, F. 1992. A tutorial on simulation optimization. In *Proceedings of the 1992 Winter Simulation Conference*, ed. J.J. Swain, D. Goldsman, R.C. Crain, and J.R. Wilson, 198–204. WSC Board of Directors.

Banks, J. and J.S. Carson, II. 1984. *Discrete-event system simulation.* Englewood Cliffs, New Jersey: Prentice-Hall.

Barton, R.R. 1992. Metamodels for Simulation Input-Output Relations. In *Proceedings of the 1992 Winter Simulation Conference*, ed. J.J. Swain, D. Goldsman, R.C. Crain, and J.R. Wilson, 289–299. WSC Board of Directors.

Bratley, P., B.L. Fox, and L.E. Schrage. 1987. *A guide to simulation.* 2nd ed. New York: Springer-Verlag.

Charnes, J.M. 1991. Multivariate simulation output analysis. In *Proceedings of the 1991 Winter Simulation Con-*

ference, ed. B.L. Nelson, W.D. Kelton, and G.M. Clark, 187–193. WSC Board of Directors.

Charnes, J.M. 1993. Statistical analysis of output processes. In *Proceedings of the 1993 Winter Simulation Conference*, ed. G.W. Evans, M. Mollaghasemi, E.C. Russell, and W.E. Biles, 41–49. WSC Board of Directors.

Cook, L.S. 1992. Factor screening of multiple responses. In *Proceedings of the 1992 Winter Simulation Conference*, ed. J.J. Swain, D. Goldsman, R.C. Crain, and J.R. Wilson, 174–180. WSC Board of Directors.

Fishman, G.S. 1978. *Principles of discrete event simulation*. New York: John Wiley & Sons.

Fu, M.C. 1994. A tutorial review of techniques for simulation optimization. In *Proceedings of the 1994 Winter Simulation Conference*, ed. J.D. Tew, M.S. Manivannan, D.A. Sadowski, and A.F. Seila, 149–156. WSC Board of Directors.

Glasserman, P. and P.W. Glynn. 1992. Gradient estimation for regenerative processes. In *Proceedings of the 1992 Winter Simulation Conference*, ed. J.J. Swain, D. Goldsman, R.C. Crain, and J.R. Wilson, 280–288. WSC Board of Directors.

Goldsman, D. 1992. Simulation output analysis. In *Proceedings of the 1992 Winter Simulation Conference*, ed. J.J. Swain, D. Goldsman, R.C. Crain, and J.R. Wilson, 97–103. WSC Board of Directors.

Goldsman, D. and B.L. Nelson. 1994. Ranking, selection and multiple comparisons in computer simulation. In *Proceedings of the 1994 Winter Simulation Conference*, ed. J.D. Tew, M.S. Manivannan, D.A. Sadowski, and A.F. Seila, 192–199. WSC Board of Directors.

Goldsman, D., B.L Nelson, and B. Schmeiser. 1991. Methods for selecting the best system. In *Proceedings of the 1991 Winter Simulation Conference*, ed. B.L. Nelson, W.D. Kelton, and G.M. Clark, 177–186. WSC Board of Directors.

Grier, D.A. 1992. Graphical techniques for output analysis. In *Proceedings of the 1992 Winter Simulation Conference*, ed. J.J. Swain, D. Goldsman, R.C. Crain, and J.R. Wilson, 314–319. WSC Board of Directors.

Ho, Y.-C. 1992. Perturbation analysis: concepts and algorithms. In *Proceedings of the 1992 Winter Simulation Conference*, ed. J.J. Swain, D. Goldsman, R.C. Crain, and J.R. Wilson, 231–240. WSC Board of Directors.

Hood, S.J. and P.D. Welch. 1992. Experimental design issues in simulation with examples from semiconductor manufacturing. In *Proceedings of the 1992 Winter Simulation Conference*, ed. J.J. Swain, D. Goldsman, R.C. Crain, and J.R. Wilson, 255–263. WSC Board of Directors.

Hood, S.J. and P.D. Welch. 1993. Response surface methodology and its application in simulation. In *Proceedings of the 1993 Winter Simulation Conference*, ed. G.W. Evans, M. Mollaghasemi, E.C. Russell, and W.E. Biles, 115–122. WSC Board of Directors.

Kelton, W.D. 1994. Analysis of output data. In *Proceedings of the 1994 Winter Simulation Conference*, ed. J.D. Tew, M.S. Manivannan, D.A. Sadowski, and A.F. Seila, 62–68. WSC Board of Directors.

Khoshnevis, B. 1994. *Discrete systems simulation*. New York: McGraw-Hill.

Kleijnen, J.P.C. 1987. *Statistical tools for simulation practitioners*. New York: Marcel Dekker, Inc.

Law, A.M. and W.D. Kelton. 1991. *Simulation modeling and analysis*. 2nd ed. New York: McGraw-Hill.

L'Ecuyer, P. 1991. An overview of derivative estimation. In *Proceedings of the 1991 Winter Simulation Conference*, ed. B.L. Nelson, W.D. Kelton, and G.M. Clark, 28–36. WSC Board of Directors.

L'Ecuyer, P. 1994. Efficiency improvement and variance reduction. In *Proceedings of the 1994 Winter Simulation Conference*, ed. J.D. Tew, M.S. Manivannan, D.A. Sadowski, and A.F. Seila, 122–132. WSC Board of Directors.

Lewis, P.A.W. and E.J. Orav. 1989. *Simulation methodology for statisticians, operations analysts, and engineers, volume I*. Belmont, California: Wadsworth, Inc.

Morgan, B.J.T. 1984. *Elements of simulation*. London: Chapman and Hall.

Nelson, B.L. 1992. Designing efficient simulation experiments. In *Proceedings of the 1992 Winter Simulation Conference*, ed. J.J. Swain, D. Goldsman, R.C. Crain, and J.R. Wilson, 126–132. WSC Board of Directors.

Ramberg, J.S., S.M. Sanchez, P.J. Sanchez, and L.J. Hollick. 1991. Designing simulation experiments: Taguchi methods and response surface metamodels. In *Proceedings of the 1991 Winter Simulation Conference*, ed. B.L. Nelson, W.D. Kelton, and G.M. Clark, 167–176. WSC Board of Directors.

Ripley, B.D. 1987. *Stochastic simulation*. New York: John Wiley & Sons.

Seila, A.F. 1991. Output analysis for simulation. In *Proceedings of the 1991 Winter Simulation Conference*, ed. B.L. Nelson, W.D. Kelton, and G.M. Clark, 28–36. WSC Board of Directors.

Seila, A.F. 1992. Advanced output analysis for simulation. In *Proceedings of the 1992 Winter Simulation Conference*, ed. J.J. Swain, D. Goldsman, R.C. Crain, and J.R. Wilson, 190–197. WSC Board of Directors.

Strickland, S.G. 1993. Gradient/sensitivity estimation in discrete-event simulation. In *Proceedings of the 1993 Winter Simulation Conference*, ed. G.W. Evans, M. Mollaghasemi, E.C. Russell, and W.E. Biles, 97–105. WSC Board of Directors.

Swain, J.J. and P.A. Farrington. 1994. Designing simulation experiments for evaluating manufacturing systems. In *Proceedings of the 1994 Winter Simulation Confer-*

ence, ed. J.D. Tew, M.S. Manivannan, D.A. Sadowski, and A.F. Seila, 69–76. WSC Board of Directors.

Sanchez, S.M. 1994. A robust design tutorial. In *Proceedings of the 1994 Winter Simulation Conference*, ed. J.D. Tew, M.S. Manivannan, D.A. Sadowski, and A.F. Seila, 106–113. WSC Board of Directors.

Thesen, A. and L.E. Travis. 1992. *Simulation for decision making*. St. Paul, Minnesota: West Publishing Company.

AUTHOR BIOGRAPHY

W. DAVID KELTON is Professor of Operations and Management Science in the Carlson School of Management at the University of Minnesota, as well as a Fellow of the Minnesota Supercomputer Institute and a member of the Graduate Faculty in the Scientific Computation Program at Minnesota. He received a B.S. 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.