Proceedings of the 2003 Winter Simulation Conference
S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, eds.

EXPERIMENTAL DESIGN FOR SIMULATION

W. David Kelton

Department of Quantitative Analysis
and Operations Management
University of Cincinnati
Cincinnati, OH 45221-0130, U.S.A.

Russell R. Barton

The Smeal College of Business Administration
The Pennsylvania State University
University Park, PA 16802, U.S.A.

ABSTRACT

This tutorial introduces some of the ideas, issues, challenges, solutions, and opportunities in deciding how to experiment with simulation models to learn about their behavior. Careful planning, or designing, of simulation experiments is generally a great help, saving time and effort by providing efficient ways to estimate the effects of changes in the model's inputs on its outputs. Traditional experimental-design methods are discussed in the context of simulation experiments, as are the broader questions pertaining to planning computer-simulation experiments.

1 INTRODUCTION

The real meat of a simulation project is running your models and trying to understand the results. To do so effectively, you need to plan ahead before doing the runs, since just trying different things to see what happens can be a very inefficient way of attempting to learn about your models' (and hopefully the systems') behaviors. Careful planning of how you're going to experiment with your models will generally repay big dividends in terms of how effectively you learn about the systems and how you can exercise your models further.

This tutorial looks at such *experimental-design* issues in the broad context of a simulation project. The term "experimental design" has specific connotations in its traditional interpretation, and I will mention some of these below, in Section 5. But I will also try to cover the issues of planning your simulations in a broader context, which consider the special challenges and opportunities you have when conducting a computer-based simulation experiment rather than a physical experiment. This includes questions of the overall purpose of the project, what the output performance measures should be, how you use the underlying random numbers, measuring how changes in the inputs might affect the outputs, and searching for some kind of optimal system configuration. Specific questions of this type might include:

- What model configurations should you run?
- How long should the runs be?
- How many runs should you make?
- How should you interpret and analyze the output?
- What's the most efficient way to make the runs?

These questions, among others, are what you deal with when trying to design simulation experiments.

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 refer you instead to any of several texts on simulation that do, and to tutorials and reviews on this subject in this and recent *Proceedings* of the Winter Simulation Conference. General book-based references for this subject include chapter 12 of Law and Kelton (2000), chapter 11 of Kelton, Sadowski, and Sadowski (2002), Banks, Carson, Nelson, and Nicol (2001), Kleijnen (1998), and Barton (1999), all of which contain numerous references to other books and papers on this subject. Examples of application of some of these ideas can be found in Hood and Welch (1992, 1993) and Swain and Farrington (1994), and another recent tutorial is Barton (2002). Parts of this paper are taken from Kelton (1997, 2000), which also contain further references and discussion on this and closely related subjects.

2 WHAT IS THE PURPOSE OF THE PROJECT?

Though it seems like pretty obvious advice, it might bear mentioning that you should be clear about what the ultimate purpose is of doing your simulation project in the first place. Depending on how this question is answered, you can be led to different ways of planning your experiments. Worse, failure to ask (and answer) the question of just what the point of your project is can often leave you adrift without any organized way of carrying out your experiments.

For instance, even if there is just one system of interest to analyze and understand, there still could be questions like run length, the number of runs, allocation of random numbers, and interpretation of results, but there are no questions of which model configurations to run. Likewise, if there are just a few model configurations of interest, and they have been given to you (or are obvious), then the

problem of experimental-design is similar to the single-configuration situation.

However, if you are interested more generally in how changes in the inputs affect the outputs, then there clearly are questions of which configurations to run, as well as the questions mentioned in the previous paragraph. Likewise, if you're searching for a configuration of inputs that maximizes or minimizes some key output performance measure, you need to decide very carefully which configurations you'll run (and which ones you won't).

The reality is that often you can't be completely sure what your ultimate goals are until you get into a bit. Often, your goals may change as you go along, generally becoming more ambitious as you work with your models and learn about their behavior. The good news is that as your goals become more ambitious, what you learned from your previous experiments can help you decide how to proceed with your future experiments.

3 WHAT ARE THE RELEVANT OUTPUT-PERFORMANCE MEASURES?

Most simulation software produces a lot of numerical output by default, and you can usually specify additional output that might not be automatically delivered. Much of this output measures traditional time-based quantities like time durations or counts of entities in various locations. Increasingly, though, economic-based measures like cost or value added are being made available, and are of wide interest. Planning ahead to make sure you get the output measures you need is obviously important if the runs are time-consuming to carry out.

One fundamental question relates to the time frame of your simulation runs. Sometimes there is a natural or obvious way to start the simulation, and an equally natural or obvious way to terminate it. For instance, a call center might be open from 8a.m. to 8p.m. but continue to operate as necessary after 8 p.m. to serve all calls on hold (in queue) at 8 p.m. In such a case, often called a terminating simulation, there is no design question about starting or stopping your simulation — these are part and parcel of the model specification itself. (By the way, you should take care to get this part of the modeling just as right as the more obvious aspects like logic and input-parameter values, since the manner in which a simulation is started and stopped can sometimes have important impact on the results.)

On the other hand, interest may be in the *long-run* (also called *infinite-horizon*) behavior of the system, in which case it is no longer clear how to start or stop the simulation (though it seems clear that the run length will have to be comparatively long). Continuing the call-center example, perhaps its hours are going to expand to 24 hours a day, seven days a week; in this case you would need a *steady-state* simulation to estimate the relevant performance measures.

Regardless of the time frame of the simulation, you have to decide what aspects of the model's outputs you want. In a stochastic simulation you'd really like to know all about the output probability distributions, but that's asking way too much in terms of the number and maybe length of the runs. So you usually have to settle for various summary measures of the output distributions. Traditionally, people have focused on estimating the *expected value* (or *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, say, 0.95.

Even if you want just simple averages, the specifics can affect how your model is built. For instance, if you want just the time-average number of parts in a queue, you would need to track the length of this queue but not the times of entry of parts into the queue. However, if you want the average time parts spend in the queue, you do need to note their time of entry in order to compute their time in queue.

So think beforehand about precisely 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. (On the other hand, asking for everything out of your run, including the kitchen sink, can have unhappy effects on computation time.)

4 HOW SHOULD YOU USE AND ALLOCATE THE UNDERLYING RANDOM NUMBERS?

Most simulations are *stochastic*, i.e., involve random inputs from probability distributions to represent things like service times, interarrival times, and pass/fail decisions. Simulation software has facilities to generate observations from such distributions, which rely at root on a *random-number generator* churning out a sequence of values between 0 and 1 that are supposed to behave as though they are independent and uniformly distributed on the interval [0, 1]. Such generators are in fact fixed, recursive formulas that always give you the same sequence of "random" numbers in the same order (provided that you don't override the default seeds for these generators). The challenge in developing such generators is that they behave as intended, in a statistical sense, and that they have a long cycle length before they double back on themselves and repeat the same sequence over again.

Obviously, it is important that a "good" random-number generator be used. And, from the experimental-design viewpoint, you can then dispense with the issue of randomizing experimental treatments to cases, which is often a thorny problem in physical experiments.

But with such controllable random-number generators, the possibility arises in computer-simulation experiments to control the basic randomness, which is a fundamentally different situation from what you encounter in physical experiments. Doing so carefully is one way of implementing what are known as *variance-reduction techniques*, which can often sharpen the precision of your output estimators without having to do more simulating. The basic question in doing so is planning how you are going to allocate the underlying random numbers to generating the various random inputs to your models.

Perhaps the first thought along these lines that seems like a "good" statistical idea is to ensure that all the random-number usage is independent within your models as well as between any alternative configurations you might run. This is certainly a statistically valid way to proceed, and is statistically the simplest approach. However, it might not be the most efficient approach, where "efficiency" could be interpreted in either its statistical sense (i.e., low variance) or in its computational sense (i.e., amount of computational effort to produce results of adequate precision). And at a more practical level, it might actually take specific action on your part to accomplish independence between alternative configurations since most simulation software is set up to start a new run (e.g., for the next model) with the same random numbers as before.

But actually, that feature of simulation software can be to your advantage, provided that you plan carefully for exactly how the random numbers will be re-used. By using the same random numbers *for the same purposes* between different alternative configurations you are running them under the same or similar external conditions, such as exactly what values the service and interarrival times take on. In this way, any differences you see in performance can be attributed to differences in the model structures or parameter settings rather than to differences in what random numbers you happened to get. This idea is usually called *common random numbers*, and can sometimes greatly reduce the variance in your estimators of the difference in performance between alternative configurations. To implement it properly, though, you need to take deliberate steps to make sure that your use of the common random numbers is *synchronized* between the systems, or else the variance-reducing effect will be diluted or maybe even largely lost. Often, using fixed streams of the random-number generator, which are really just particular subsequences, can facilitate maintaining proper synchronization.

There are several other variance-reduction techniques that also rely on (carefully) re-using previously used random numbers, such as *antithetic variates*. Most of these techniques also rely on some kind of careful planning for synchronization of their use.

5 HOW SENSITIVE ARE YOUR OUTPUTS TO CHANGES IN YOUR INPUTS?

As part of building a simulation model, you have to specify a variety of *input factors*. These include quantitative factors like the mean interarrival time, the number of servers, and the probabilities of different job types. Other input factors are more logical or structural in nature, like whether failure/feedback loops are present, and whether a queue is processed first-in-first-out or shortest-job-first. There can also be factors that are somewhere between being purely quantitative and purely logical/structural, like whether the service-time probability distribution is exponential or uniform.

Another classification dimension of input factors is whether they are (in reality) controllable or not. However, when exercising a simulation model, all input factors are controllable, whether or not they can in reality be set or changed at will. For instance, you can't just cause the arrival rate to a call center to double, but you'd have no problem doing so in your simulation model of that call center.

In any case, exactly how you specify each input factor will presumably have some effect on the output performance measures. Accordingly, it is sometimes helpful to think of the simulation as a function that transforms inputs into outputs:

$$\begin{aligned} \text{Output}_1 &= f_1(\text{Input}_1, \text{Input}_2, \ldots) \\ \text{Output}_2 &= f_2(\text{Input}_1, \text{Input}_2, \ldots) \\ &\vdots \end{aligned}$$

where the functions $f_1, f_2, \ldots$ represent the simulation model itself.

It is often of interest to estimate how a change in an input factor affects an output performance measure, i.e., how sensitive an output is to a change in an input. If you knew the form of the simulation functions $f_1, f_2, \ldots$, this would essentially be a question of finding the partial derivative of the output of interest with respect to the input of interest.

But of course you *don't* know the form of the simulation functions — otherwise you wouldn't be simulating in the first place. Accordingly, there are several different strategies for estimating the sensitivities of outputs to changes in inputs. These strategies have their own advantages, disadvantages, realms of appropriate application, and extra information they might provide you. In the remainder of this section I'll mention some of these, describe them in general terms, and give references for further details.

5.1 Classical Experimental Design

A wide variety of approaches, methods, and analysis techniques, known collectively as *experimental design*, has

been around for many decades and is well documented in books like Box, Hunter, and Hunter (1978) or Montgomery (1997). One of the principal goals of experimental design is to estimate how changes in input factors affect the results, or *responses*, of the experiment.

While these methods were developed with physical experiments in mind (like agricultural or industrial applications), they can fairly easily be used in computer-simulation experiments as well, as described in more detail in chapter 12 of Law and Kelton (2000). In fact, using them in simulation presents several opportunities for improvement that are difficult or impossible to use in physical experiments.

As a basic example of such techniques, suppose that you can identify just two values, or *levels*, of each of your input factors. There is no general prescription on how to set these levels, but you should set them to be "opposite" in nature but not so extreme that they are unrealistic. If you have k input factors, there are thus 2^k different combinations of the input factors, each defining a different configuration of the model; this is called a 2^k *factorial design*. Referring to the two levels of each factor as the "–" and "+" level, you can form what is called a *design matrix* describing exactly what each of the 2^k different model configurations are in terms of their input factor levels. For instance, if there are $k = 3$ factors, you would have $2^3 = 8$ configurations, and the design matrix would be as in Table 1, with R_i denoting the simulation response from the ith configuration.

Table 1: Design Matrix for a 2^3 Factorial Experiment

Run (i)	Factor 1	Factor 2	Factor 3	Response
1	–	–	–	R_1
2	+	–	–	R_2
3	–	+	–	R_3
4	+	+	–	R_4
5	–	–	+	R_5
6	+	–	+	R_6
7	–	+	+	R_7
8	+	+	+	R_8

The results from such an experiment can be used in many ways. For instance, the *main effect* of Factor 2 in the above example is defined as the average difference in response when this factor moves from its "–" level to its "+" level; it can be computed by applying the signs in the Factor 2 column to the corresponding responses, adding, and then dividing by $2^{k-1} = 4$:

$$(-R_1 - R_2 + R_3 + R_4 - R_5 - R_6 + R_7 + R_8)/4.$$

The main effects of the other factors are computed similarly.

Further, you can ask whether the effect of one factor might depend in some way on the level of one or more other factors, which would be called *interaction* between the factors if it seems to be present. To compute the interactions from the experimental results, you "multiply" the columns of the involved factors row by row (like signs multiply to "+," unlike signs multiply to "–"), apply the resulting signs to the corresponding responses, add, and divide by $2^{k-1} = 4$. For instance, the interaction between Factors 1 and 3 would be

$$(+R_1 - R_2 + R_3 - R_4 - R_5 + R_6 - R_7 + R_8)/4.$$

If an interaction is present between two factors, then the main effect of those factors cannot be interpreted in isolation.

Which brings up the issue of limitations of these kinds of designs. There is a specific linear-regression model underlying designs like these, which have present an independent-variable term involving each factor on its own (linearly), and then possible cross-products between the factor levels, representing interactions. As suggested, significant interactions cloud the interpretation of main effects, since presence of the cross product causes the main effect no longer to be an accurate measure of the effect of moving this factor from its "–" level to its "+" level. One way around this limitation is to specify a more elaborate and more general underlying regression model, and allow for more than just two levels for each input factor. This gives rise to more complex designs, which must be set up and analyzed in more sophisticated ways; see the experimental-design references cited earlier.

Another difficulty with full-factorial designs is that if the number of factors becomes even moderately large, the number of runs explodes (it is, after all, literally exponential in the number of factors). A way around this is to use what are known as *fractional-factorial* designs in which only a fraction (sometimes just a small fraction) of all the possible factor-combinations are run. You must take care, however, to pick the subset of the runs very carefully, and there are specific prescriptions on how to do this in the references cited earlier. The downside of doing only a fraction of the runs is that you have to give up the ability to estimate at least some of the potential interactions, and the smaller the number of runs the fewer the number of interactions you can estimate.

A final limitation of these kinds of designs is that the responses are random variables, as are all outputs from stochastic simulations. Thus, your estimates of things like main effects and interactions are subject to possibly-considerable variance. Unlike physical experiments, though, you have the luxury in simulation of *replicating* (independently repeating) the runs many times to reduce this variance, or perhaps replicating the whole design many times to get many independent and identically distributed estimates of main effects and interactions, which could then be combined to form, say, a confidence interval on the expected main effects and interactions in the usual way. This is a good approach for determining whether a main effect or interaction is really present — if the confidence

interval for it does not contain zero, then it appears that it is really present.

There are certainly many other kinds of more sophisticated factorial designs than what I have described here; see the references cited earlier for examples

5.2 Which Inputs are Important? Which are Not?

As mentioned above, if the number of factors is even moderately large, the number of possible factor-level combinations simply explodes far beyond anything remotely practical. It is unlikely, though, that *all* of your input factors are really important in terms of having a major impact on the outputs. At the very least, there will probably be big differences among your factors in terms of their impact on your responses.

Since it is the number of factors that causes the explosion in the number of combinations, it would be most helpful to identify early in the course of experimentation which factors are important and which are not. The unimportant factors can then be fixed at some reasonable value and dropped from consideration, and further investigation can be done on the important factors, which will be fewer in number. There are several such *factor-screening* designs in the literature (see the references cited earlier), and they can be extremely helpful in transforming a rather hopelessly large number of runs into something that is eminently manageable.

5.3 Response-Surface Methods and Metamodels

Most experimental designs, including those mentioned above, are based on an algebraic regression-model assumption about the way the input factors affect the outputs. For instance, if there are two factors (X_1 and X_2, say) that are thought to affect an output response Y, you might approximate this relationship by the regression model

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1 X_2 + \beta_4 X_1^2 + \beta_5 X_2^2 + \varepsilon$$

where the β_j coefficients are unknown and must be estimated somehow, and ε is a random error term representing whatever inaccuracy such a model might have in approximating the actual simulation-model response Y. Since in this case the above regression model is an approximation to another model (your simulation model), the regression is a "model of a model" and so is sometimes called a *metamodel*. And since a plot of the above situation (with two independent input variables) would be a three-dimensional surface representing the simulation responses, this is also called a *response surface*.

The parameters of the model are estimated by making simulation runs at various input values for the X_j's, recording the corresponding responses, and then using standard least-squares regression to estimate the coefficients. Exactly which sets of input values are used to make the runs to generate the "data" for the regression fit is itself an experimental-design question, and there are numerous methods in the references cited above. A more comprehensive reference on this subject is Box and Draper (1987).

In simulation, an estimated response-surface metamodel can serve several different purposes. You could (literally) take partial derivatives of it to estimate the effect of small changes in the factors on the output response, and any interactions that might be present as modeled would show up naturally. You could also use the estimated metamodel as a proxy for the simulation, and very quickly explore many different input-factor-level combinations without having to run the simulation. And you could try to optimize (maximize or minimize, as appropriate) the fitted model to give you a sense of where the best input-factor-combinations might be.

An obvious caution on the use of response surfaces, though, is that they are estimated from simulation-generated data, and so are themselves subject to variation. This uncertainty can then have effects on your estimates of unsimulated models, derivatives, and optimizers. Barton (1998) and the references cited above discuss these issues, which are important in terms of understanding and interpreting your results and estimates realistically.

5.4 Other Techniques

The discussion above focuses on general approaches that originated in physical, non-simulation contexts, but nevertheless can be applied in simulation experiments as well. There are a variety of other methods that are more specific to simulation, including *frequency-domain methods* and *perturbation analysis*. For discussions of these ideas, see advanced or state-of-the-art tutorials in this or recent *Proceedings* of the Winter Simulation Conference.

6 WHAT IS THE "BEST" COMBINATION OF INPUTS?

Sometimes you have a single output performance measure that is of overriding importance in comparison with the other outputs (different outputs can conflict with each other, like the desirability of both high machine utilization and short queues). This might be a measure of direct economic importance, like profit or cost. If you have such a measure, you would probably like to look for an input-factor combination that optimizes this measure (e.g., maximizes profit or minimizes cost). Mathematically, this can take the form of some kind of search through the space of possible factor combinations. For a review of the underlying methods, see Andradóttir (1998); for a comprehensive survey on both the theory and practice, see Fu (2002) and the ensuing Commentaries and Rejoinder there.

This is a tall order, from any of several perspectives. If there are a lot of input factors, the dimension of the search space is high, requiring a lot of simulations at a lot of different points. And in stochastic simulation, the responses are subject to uncertainty, which must be taken into account when deciding how best to proceed with your search.

Fortunately, several heuristic search methods have been developed that "move" you from one point to a more promising one, and make these decisions based on a host of information that is available. And we are now beginning to see some of these methods coded into commercial-grade software and even integrated in with some simulation-software products. For example, see Glover, Kelly, and Laguna (1999).

CONCLUSIONS

My purpose here has been to make you aware of the issues in conducting simulation experiments that deserve your close attention. An unplanned, hit-or-miss course of experimentation with a simulation model can often be frustrating, inefficient, and ultimately unhelpful. On the other hand carefully planned simulation studies can yield valuable information without an undue amount of computational effort or (more importantly) your time. Indeed, I would go so far as to say that any simulation study without a design-of-experiments aspects has probably squandered the probably-considerable effort that went into the modeling, since it's just not that hard to do at least something to design and analyze an informative experiment; your computer might need to grind away for a while, but that's cheap compared to the time you put into the modeling, and compared to the significance of the decisions that will based on what's learned from the simulation experiments.

REFERENCES

Andradóttir, S. 1998. A review of simulation optimization techniques. In *Proceedings of the 1998 Winter Simulation Conference*, ed. D.J. Medeiros, E.F. Watson, J.S. Carson, and M.S. Manivannan, 151–158. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Banks, J., J.S. Carson, B.L. Nelson, and D.M. Nicol. 2001. *Discrete-event system simulation*. 3rd ed. Upper Saddle River, N.J.: Prentice-Hall.

Barton, R.R. 1998. Simulation metamodels. In *Proceedings of the 1998 Winter Simulation Conference*, ed. D.J. Medeiros, E.F. Watson, J.S. Carson, and M.S. Manivannan, 167–174. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Barton, R.R. 1999. *Graphical methods for the design of experiments*. New York: Springer-Verlag.

Barton, R.R. 2002. Designing simulation experiments. In *Proceedings of the 2002 Winter Simulation Conference*, ed. E. Yücesan, C.-H. Chen, J.L. Snowdon, and J.M. Charnes, 45–51. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Box, G.E.P. and N.R. Draper. 1987. *Empirical model-building and response surfaces*. New York: John Wiley.

Box, G.E.P., W.G. Hunter, and J.S. Hunter. 1978. *Statistics for experimenters: an introduction to design, data analysis, and model building*. New York: John Wiley.

Fu, M.C. 2002. Optimization for simulation: theory vs. practice. *INFORMS Journal on Computing* 14 (3): 192–227.

Glover, F., J.P. Kelly, and M. Laguna. 1999. New advances for wedding optimization and simulation. In *Proceedings of the 1999 Winter Simulation Conference*, ed. P.A. Farrington, H.B. Nembhard, D.T. Sturrock, and G.W. Evans, 255–260. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

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. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

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. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Kelton, W.D. 1997. Statistical analysis of simulation output. In *Proceedings of the 1997 Winter Simulation Conference*, ed. S. Andradóttir, K.J. Healy, D.H. Withers, and B.L. Nelson, 23–30. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Kelton, W.D. 2000. Experimental design for simulation. In *Proceedings of the 2000 Winter Simulation Conference*, ed. J.A. Joines, R.R. Barton, K. Kang, and P.A. Fishwick, 32–38. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Kelton, W.D., R.P. Sadowski, and D.A. Sadowski. 2002. *Simulation with Arena*. 2nd. ed. New York: McGraw-Hill.

Kleijnen, J.P.C. 1998. Experimental design for sensitivity analysis, optimization, and validation of simulation models. In *Handbook of simulation*, ed. J. Banks, 173–223. New York: John Wiley.

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

Montgomery, D.C. 1997. *Design and analysis of experiments*. 4th ed. New York: John Wiley.

Swain, J.J. and P.A. Farrington. 1994. Designing simulation experiments for evaluating manufacturing systems.

In *Proceedings of the 1994 Winter Simulation Conference*, ed. J.D. Tew, M.S. Manivannan, D.A. Sadowski, and A.F. Seila, 69–76. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

AUTHOR BIOGRAPHIES

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* (2nd ed., 2002, 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 <www.cba.uc.edu/faculty/keltonwd>.

RUSSELL R. BARTON is a Professor and Associate Dean for Research and Ph.D./M.S. Programs in the Smeal College of Business Administration at the Pennsylvania State University. He has a B.S. degree in Electrical Engineering from Princeton and M.S. and Ph.D. degrees in Operations Research from Cornell. Before entering academia, he spent twelve years in industry. He is Vice President for the INFORMS College on Simulation. His research interests include applications of statistical and simulation methods to system design and to product design, manufacturing and delivery. His email address is <mailto:rbarton@psu.edu>.