Proceedings of the 2009 Winter Simulation Conference
M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls, eds.

REPRESENTING AND GENERATING UNCERTAINTY EFFECTIVELY

W. David Kelton

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

ABSTRACT

Stochastic simulations involve random inputs, so produce random outputs too. This introductory tutorial is meant to call attention to the need to model and generate such inputs in ways that may not be the standard or defaults in simulation-modeling software, yet can be critical to model validity (a.k.a. getting right rather than wrong answers). There are both dangers involved with doing this inappropriately, as well as opportunities to do things better, making for more accurate and more precise results from simulations. Specific issues include possible dependence across and within random inputs, use of empirical distributions even if a "standard" fits the data, and non-default use of the underlying random-number generator. Suggestions for novel ways of implementing some of these ideas in simulation-modeling software are offered.

1 INTRODUCTION

My purpose in this tutorial is to discuss some issues and approaches for effective representation and generation of uncertain inputs when building and exercising computer-simulation models. I won't be going into a lot of technical or mathematical detail, as such issues are very well covered elsewhere (see, for example, Kuhl et al. 2008, and elsewhere in these *Proceedings* as well).

Rather, I'd like to raise some issues that are sometimes (maybe often) overlooked, but that can be important for simulation models and projects. If ignored, these issues can lead to inefficient or imprecise results, or (worse) inaccurate or invalid models. On the other hand, if considered and effectively addressed, there's an opportunity to improve both model validity and precision. I'd also like to suggest some nonstandard, if not downright heretical, ideas on how randomness might be better and more easily represented and implemented in simulation models and software.

Section 2 delineates different kinds of simulation models and inputs. Section 3 discusses the common assumption that all random inputs across a simulation model are independent of each other, and independent and stationary within themselves. Section 4 considers different ways in which observed data from the field might be used to represent randomness in a simulation model. Section 5 briefly discusses ways in which the underlying random numbers might be deployed to effect more precise simulation results. Finally, Section 6 draws some general conclusions.

Some of these notions were discussed previously in Kelton (2001, 2006), and this paper is an update of Kelton (2007).

2 DETERMINISTIC VS. RANDOM INPUTS

Most simulation models, and indeed most operations-research models, might be viewed as having two aspects: *structural* and *quantitative*.

The *structural* components include the logical elements and relationships among them. In an optimization model these would be the form of the objective function, the number and form of the constraints, and whether we seek to minimize or maximize the objective function. In a dynamic simulation model these would include the flowchart topology of entity movements and how they interact.

The *quantitative* components of a model are the values of numerical inputs, or ranges or probability distributions that describe what values these inputs might assume. In an optimization model these could be the coefficients and other parameters

of the objective function and constraints. In a dynamic simulation model of a manufacturing process, these might include the number of servers at a workstation, the probability distribution (including its parameter values) of the time required to process a part, and the probabilistic process governing part arrivals.

Structural modeling might be more interesting (and fun), but quantitative modeling needs to be done right too, as both aspects affect the model's results, and thus conclusions drawn. This tutorial is about the quantitative input aspects in dynamic simulation models, and in particular how random inputs can be represented (or modeled) and realized (or generated) effectively and efficiently.

Most quantitative inputs to simulation models can be classified as *deterministic* or *random*. *Deterministic* inputs are those that have known and unchanging values, perhaps like the number of servers at a workstation if they never change and never break down. *Random* inputs are deemed to vary in an uncontrolled way (though not completely unpredictably), like part-processing times or arrival processes.

My focus here is on representing and generating random inputs, but deterministic inputs are sometimes not known exactly either, even though they're modeled as constant and unchanging. In such situations we might ask about *sensitivity* of key outputs to changes in inputs, and seek to identify and quantify these effects, e.g. via what's known as a *Data Table* in a simple spreadsheet model in Microsoft Excel™, where we specify a range of values for one or two input parameters and "run" the spreadsheet for all combinations of them to see how the outputs vary in response. Sensitivity analysis is common in optimization modeling, and is often a question of interest in simulation models as well.

And speaking of simple spreadsheet stochastic simulation models (usually static rather than dynamic), some inputs might be most appropriately modeled not as constants (even subject to controlled change), but rather as random variables. In these cases, an add-in like @RISK™ (www.palisade.com), or Crystal Ball™ (www.decisioneering.com) could be used to replace those inputs with probability distributions, and the simulation is then "run" hundreds or thousands of times to produce a histogram and output summary measures of key results, allowing assessment of risk, among other things.

But for the rest of this tutorial I'll focus on dynamic, stochastic simulations — those in which time plays a critical role, and in which at least some inputs are not constants but rather "draws" from probability distributions. Such models are exemplified by queueing networks that might represent operations in manufacturing, logistics, a hospital, or a supply chain.

3 THE DECLARATION OF INDEPENDENCE ACROSS AND WITHIN RANDOM INPUTS

In dynamic stochastic simulations it's common to assume that the various random inputs across a model are mutually independent of each other. Further, it's common to model each of these random inputs itself as a stream of independent and identically distribution observations from an unchanging, i.e. stationary, probability distribution.

Indeed, it's difficult with most simulation-modeling software to do anything different from this. And if such assumptions are in fact borne out by the real target system, then fine. But it's easy to think of examples where this would be invalid:

- A patient arriving to an urgent-care facility might require four steps for processing, the times for which might well be positively correlated with each other (four small times for a routine case, but four large times for a more complex or critical case). Declaring them instead to be independent could lead to model invalidity, and indeed to erroneous results. Instead, a multivariate statistical model of the four times should be specified, replete with possible relationships like correlations across the components of the four-dimensional random vector.
- A telecommunications system might receive packets of sizes that are positively auto-correlated within the stream. Thus, large packets tend to be followed by other large packets, and similarly for small packets. Erroneously modeling the sequence of packet sizes as independent and identically distributed could lead to undercongestion in comparison with the true situation that suffers great congestion build-ups during periods of large packets.
- A call center has two peak periods each work day, and "flattening" them to a constant average rate would clearly result in misrepresentation of congestion. In the example in Harrod and Kelton (2006), a "correct" simulation of such a system, i.e. representing the peak and slow times correctly, resulted in a 95% confidence interval of 68.1 ± 5.1 minutes for the expected mean delay in queue; flattening the peaks and filling the valleys to a constant rate with the same daily expected number of arrivals produced a confidence interval of 29.6 ± 4.6 minutes, clearly in serious error. What is needed instead is some kind of nonstationary arrival processes, perhaps the well-known and much-studied nonstationary Poisson process.

The point of the above examples is to illustrate that something other than standard independent and identically distributed input modeling could easily be indicated in many situations. Some simulation-modeling software can represent,

say, nonstationary Poisson processes, but the capability for specifying and generating more general non-independent and nonstationary random structures is not where it should (and could) be at present. Kuhl et al. (2008) detail methods to do this.

A final note here concerns the most comfortable of all distributions, the normal. This is an option provided in most simulation software to model random inputs, and sometimes it's appropriate (e.g., the error in a measurement, going back to Gauss). However, many dynamic simulations have as their random input time durations (processing times, times between successive arrivals, etc.), and using a normal distribution — *any* normal distribution — is just not appropriate in such cases. The reason is that the normal distribution has infinite tails both ways, i.e., is unbounded in both directions, and it's the left side that's troublesome, producing negative values at least once in a while, which make no physical sense for time durations. As any beginning-statistics student knows, if the normal mean is positive and several times bigger than the standard deviation, the chance of getting a negative value is very, very small. But it must be remembered that we're simulating here, and as such are making thousands, easily millions of draws from these input distributions, so an extremely rare event can easily happen, and it's difficult to predict in general how the model will greet a request to move the clock backwards. There are many other similar "hump-shaped" distributions that don't have a negative left tail . . . they're just not as comfortable.

4 USING ACTUAL DATA VS. FITTING VS. EMPIRICAL DISTRIBUTIONS

Typically, there will be some real data from the field available (or budgeted to collect) on the various random inputs to a simulation model. For instance, in an emergency-department simulation there would be data on times to check it, triage, collect vital signs, read radiology work, etc. These data could be used in several ways to represent and generate the random inputs to a simulation:

- Perhaps the most natural approach, and one often asked about, is simply to feed the observed data into the simulation where they belong in the model. Most simulation software has facility for doing this. And there can be no question about whether this is valid in the sense of being true to reality. However, this approach is problematic, for several reasons. First, no matter how large the sample size, it's likely that this data set will be quickly exhausted in long simulations, or those that are replicated many times. Another practical computational drawback is that this tends to be slow, involving intensive input operations. A more fundamental concern is that the simulation model's results are limited to those induced by this particular data set, and are thus not as generalizable as they should be. For these reasons, this approach is usually not advised, despite its intuitive appeal.
- Possibly the standard today is to use statistical methods like maximum likelihood to fit one of a variety of standard distributions to the data on an input, and then use standard generation methods to draw from these fitted distributions to drive the simulation. There are several software packages that do this, either with simulation-modeling software or on a stand-alone basis. This approach has the advantages of filling in anomalous "holes" that might be present in the data, especially in small samples, and extending the range of the representing distribution beyond the data values that happen to have been observed (if that's desirable); these aspects make this approach more generalizable. There are also some disadvantages. Especially in very large samples, which are becoming increasingly common with technologies like RFID, standard goodness-of-fit tests may well reject the fit of all available standard distributions due to their comparatively high statistical power with such large samples. Another practical issue is that, depending on the software, this can be a laborious process if the simulation model has a large number of random inputs. And a more fundamental question is the underlying reason of wanting to find a fit to our observed data from among a relatively short list of probability distributions that were developed decades ago largely for their analytic tractability.
- A middle ground between the above two approaches is to use the data to specify some sort of empirical, or histogram-based input distribution, with no (or, at least, very little) fitting involved. There are several methods for this, ranging from simple histogram specification to more advanced methods like fitting an exponential tail (see Bratley, Fox, and Schrage 1987). This has several advantages. There is almost no issue of poor fit, almost by definition, since there is little or no fitting going on. Once implemented, it is very simple, to the point of being automatic and could happen in the background, perhaps even hot-linking back to a dynamically changing database that is refined as additional real data are collected, though this is not supported by current simulation-modeling software. And, this approach maps very well onto modern data-collection methods and the attendant large (or huge) sample sizes, and becomes smoother and more representative with more data. Generating draws from such empirical distributions is easy and fast, using linear interpolation most of the time in the context of inverse-distribution-function generation, which itself has advantages. But as with the other methods, there are disadvantages too. With a small sample size, this method tends to be high-variance, not filling in the "holes" well at all. And, though hopefully

just temporary, this approach is not as well supported by software as the standard smooth fitting methods in the preceding bullet point.

To close this section consider a radical idea. Rather than looking at specifying input distributions as a separate activity in a simulation project, it would be nice to integrate the simulation model directly back to the database of observed real input data. The intermediate link could be some sort of automatic specification protocol (perhaps one the of empirical distributions mentioned in the third bullet point above), invisible to the modeler, that not only removes this activity from the analyst's conscious to-do list, but also serves to update the simulation model itself as additional data are collected and added to the database, or as conditions change and a new database from a different regime is compiled. In this way, the simulation model and the supporting database would become more integrated, and could be kept current more easily.

5 ASSIGNING RANDOM NUMBERS TO IMPROVE PRECISION

All stochastic simulations rely on a random-number generator, which is supposed to produce a sequence of independent and identically distributed draws from the continuous uniform distribution between 0 and 1. All methods for generating draws from other distributions and processes depend on, and indeed assume, a good random-number generator. There has been much research on constructing good random-number generators, and there are now very good generators available. Few simulation modelers think much about this, and rely on the generator in the simulation-software package. While most simulation-software developers are aware of the need to provide good generators, users have a responsibility to check this out and, if necessary, ask questions of the vendor.

There are also some issues and opportunities in exactly how the random numbers are used and assigned, which can affect the quality of simulation results, especially in terms of precision (low variance). Many of these are discussed in Kelton (2006), with examples, so only a brief summarization will appear here.

First, it's essential to have a modern, long-period generator. All generators will eventually cycle, i.e., return to their starting point and generate the same sequence of random numbers; the time before a generator does this is called its *period*. Clearly, long periods are essential, but unfortunately some still-used generators are several decades old now, during which time computers have become much faster, so much longer periods are really needed now. For instance, many older generators have a period on the order of $2^{31} \approx 2.1$ billion, or on the order of 10^9, which seems like a lot until you realize that this whole cycle can be exhausted in just a couple of minutes on an average PC today, after which the sequence repeats, obviously endangering the simulation's validity. More modern generators have periods on the order of 10^{60} or more, requiring millions of years to exhaust on today's computers (and, if you believe Moore's "law," being good for a couple of centuries yet as computers continue to speed up).

Another issue with generators is that their cycle should be easily divided into *streams*, which are non-overlapping random-number subsequences that are themselves astronomically long, and which can be addressed easily by the modeler. This allows the modeler to assign separate random-number streams to separate random inputs; this approach is called *faucets* in Kelton (2006). Now this is not required for the model to be valid, but it does create a situation where, in comparing multiple scenarios (a common goal in simulation studies), lower variance (i.e., higher precision) will be attained on estimates of the differences in key model responses. While there is a probabilistic reason for this, the intuition is that the multiple scenarios are encountering the same (or, at least similar) external randomness, so any differences in their responses are due to the fact that they are different scenarios, not due to different external random "bounce." The precision gains can sometimes be impressive, depending on the models' structures, and they come at essentially no cost in extra computation, and at only very little cost in terms of the analyst's time and effort.

An idea different from faucets is to pre-generate and pre-assign to simulation entities (customers, patients, widgets, etc.) all the downstream random attributes they might need during their lives in the model, and, as a matter of programming, access them from the entities later when needed rather than generating them there. This can often go a long way toward ensuring that the "same" entities are moving through all the different scenarios. This approach is called *body art* in Kelton (2006).

In general, simulation software does not do such random-number assignment by default, so it is currently up to the user to program the model to make it happen. It seems that faucets could, however, be implemented in simulation software so that it became the default.

6 CONCLUSIONS

In simulation projects there are a number of issues in specifying and generating input random process that deserve some thought on the part of the modeler. Some of these can affect the model's validity (e.g., modeling dependencies and time-dependence appropriately), while others can improve model precision. Simulation-modeling software may not provide all the support that might seem helpful on these issues, so at present, the modeler needs to be aware and take appropriate action.

ACKNOWLEDGMENTS

I thank Natalie Steiger for her gracious invitation and Jim Wilson for his inimitable encouragement.

REFERENCES

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

Harrod, S., and W.D. Kelton. 2006. Numerical methods for realizing nonstationary Poisson processes with piecewise-constant instantaneous-rate functions. *Simulation: Transactions of The Society for Modeling and Simulation International 82*: 147–157.

Kelton, W.D. 2001. Some modest proposals for simulation software: design and analysis of experiments. In *Proceedings of the 34th Annual Simulation Symposium*, Advanced Simulation Technologies Conference, Seattle, Washington, keynote address, 237–242. New York: Institute of Electrical and Electronics Engineers, Inc. Computer Society.

Kelton, W.D. 2006. Implementing representations of uncertainty. In *Handbooks in Operations Research and Management Science, Volume 13: Simulation*, ed. S.G. Henderson and B.L. Nelson, 181–191. Amsterdam, The Netherlands: Elsevier B.V.

Kelton, W.D. 2007. Representing and generating uncertainty effectively. In *Proceedings of the 2007 Winter Simulation Conference*, ed. S.G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J.D. Tew, and R.R. Barton, 38–42. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Kuhl, M.E., E.K. Lada, N.M. Steiger, M.A. Wagner, and J.R. Wilson. 2008. Introduction to modeling and generating probabilistic input processes for simulation. In *Proceedings of the 2008 Winter Simulation Conference*, ed. S.J. Mason, R.R. Hill, L. Mönch, and J.W. Fowler, 48–61. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

AUTHOR BIOGRAPHY

W. DAVID KELTON is a Professor in the Department of Quantitative Analysis and Operations Management at the University of Cincinnati, where he teaches simulation, statistics, and operations research. His primary research interests are in simulation methods and applications. He was Editor-in-Chief of the *INFORMS Journal on Computing* from 2000 to mid-2007, and has served as simulation Area/Department editor for the *INFORMS Journal on Computing*, *Operations Research*, and *IIE Transactions*. He is co-author, with Randy Sadowski and Nancy Swets, of *Simulation With Arena*, now in its fifth edition with McGraw-Hill. He is a member of the INFORMS Simulation Society, and was president of its predecessor's predecessor, the TIMS College on Simulation, from 1990–1992. Since its founding in 2003, he has been a Trustee of the WSC Foundation. He was WSC Program Chair in 1987, WSC General Chair in 1991, and on the WSC Board of Directors from 1991–1999. He is a Fellow of both INFORMS and IIE. His website is <www.cba.uc.edu/faculty/keltonwd/>.