Proceedings of the 1994 Winter Simulation Conference
ed. J. D. Tew, S. Manivannan, D. A. Sadowski, and A. F. Seila

SIMULATION STATISTICAL SOFTWARE: AN INTROSPECTIVE APPRAISAL

Organizer and Chair

Paul J. Sanchez
Indalo Software
12338 Cape Cod Dr.
St. Louis, MO 63146-4645

Panelists

Frank Chance

4121 Etcheverry Hall
Department of IE/OR
University of California, Berkeley
Berkeley, CA

Kevin J. Healy

School of Industrial Engineering
Purdue University
1287 Grissom Hall
West Lafayette, IN 47907-12807

James O. Henriksen

Wolverine Software Corporation
7617 Little River Turnpike
Suite 900
Annandale, VA 22003-2500

W. David Kelton

Department of Operations and Management Science
Carlson School of Management
University of Minnesota
Minneapolis, Minnesota 55455

Stephen G. Vincent

Averill M. Law & Associates
P.O. Box 40996
Tucson, AZ 85717

ABSTRACT

Simulation experiments are sampling experiments by their very nature. Statistical issues dominate all aspects of a well-designed simulation study – model validation, selection of input distributions and associated parameters, experiment design frameworks, output analysis methodologies, model sensitivity, and forecasting are examples of some of the issues which must be dealt with by simulation experimenters. There are many factors which complicate analyses, such as multivariate input distributions, serially correlated model inputs and outputs, multiple performance measures, and non-linear system response, to name a few.

The purpose of this panel is to discuss any and all issues related to software tools available for dealing with these and other problems. I asked five experts from within the simulation community to share their opinions and insights on the availability and quality of software to meet the statistical needs of the simulation community. The position statements provided by them below are intended to serve as a springboard for a more extensive exchange of ideas during the discussion at the conference.

1 Frank Chance

Converting simulation output from raw numbers to system insight is often a difficult task. Good statistical output speeds the process; poor statistical output makes it impossible. At a minimum, simulation output should include automatic generation of confidence intervals and graphical output charts. The simulation should also integrate easily within a design and analysis of experiments package.

Confidence intervals are a good step forward from simple point estimates. However, methods and values used to generate confidence bounds should be placed directly in the output report. For steady-state analysis, the number of batches or batch length should be stated, and for any confidence interval the confidence level must be given. The exact formula used to generate confidence bounds should be noted in accompanying documentation. If the method relies on normally distributed data, then quantile-quantile or probability-probability plots of the data versus a normal distribution should be available (see Law and Kelton, p. 375). For steady-state analysis, one- or two-lag correlations in the batch means should be estimated and displayed,

to indicate the possible need for larger batch sizes.

Graphical output is a basic necessity. For any time series of interest, a plot of the data over time, and its estimated autocorrelation structure should be available (see Law and Kelton, p. 285). For replicated data, a smoothed average across replications is very helpful in understanding a system's transient behavior (see Law and Kelton, p. 545). Standardized time series plots are also useful in this task (see Schruben, p. 205). To aid in fitting autoregressive or moving average models, a plot of the partial autocorrelations is helpful (see Brockwell and Davis, p. 102).

Completing a designed experiment with a simulation model can be a daunting task, unless the generation of design points is automated. Imagine generating, by hand, all the input models or run conditions necessary for a large fractional factorial design. While this issue is not strictly that of statistical output, it is a necessary condition for the successful completion of a designed experiment. Also, the gathering and processing of output data should be automatic. At no time should the user be required to manually scan output reports and enter information from those reports into a statistical analysis package. This task is best left to software.

In summary, three goals seem evident: automatic generation of confidence intervals with accompanying diagnostics, generation of useful graphical charts, and integration of the simulation within a larger design and analysis environment. As the simulation user is freed from mundane tasks, the development of insight should be maximized.

2 Kevin Healy

In many respects, the state of statistical software available for simulation purposes has lagged that of the software tools for building and implementing simulation models. The gap has narrowed recently though in several important areas, the most conspicuous being the general availability of such software. In addition to the offerings of traditional commercial vendors of simulation modeling software, there has been a dramatic increase in add-on and stand alone tools from independent developers. This coincides with the widespread migration of general purpose statistical analysis software to a variety of desktop computer platforms. A notable contribution has been the incorporation of many statistical analysis features into generic spreadsheet software which seems a natural extension given their data management and manipulation capabilities. The tools themselves are also greatly improved, particularly the offerings from the traditional vendors of simulation modeling software which have in the past seemed much like a neglected child. Most would argue that software for such purposes is also more accessible in the sense of ease-of-use. Like other software in general, many benefit from the adaptation and maturation of standard graphical user interfaces that has taken part in the computer industry. Many such capabilities have also been integrated into modeling software in the form of scenario management and analysis modules; a perceived advantage over the "separate but equal" doctrine.

Increased accessibility brings with it the increased potential for misuse, a criticism to which simulation in particular seems more susceptible than other techniques. It seems to me the validation issue poses no more or less of a burden in simulation studies than it does in other methods of investigation. Such problems lie not with the tools themselves but in their application. Proper application always requires an understanding of the fundamental assumptions on which a particular method is based. Most software, as is, serves only as a matter of convenience. To the extent possible though, it would be desirable for software to gauge and inform users regarding the validity of underlying assumptions. Perhaps an expert systems approach would prove useful.

There is also, in my opinion, another important issue concerning the presentation and interpretation of results of statistical analyses. We need to do a better job of conveying the practical significance of such analyses in a way that is as accepted and routine as the more abstract framework of statistical significance. Having solved these problems, we can then direct our efforts toward achieving the most elusive goal of all - devising a convenient mechanism for formatting statistical summary reports.

3 David Kelton

Like everybody, I'd like to see stronger, more comprehensive, and friendlier statistical-analysis capabilities built directly into simulation software. And getting at these capabilities ought to be easy and familiar within the context of whatever simulation environment one chooses, using whatever terminology, logic constructs, data files, and user interface are consistent with that environment. I feel that the reality is that if it requires learning a whole new system, new interface, output-file design, and file export and import between uncongenial formats, the statistical analysis will just not happen.

While statistical analysis of simulation output data, in post-processing mode, is clearly essential, it is far from the whole story, in my opinion. Practitioners' needs for design-and-control software are just as urgent, and maybe more so. I refer here to a capability that would take a general model (already validated and verified) and a simple description of what is to be done with it, and will then go do it, maybe with a guess ahead of time about how long it might take, given experience with this model on this platform.

As an example of a simple experiment (and clearly not the only kind of thing that one might like to do), a model

could be built generally enough to accommodate alternative levels of input *factors* (qualitative as well as numerical) to define alternative configurations, a factorial design would be automatically set up and then carried out, and the results reported perhaps in the form of an ANOVA for starters, estimates of factor effects and interactions, and an estimated response surface (all in graphical format where appropriate). Another example would be to take a general model, a file containing descriptions of the alternative configurations of this model that are under consideration for implementation, and then make whatever simulation runs are required to select the "best" system, or perhaps compare each configuration with the best of the others.

Many of these kinds of things are, right now, possible from the methods point of view. But to *do* them with a model developed in a high-level environment leads a lot of people into a laborious and awkward process of either making all the runs by hand or writing elaborate script or batch files (perhaps themselves generated by a custom-written program), and then writing yet more custom code to do the statistical analysis or perhaps porting to a statistical package.

And not only should such a capability carry out these tasks, but it should also take care to manage things behind the scenes in a statistically valid and efficient way. This would include, for instance, keeping track of the random-number streams to ensure independence or re-use, as appropriate, as well as making runs long enough and in sufficient numbers to ensure statistical precision as specified by the user or as determined endogenously.

These are issues that are of frankly not much interest to many practitioners, but which must be done appropriately and carefully if the simulation project's results are to be effective. And by knowing that these issues are being taken care of silently, users can instead concentrate on what is most important to them—using the model effectively to make decisions.

4 Jim Henriksen

My perspective on the directions of growth for statistical tools to be used with simulation is influenced by years of experience as a software tool builder. In the process of developing software for use by others, one needs to acquire or develop the proper tools to get the job done. You might expect a tool builders' tools to be highly automated, powerful tools of uniformly state-of-the-art quality. Would that it were so! Often, the tools we would like to have simply don't exist, and the effort to create them solely for our own use would be too costly. As a consequence, we make heavy use of *partially* automated tools which perform laborious chores best relegated to a computer, but require an intelligent orchestrator to use them most effectively. Software development is as "man-in-the-loop" as any process could be.

The systems to which simulation is being applied today are more complex than ever. Many of these systems exhibit "reactive" behavior. For example, computer-controlled systems usually contain logic for altering behavior in response to operating conditions. Analyzing the performance of such systems presents a formidable challenge to the modeller. The "design of experiments," to borrow a phrase from the curricula of statistics, is hard to come by. In an ideal world, automated software would design the experiments and, to a great extent, conduct, or assist in the conducting of, the necessary analysis. Unfortunately, we live in a less than ideal world. The technology for fully automated tools does not exist, even in laboratory form, let alone in commercial form. To deal with the lack of fully automated tools, we can follow the example of the software developer, and make intelligent, selective use of partially automated tools. In the next several paragraphs, I will consider several examples.

First, consider the problem of fitting statistical distributions to empirical data. A tool such as UniFit provides excellent capabilities for performing this chore. Given a data set, it *will find* a statistical distribution that "best" fits the data. The extent to which a user participates in the process of finding the distribution is up to the user. One can let UniFit do the entire job automatically, or one can, through the use of graphs and numeric output produced by UniFit, make a decision "manually." I'd like to see this process carried a step further. To me the "name" of the distribution rarely matters - I don't care if the distribution is log-normal. What I'd like to have is the ability to manually modify a chosen density function by using a mouse. This would allow me to act on hunches such as "in reality, I believe the distribution has a longer tail to the right." I could make such changes quickly, and get rapid feedback on their consequences.

Second, consider the problem of determining how long a model must be run in order to get "good" statistics. In the absence of any information about *which* statistics are important, this is an intractable problem. However, if users were given a means of specifying the statistics of interest, a variety of solutions *would* be possible.

Third, consider how one could go about sensitivity analysis for model parameters. While a number of tools have been developed to address this problem, no *universal* tools exist. For example, perturbation analysis can be applied to some systems, but is far from universally applicable. What I would like to have is the ability to easily specify the range of reasonable values for a small collection of model parameters and have software perform a search of the resultant response surface. Absent an automated search, simply *displaying* the response surface would be of great assistance.

Finally, let us consider the ideal user interface for par-

tially automated tools of the variety described above. For the beginner, an *interrogative* interface would probably be easiest to use. Upon request, the software could, through interactive dialog, *lead* the user through the maze of choices. For the more advanced user, a *declarative* interface would be more appropriate. A declarative interface allows a user to *tell* the software, without being *asked*, exactly *what* is to be done (but not *how*). The most advanced users of simulation software will inevitably reach a point where their requirements cannot be specified through an interrogative or a declarative interface. Such users need an *imperative* interface which allows them to specify not only *what* must be done, but *how*. No matter how sophisticated a declarative interface is, it can never provide access to the total universe of possible approaches. In an ideal world, software should contain all three forms of interfaces, with transition from one to another as seamless as possible. Such software would provide an evolutionary growth path for its users, allowing them to *choose* the style of interaction which best suits their needs.

5 Stephen Vincent

In the following position statement I will consider only the simulation input modeling activity and propose a focus for the actual panel discussion. I would like to address first the importance of the input modeling activity itself. In the past, some practitioners have ignored or downplayed the importance of both input modeling and output analysis. Even today, some inexperienced individuals are unaware of the possible dangers implicit in using triangular or normal distributions with guessed parameter values. It is my opinion that proper input analysis is a crucial aspect of a sound simulation study. Therefore, the quality of the input modeling software one uses is of paramount importance as well.

If we are to evaluate the quality of any type of software we must characterize the needs of the intended user audience and the current limits of technology that can be used to satisfy those needs. With regard to simulation input modeling we find:

1. Differing levels of user expertise and needs: Simulation methodology is used now by a wide variety of professionals – not just full-time simulationists. As a result, developers of input modeling software should expect their users to have a wide variety of backgrounds in probability, statistics, and stochastic processes, with few truly expert users. The least experienced users will require extensive procedural support (e.g., expertise must be built into the software).

2. Differing levels of availability of applicable methodology: It is generally recognized that we lack complete theoretical understanding of how to perform simulation input modeling in the simplest situations (e.g., the univariate, IID case with or without data). (We are virtually ignorant concerning the more difficult situations [e.g., univariate non-IID cases and multivariate cases].) As a result, we should expect developers of input modeling software to consider only the simplest data analysis situations. Further, the lack of completely specified methodology for even these simple cases invariably requires fundamental research and development by developers.

I propose that in the panel discussion we focus our attention on the least experienced users and the simple data analysis situations that they are likely to attempt (e.g., univariate IID case). We can therefore dispense with discussion of what all must acknowledge to be the limited support that software gives the expert user for the extremely difficult multivariate cases.

REFERENCES

BROCKWELL, P.J. AND R.A. DAVIS. 1991. *Time Series: Theory and Methods*. Springer-Verlag.

LAW, A.M. AND W.D. KELTON. 1991. *Simulation Modeling and Analysis*. McGraw-Hill.

SCHRUBEN, L.W. 1994. *Graphical Simulation Modeling and Analysis: Using SIGMA for Windows*. Boyd & Fraser.

AUTHOR BIOGRAPHIES

FRANK D. CHANCE is a visiting assistant professor in the Department of Industrial Engineering and Operations Research at the University of California, Berkeley. His research interests include factory performance analysis via simulation and queuing approximations, simulation output analysis, and transient simulation.

KEVIN J. HEALY is an assistant professor in the School of Industrial Engineering at Purdue University. He holds B.S. and M.S. degrees in Industrial Engineering from The Pennsylvania State University and a Ph.D. in Operations Research from Cornell University. He has taught numerous simulation courses at both the undergraduate and graduate level as well as short courses to practitioners of simulation from industry, government and academia. In a former life, he was Vice-President of the Systems Modeling Corporation and a key developer of the SIMAN and Cinema simulation software. His research interests include stochastic optimization, simulation output analysis, and modeling formalisms.

JAMES O. HENRIKSEN is the president of Wolverine Software Corporation, located in Annandale, Virginia (a suburb of Washington, D.C.) He is a frequent contributor to the literature on simulation and has presented many papers at the Winter Simulation Conference. He has served as Business Chair and General Chair of the conference, and was chosen as one of four keynote speakers for the twenty-fifth anniversary conference. He has served on the Board of Directors of the conference as the ACM representative.

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. degree in Mathematics from the University of Wisconsin–Madison, an M.S. degree in Mathematics from Ohio University, and M.S. and Ph.D. degrees in Industrial Engineering from Wisconsin.

PAUL J. SANCHEZ is the president of Indalo Software, specializing in simulation training, consulting, and custom software development. After earning his bachelor's degree in economics from MIT, he worked for several years before returning for graduate studies. He received his Ph.D. in Operations Research from Cornell. His research interests span many areas, including statistical experiment design, response surface methodology, random number generation and testing, simulation data structures, and object-oriented modeling. He is a member of TIMS, and was the local arrangements chair for the 1991 Winter Simulation Conference.

STEPHEN VINCENT is Vice President for Software Development of Averill M. Law & Associates and is the co-developer of the UniFit II software package. He received his Ph.D. in Management Information Systems from the University of Arizona and has B.S. and M.S. degrees in Industrial Engineering from the University of Wisconsin- Madison. He was an Assistant Professor and taught courses in simulation modeling and software engineering at the University of Wisconsin-Milwaukee.