13

How to simulate if you must

Peter Freeman
Department of Mathematics, University of Leicester

13.1 Summary

A simulation is not something to be undertaken lightly. It is demanding of time, effort and skill. Without very careful planning it can be even more demanding and have a good chance of reaching no clear conclusion. Its implementation on a computer relies heavily on the quality and efficiency of a pseudo-random number generator. The analysis of the results and the reporting of the whole exercise deserve far more detailed attention than they usually receive.

A few techniques for getting more information given the same amount of computer time are briefly discussed.

13.2 Introduction

The advantages which simulation can bring to understanding the many complex processes which interest archaeologists are by now well known and this paper will not discuss them further. We shall instead concern ourselves with ways of ensuring that a simulation exercise can be conducted so as to deliver maximum benefit for minimum effort. Many papers in the archaeological literature that report such exercises show clearly that the authors (and their computers) have invested a lot of time on their project, probably much more than they had envisaged at the outset. Yet many of them leave me wondering whether the subject has been advanced at all, whether the results really do shed much light on dark problems and whether, as a statistician, I am convinced that the results are valid and reproducible. What follows, then, is a personal view of the perils and pitfalls of simulation.

13.3 The stages of a simulation

Other authors (*e.g.* Hamond 1978) have divided the process of simulating into component stages:

1. hypothesis conceptualisation;
2. model construction;
3. computer implementation;
4. hypothesis validation;
5. publication.

I shall broadly go along with this classification of activity and discuss each stage in turn, with most to say about stages 3 and 5. Archaeologists will no doubt disagree with this emphasis, as it is in the first two stages, surely, that the main interest lies. Here is the chance to be creative, to exercise imagination, to cross-relate knowledge from widely different areas and generally to show all those intellectual qualities that earn admiration from colleagues and job offers from vice-chancellors. Yes indeed, but ideas are cheap and it is sorting good ideas from bad, in terms of agreement with observation, that is both tedious and expensive. Before going into more detail, however, let me expand upon what I think is the most important stage of all, before the other five:

0. Must I simulate?

The decision to embark on a simulation is not one to be taken lightly or wantonly. The project will take up a large part of your life. It will grow and develop away from your original simple conception. It will require you to develop skills you never thought you had any interest in acquiring. It will ask you to specify precisely all sorts of quantities that you really haven't got the foggiest idea about. You will have to conjecture and speculate about things on which you have no hard information whatsoever. It will, almost certainly, make you wish at some point that you had never started. So, before taking that first fateful step, you should think long and hard about what it is you want to achieve, whether it is achievable, what benefits achievement might bring, and whether they are worth all the effort that will be needed. Explicitly listing aims and objectives is an indispensable way of clarifying thoughts. Your first attempts at such a list will probably be far too ambitious. Go back over and over again simplifying and cutting down. A simple simulation will turn out to be much more complicated than you expect, but it has some chance of producing results that bear some relation to reality. With a complex simulation that chance tends to zero just as the amount of effort inexorably tends to infinity.

Careful preliminary thought also needs to be given to the question of whether the simulation can break out of its dangerous propensity to be circular. You will obviously need to draw on all your existing knowledge to construct the most realistic model you can. The results of your efforts will presumably be judged by seeing how closely they resemble known facts, but if those facts are the self-same ones that you fed into the system in the first place, the final outcome can only be a massive 'So What?' Any model can only properly be validated against a completely separate set of facts and if there are none such available that's a pretty sure sign that the problem being tackled is too esoteric to be of interest to anyone else. Even producing good validation isn't the end of the process, of course. Whether other, completely different, models exist that accord just as well, if not better, with all known facts must remain an open question. The value of any simulation lies not in how well it fits so much as in what further testable hypotheses it generates and in what predictions it makes if it is assumed to be 'true'.

13.4 Hypothesis conceptualisation

This is the stage at which all the big questions about the balance between simplicity and complexity have to be faced. Your resolve to stay within the limitations of scope you hopefully imposed upon yourself at stage 0 will be sorely tested. The real world is complex, so no simple model can be expected to match it, yet every step to elaborate the model has to be resisted if at all possible, as a complex model will end up no easier to understand than the real process it tries to explain. Many so-called expert systems for medical diagnosis have now become so complicated by accretion of more and more decision rules that no single person can any longer understand how they reach their conclusions and doctors are rightly mistrustful of using what verges on the high technology version of black magic. A simple model, on the other hand, even an obviously and absurdly simple one, can well show clearly, by the kinds of discrepancies between its predictions and the real world, what important factors have been omitted. Such a 'stepwise-up' approach is far more likely to succeed than an all-embracing 'cathedral' approach dripping with gothic ornamentation.

13.5 Model construction

Here come all the hard questions! Vague general hypotheses have to be developed into sharp particular ones. What entities are to be considered, how many of them? What of all the unknown quantities that inevitably crop up? Which are to be treated as fixed and known, which as fixed but unknown and which as random? Which will be allowed to change over time and, if so, how? Exactly how will entities in the system interact? Will feedback mechanisms tend to produce stable, cyclic or unstable behaviour?

As a very simple example, consider simulating the dispersal of artefacts from some central source (see Hodder & Orton 1976). At stage 1 we merely need to envisage an artefact embarking on a journey consisting of a sequence of random steps before finally coming to rest in the place where it is eventually found. But at stage 2 we must decide how many steps there will be. A constant number? If so, how many? If not, what probability distribution will the random number of steps have? What directions will the steps take? Will it be purely random or will there be a tendency to prefer one direction over others, producing a general 'drift' to the dispersal? Is there any barrier to restrict the dispersal (as with a mountain range or coast)? What are the lengths of the steps? Are they constant or random?

One can, of course, dodge all these questions by answering 'Don't know, try it both ways' but the consequences can be serious. The number of combinations rises alarmingly. Even in this simple example Hodder and Orton actually used

- **number of steps**: 2, 6, 10, 14, 18, 22, 26 and uniformly distributed between 1 and 2, 4, 6, 8, 10, 14, 18
- **step lengths**: 0.25, 0.5, 1.0, exponentially distributed with mean 0.5, 1.5 or uniformly distributed between 0 and 1
- **directions**: none or outward movement only (no drift)
- **barrier**: none or circular radius 0.75, 1.5 (probability 0.5 of crossing)

Not all of the $14 \times 6 \times 2 \times 3 = 504$ combinations are actually reported, of course, but enough to show that there are many combinations that give rise to patterns of spatial distribution closely similar to each other and to observational patterns too.

There is not much difficulty actually simulating all these combinations since it's easy to embed the basic program in a set of nested loops, but the labour-intensive task of looking at the un-nested set of computer outputs is likely to induce distinct regret for being so free with the DO loops.

It is also dangerously easy to set up an iteration between stages 1 and 2, modifying the simulation to reproduce ever more complicated models. With artefact distribution, for example, a hierarchy of random walks can easily be conceived, from primary source to secondary distribution centres, to tertiary ones and so on all the way down to your friendly corner stone axe retailing shop. The number of combinations to be considered now becomes astronomical, or at least exceeds the number of archaeologists in the known universe.

13.6 Computer implementation

Provided all the questions have been posed and answered in the previous stages, it should now be straightforward to convert the model into a flowchart and thence into a program in whatever is your favourite language. Getting the program to work and then checking that it is working correctly are not exactly trivial exercises but the value of very careful checking at every stage cannot be overemphasised. The number of hours of computer time wasted by subsequent discovery of programming mistakes can be enormous.

At the core of any simulation program is the random number generator—or rather pseudo-random since all computers, except those built for very special jobs such as picking lottery winners, are deterministic machines. Most machines and most programming languages have such generators built in these days, but the number one golden rule of all simulation is **do not trust them**. There are so many past instances where such generators have been found to have disastrous properties, thereby invalidating all work that had used them, that even in these more enlightened times it is prudent to use a generator that you know has good properties. For example, the FORTRAN subroutine RANDU was for some years the most widely-used generator in the world. Its individual values appeared random enough but each value was very highly predictable from the two previous values, so that any program that relied on sets of three or more pseudo-random values generated successively gave very far from the random ones it purported to produce.

So, do not rely on the simple RAN or RND or RNDM commands within the programming language you are using. While it is impossible to find a single algorithm that will work well at all times on all machines, the one given by Wichmann & Hill 1982 seems to me to come close to being generally satisfactory. The problem of dependencies between pairs and triples is particularly acute on machines with short word length. Here the algorithm by Kral 1972 seems to give good results.

All pseudo-random generators need a SEED, a number to start off the sequence. The user is sometimes saved the job of supplying this by a piece of machine code that automatically picks a seed from the computer's internal clock or from some other register whose contents change rapidly. This is never a satisfactory procedure and should always be avoided. Insist on specifying your own seed, use a table of random numbers to choose one, and keep a note of what it is. You will then be able to rerun your program using exactly the same sequence of random numbers if this is required. Similarly, keep a note of the value of the seed at the end of each simulation run (the value gets changed each time the generator is called) so that you can, if necessary, restart the sequence from where you left off.

I will not go into detail here of all the many tests that can be used to check that a sequence of numbers is suitably random. The book by Kennedy & Gentle 1980 gives a readable survey, but see Knuth 1981 for the full works. It should not be necessary actually to program these tests into your simulation since they would markedly slow down the operation of the program and there are so many of them that almost any generator will fail a few of them sometimes. Two safeguards I would highly recommend (but which sadly I have never seen done in practice) are to repeat the whole simulation process using

1. two or more different seeds, and

2. two or more different generators.

This may sound like a counsel of perfection but it really is little more trouble to replace a single run of, say, 1000 replicated simulations by four runs each of 250 replications. It is then a simple matter to feed all the important quantitative features of the results into a statistics package to do a two-way analysis of variance between seeds and between generators. If, as you expect, there are no differences anywhere near to statistical significance, you can happily pool all 1000 replications and smugly reassure your readers that the results are valid. Equally, if anything significant does turn up, then you know your results are meaningless and your reports will not get consigned to the vast literature on non-reproducible results. A post-mortem into which generator is the dud one and why will quickly lead to a replacement and eventual, demonstrable, success.

The pseudo-random generator will be at the very heart of your simulation. It will get called thousands, perhaps millions, of times, so efficiency is vital. Saving a few milliseconds per call can knock hours off running times. This applies even more dramatically when you want to generate numbers at random from some distribution other than the uniform. The normal distribution is probably the most frequently required, and here it is best to feed a random uniform value into an efficient inverse normal algorithm, such as Beasley & Springer 1977. The simple method of generating twelve random uniforms, adding them and subtracting 6 is neither accurate nor efficient, while the Box-Muller transformation, which is theoretically sound, can give disastrous results in practice—see Neave 1973. An exponential distribution with mean k is best simulated by calculating $-k \log(U)$ where U is a random uniform. Taking the integer part of this quantity gives an observation from the (discrete) geometric distribution. Efficient algorithms for other distributions are given by Ahrens & Dieter 1974, Cheng & Feast 1979 and Kinderman & Monahan 1980. An excellent survey, albeit rather mathematical, is Ripley 1983.

It is sometimes desirable to sample values not from some theoretical distribution but from some large database of actually observed values. There is no objection to doing this, of course, save for the circularity problem mentioned earlier. McLeod & Bellhouse 1983 give an efficient algorithm, even when the number of items in the database is initially unknown.

The next major consideration is the accuracy, or rather reliability, of the results of a simulation. Because such results are, by their very nature, random they can only indicate the values of underlying 'true' quantities to within certain limits of accuracy. These limits depend on the number of replications on which the results are based, and they should ALWAYS be shown in published summaries of results (again something I have never seen in my limited reading of the archaeology literature). It is never sufficient to run, say, 1000 replicates of a simulation of the way Stone Age widgetts have been buried by the action of earthworms and bird droppings, and merely to report the mean depth of buried widgetts. The variability of those 1000 simulated depths also needs recording so that a standard error may be attached to that mean depth. All graphs and tables of such means should include two-standard-error limit values. Many papers I have seen would not only have been improved by the inclusion of such values, but the authors would have been saved much speculation about apparent differences that were actually well within the range of error variation. Most simulations are far too small and some are ludicrously small, with little or no replication at all.

A little forward thinking will sometimes enable a rough estimate of how many replicates will be necessary. If one is trying to estimate some kind of proportion, for example, and if that proportion is thought to be around 50%, then 100 replications will only determine it to within $\pm 10\%$ and 10000 are needed to get within $\pm 1\%$.

To summarise this section, then, you will have gathered by now that I think there is much room for improvement in the conduct (or at least reporting) of simulations in archaeology. The following is a list, extended from that of Hoaglin & Andrews 1975, of pieces of information that I should expect to find in any simulation report:

1. what computer, what programming language;
2. what generator(s), what seed(s);
3. what algorithms for non-uniform values;
4. how many replications; and
5. what standard errors of all summary statistics.

13.7 Variance reduction

Simulations can often be disheartening in that if you do go to the trouble of putting standard errors on estimates, they turn out to be depressingly large and there seems no alternative but to go back and do many more replications. There can, however, be clever ways of getting the variances down without resorting to such brute-force tactics. Usually they have to be devised separately for each individual application, but the basic principle is to get as much value as possible out of each pseudo-random number.

One good general rule is that if you want to simulate a model under several different combinations of conditions then you should use the same sequence of pseudo-random numbers for each combination (by starting with the same seed). This will greatly clarify the differences between the results for the various combinations, at the (slight) risk of biassing all the results if a peculiar sequence of numbers happens to turn up. Hence the earlier advice to repeat the whole thing with one or more different seeds.

Two other more technical tricks for reducing variance are to use so-called antithetic or control variables. As an example of the former, consider the problem of estimating the value of π. There are hundreds of ways of doing this but we shall consider just three:

Hit or Miss. Think of a square whose sides are of unit length and which has a quarter of a complete circle of unit radius, centred on one corner, drawn inside it. The area of the quarter-circle is $\pi/4$, so that is the probability that a point chosen at random in the square will fall inside the quarter-circle. A random point is easily obtained by generating two pseudo-random numbers U and V, and that point will be in the circle if its distance from the origin is less than 1. If we repeat this n times, so using $2n$ random numbers, and find that r points fall inside the circle, our estimate of π is $4r/n$ and this has variance $2.697/n$. If we want our estimate to be accurate to two decimal places, we need to have four standard errors less than 0.005, and hence need n to be at least $1,726,080$.

Crude. It is easily shown that if U is a uniform variate, the average value of the square root of $1 - U^2$ is $\pi/4$, so we simply generate $2n$ numbers $U_1, U_2, ..., U_{2n}$ and calculate

$$\frac{4\sum\sqrt{1-U_i^2}}{2n}$$

as our estimate of π. This has variance $0.398/n$, about one-seventh of the previous value. Notice how we are now using all the actual values of the U's we generate, not just whether or not their values satisfy some condition.

Antithetic. This hinges on the fact that if U is uniform, so is $1 - U$, and so the average of the square root of $1 - (1 - U)^2$ is also $\pi/4$. Generating $2n$ random numbers as before therefore yields

$$\frac{4\left[\sum\sqrt{1-U_i^2}+\sum\sqrt{1-(1-U_i)^2}\right]}{4n}$$

as our estimate of π, with variance $0.042/n$, a reduction factor of one sixty-fourth, bringing the required value of n down to a mere 26,880. Note how this time each random number gets used twice. It is the negative correlation between U and $1 - U$ that produces such a striking reduction of variance in this example.

The other trick is to use a control variable, one whose distribution is known but which is correlated with the variable you are interested in. This is often useful when you have some test statistic whose sampling distribution you do not know.

Full details may be found in the excellent book by Morgan 1985.

13.8 Hypothesis validation

It is difficult to generalise about how the results of a simulation should be compared either with real data or with the predictions of some theory or other. The value of careful thought and planning at stages 0 and 1 now becomes apparent, since the computer runs will have been carefully targetted at their objective and the volume of printed output kept down to reasonable limits (measured in milliforests at most). Informal visual inspection of graphs and plots and more formal goodness-of-fit tests both have their uses. It is generally those places where there is lack of fit that prove most informative, indicating what aspects of the data are inadequately explained and which parts of a model need modifying. A good fit creates danger of complacency, to which the 'so what?' and 'how many other good fits?' questions are rapid antidotes.

Often the simulation can indicate the need to look at aspects of the data not hitherto considered. Hodder and Orton found, for example, that fall-off patterns (the ways in which numbers of artefacts decrease with increasing distance from the source) can be simulated under a wide variety of model assumptions. This strongly suggests (as is obvious anyway) that other ways of looking at the data, such as tendency to cluster or to fall around radial trade routes, need to be used to distinguish between models or to suggest new ones.

13.9 Discussion

Simulation clearly has a useful rôle to play in archaeology. As with other highly complex systems, understanding can be gained of how human societies worked and gave rise to the phenomena we observe today. The computer can even experiment on such societies and explore how they would have behaved under a wide variety of conditions. It cannot, however, provide a substitute for clear thought and careful planning, and it is always ready to lead the unwary simulator into the trap of an ever-increasing program, producing (in the opposite direction to the usual definition of an expert) less and less information about more and more. It should never be forgotten that a simulation is an experiment, and so should be planned and conducted with as much care and control as in any other scientific discipline. Its results are just as subject to variability as those of any other experiment, and should be treated as such. Finally, the whole exercise should be written up with the same rigour, informing the reader exactly what was done and how. We surely owe it to our long-dead ancestors to at least simulate them properly, as we ourselves would like to be simulated.

References

AHRENS, J. H. & U. DIETER 1974. 'Computer methods for sampling from gamma, beta, poisson and normal distributions', *Computing*, 12, pp. 223–246.

BEASLEY, J. D. & S. G. SPRINGER 1977. 'Algorithm AS111—the percentage points of the normal distribution', *Appl. Statist.*, 26, pp. 116–121.

CHENG, R. C. H. & G. M. FEAST 1979. 'Some simple gamma variate generators', *Appl. Statist.*, 28, pp. 290–295.

HAMOND, F. W. 1978. 'The contribution of simulation to the study of archaeological processes', *in* I. R. Hodder, (ed.), *Simulation studies in archaeology,* Cambridge University Press.

HOAGLIN, D. C. & D. F. ANDREWS 1975. 'The reporting of computation-based results in statistics', *American Statistician,* 29, pp. 122–126.

HODDER, I. R. & C. R. ORTON 1976. *Spatial analysis in archaeology,* Cambridge University Press.

KENNEDY, W. J. & J. E. GENTLE 1980. *Statistical computing,* Dekker, New York.

KINDERMAN, A. J. & J. F. MONAHAN 1980. 'New methods for generating student's t and gamma variables', *Computing,* 25, pp. 369–377.

KNUTH, D. E. 1981. *The art of computer programming,* Addison-Wesley, New York.

KRAL, J. 1972. 'A new additive pseudorandom number generator for extremely short word length', *Inf. Processing Letters,* 1, pp. 164–167.

MCLEOD, A. I. & D. R. BELLHOUSE 1983. 'A convenient algorithm for drawing a simple random sample', *Appl. Statist.,* 32, pp. 188–190.

MORGAN, B. J. T. 1985. *Elements of simulation,* Chapman and Hall, London.

NEAVE, H. R. 1973. 'On using the Box-Muller transformation with multiplicative congruential pseudo-random number generator', *Appl. Statist.,* 22, pp. 92–97.

RIPLEY, B. D. 1983. 'Computer generation of random variables: a tutorial', *Int. Statist. Rev.,* 51, pp. 301–319.

WICHMANN, B. A. & I. D. HILL 1982. 'Algorithm AS183 an efficient and portable pseudo-random number generator', *Appl. Statist.,* 31, pp. 188–190.