

Title: When Number-Crunching PCs Leave a Bad Taste

Summary: Small PC-introduced arithmetical errors are not a big deal. But when amassing and aggregating numbers, such as for derivative transactions, things can get messy. Conventional quant and risk analysis wisdom says that inaccurate results come from inadequate models. True, but sometimes what really is at work is that computers cannot add or multiply accurately or reliably the same way each time.

Who'da thunk PCs could make mistakes? In general, small PC-introduced arithmetical errors are not a big deal. But when amassing and aggregating numbers, such as for derivative transactions, things can get messy. PC user: Beware or be square.

Conventional quant and risk analysis wisdom says that inaccurate results come from inadequate models. True, but sometimes what really is at work is that computers cannot add or multiply accurately or reliably the same way each time.

Of the many limitations of computers that impact risk management, consider these three:

1. Finite number storage capacity.
2. (Un)graceful failures.
3. Nonuniformity of standards.

Finite Capacity and Round-Off Errors

Consider the problem, for example, of round-off errors. A computer, *any* computer, can only retain a finite number of significant digits to represent the result of an operation. If and when a result cannot be expressed exactly, a round-off error is introduced. Errors accumulate each time a result that cannot be exactly represented is combined with other inexactly represented results. In the real world of managing complex derivative models, calculations run up against this limitation again and again. When they do, there is the real potential to accumulate enough errors to create suboptimal decisions.

For example, in a binomial price model, an everyday derivative problem, there is a random walk in the logarithm of the price, as shown [below] in a simplified two-figure example. As the number of steps becomes large, the price distribution approaches what is called lognormal. The tails of the distribution contain the extremes of both potential profit and loss and are often what determine whether or not an option on the equity is expected to be profitable. You get different answers if you add from top to bottom or vice versa in this hypothetical two-figure computing machine. Investing in computers with powerful number-crunching abilities can reduce these errors but not eliminate them.

(Un)graceful Failures

When a derivative or any other computation encounters data or other conditions outside the expectations of the creator of the routine, there is potential for one of three situations:

1. The computer routine fails “gracefully,” providing a message to help diagnose the problem.
2. The routine can fail “ungracefully” with system error codes that obscure the real problem.
3. The routine can produce a computed result that is incorrect and the application proceeds as if the results were correct.

This third possibility is especially troublesome for financial applications that project future conditions, such as option pricing methods, where it might be difficult to warn of clearly unreasonable answers.

Nonuniformity of Standards

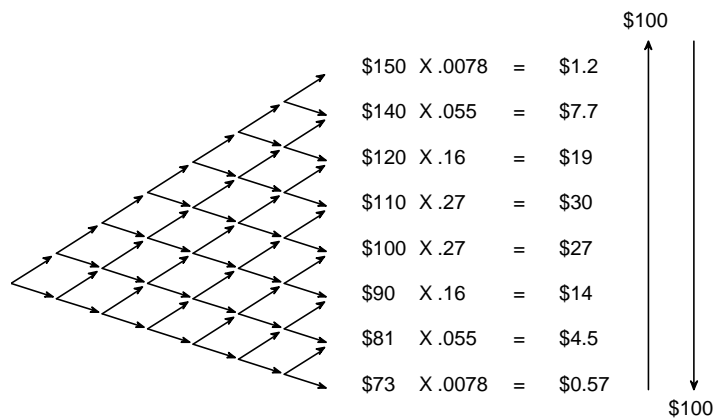
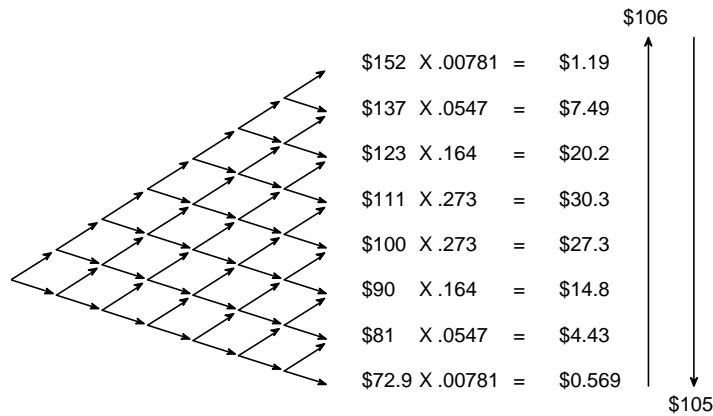
With the large datasets and multi-step processes of option pricing methods, the “same” application, using the same data, can and does sometimes produce different results on different platforms due to differences in how low-level math functions are implemented. This problem is also somewhat akin to the problems of round-off errors discussed earlier, in that such errors can accumulate. So what can you do to protect yourself?

Get Proactive

First, protect your organization from nonuniformity of standards by using code for the underlying building blocks of programs that has been rigorously tested for uniformity on all major platforms. When the inevitable operating system upgrade or processor change happens, your organization is then spared the effort of either independently validating results on new platforms or leaving the impact of multiple platform inconsistencies unexplored and unaccounted for in results.

The best building block algorithms are cognizant of failure potentials and do their best to work around them. They give the user not only an “answer” but also an error estimate or warning of detected inaccuracy. The trick, so to speak, is to ensure that such mechanisms are built into the basic building blocks of the routines used to create computing models for derivative calculations.

Using extensively stress-tested algorithms from commercially available libraries is the tried and true method to protect your organization from the inherent limitations of computer arithmetic. When internal staff develop code for basic math and statistical functions, resources and expertise are rarely available to stress-test their building block algorithms and determine their limitations. Similarly, the higher level math and statistical software packages that many quants have brought from their university and other experiences, while offering many useful and innovative routines, cannot always address all of the requirements of particular financial models. Practically speaking, extensively tested algorithms are well documented, providing a buffer against the inevitable quant turnover in organizations. The bottom line is this: Computer limitations can create problems with your models, but these can be overcome by using trusted code and good practices.



By Rob Meyer. Meyer is EVP of the Numerical Algorithms Group (NAG), a worldwide organization dedicated to developing mathematical and statistical components and 3D visualization software for professional developers.

Originally published by The RMA Journal, September 2001.

Numerical Algorithms Group

www.nag.com

infodesk@nag.com