Numerical Analysis

Gordon K. Smyth

May 1997

Numerical analysis is concerned with the accurate and efficient evaluation of mathematical expressions, especially on computers with floating point arithmetic. While scientists have always been concerned to some extent with numerical computation, the modern discipline of numerical analysis is almost entirely a product of the period since 1950 during which there has been an explosion in the availability of electronic computers. There are three main issues:

- To organize computations so that there is minimum accumulation of error in floating point arithmetic;
- To organize computations efficiently, so that they consume the least possible resources; and
- To obtain accurate numerical approximations to quantities which may not have explicit mathematical expressions.

In other words, do it accurately, do it quickly, do it cheaply.

It might be thought that numerical evaluation consists largely of translating textbook formulae into a computer programming language, a mere coding exercise, but this is very far from the case. Direct translations of expressions from mathematical theory are seldom optimal, and very often are found to fail in circumstances when a less obvious numerical process would have succeeded.

The focus on numerical results means that one is not limited to direct expressions, but can evaluate functions which are defined only indirectly, for example through integrals, differential equations, series or as solutions to equations. An important example is the maximum likelihood estimator for a nonlinear statistical model. Indeed, an indirect method is often preferred for numerical evaluation even when a direct expression exists.

While efficiency and accuracy are both aims, it is accuracy which takes precedence, since a slightly slower accurate program is invariably preferred to a faster one with unreliable accuracy. Errors arise from three sources: (i) errors in the input data, (ii) computation errors due to finite precision arithmetic and (iii) approximation error. The first of these are not under the control of the calculation; in fact it might be considered to be the special concern of the statistician. Computation error appears because of the difference between exact arithmetic and the finite-length arithmetic available on digital computers and hand calculators. Approximation error occurs when the computed expression is not exactly equal to the theoretical quantity even in exact arithmetic. An integral is replaced with a sum, for example, or an infinite series is evaluated only to a finite number of terms.

Efficiency is usually measured by counting basic floating point operations (flops), such as additions, subtractions, multiplications and divisions. Another consideration is to minimize the use of computer memory and other space requirements, especially for large jobs. More recent concerns which arise from modern computer architecture include parallel computing (designing algorithms so that they can be evaluated in parallel streams on fast computers with multiple processors) and local referencing (minimizing unnecessary paging of virtual memory).

It is also desirable to keep programs simple and understandable, thus making the programs easy to maintain and to modify. Users must often choose between using compact programs which can be tinkered with for their own use, and using sophisticated high performance software from public libraries, which cannot be modified and must be taken somewhat on trust.

Most biostatisticians can benefit from familiarity with numerical analysis. An understanding of the numerical methods being used and an idea of when they will perform well or poorly is necessary even for users of standard statistical package programs. You must still understand the program's purpose and limitations to know whether it applies to your particular situation or not. More importantly, many problems cannot be solved by simple application of a standard program. If you develop your own software, a knowledge of numerical analysis can help avoid numerical pitfalls that can occur easily in a number of problems.

A justifiably popular text on scientific computing is Press et al [9], which contains a lot of advice on routines to use. Other good general texts on numerical analysis are Atkinson [3] and Stoer and Bulirsch [12]. An introduction with some statistical orientation is Thisted [13]. An elegant and elementary introduction to the fundamental ideas of numerical analysis is given by Stewart [11].

This article discusses the basic ideas of accuracy and describes briefly key topics in numerical analysis which are treated at more depth in separate articles. Pointers to available software are given at the end of the article.

Conditioning

The concept of conditioning refers to the intrinsic difficulty of a numerical problem. A problem is ill-conditioned if it is sensitive to perturbations in the data, and well-conditioned if it is not. Conditioning is often quantified by a *condition number* which refers to the amplification of relative errors. Suppose that x is the exact argument to a function f, but unfortunately only an approximation $\tilde{x}$ is available. The *condition number* κ of f at x is defined, with respect to a given norm ($\|\cdot\|$), by the relation

$$\frac{\|f(\tilde{x}) - f(x)\|}{\|f(x)\|} \approx \kappa \frac{\|\tilde{x} - x\|}{\|x\|}$$

for $\tilde{x}$ near x. If κ is large, then errors in x are magnified in the evaluation of $f(x)$, while the opposite is true if κ is small. If $\kappa = 10^k$, then k is roughly the number of significant figures of accuracy we can expect to lose in the computation.

For univariate, differentiable functions, the condition number is essentially $\kappa = |xf'(x)/f(x)|$. For example, $f(x) = (x-1)^6$ is an ill-conditioned function near $x = 1$, while $f(x) = x^{1/2}$ is well-conditioned for any $x > 0$.

For general multivariate functions, the specific definition of condition number depends on the problem. For example the computation of regression coefficients from a multiple regression is ill-conditioned when the design matrix X displays collinearity. Conditioning also depends on the quantity of interest. A least squares regression may be ill-conditioned from the point of view of the regression coefficients, but well-conditioned from the point of view of the fitted values.

Stability

A stable algorithm is one which evaluates a function to the accuracy allowed by the function's condition number. A stable algorithm therefore will evaluate a well-conditioned function accurately, and will do as well as can be expected on an ill-conditioned problem. For example consider the problem of computing the sample variance of the three numbers:

$$62, 63, 64$$

using four digit decimal arithmetic. A commonly taught formula for the variance is

$$s^2 = \frac{1}{n-1}\left(\sum_{i=1}^{n} x_i^2 - n\bar{x}^2\right)$$

where n is the sample size and the x_i are the observations. Since the data are given to two significant figures, it might be thought that carrying four significant figures through the calculation will leave a more than adequate safety margin. In this case $\sum x_i^2 = 3844 + 3969 + 4096$ which is 1191×10^1 in 4 digit arithmetic. Similarly $n\bar{x}^2 = 3(63^2)$ is 1191×10^1 to 4 digits. Therefore s^2 is computed to be 0, a 100% error compared with the true value of 1. Alternative algorithms are available: for example, $s^2 = \sum(x_i - \bar{x})^2/(n-1)$ which evaluates to $[(-1)^2 + 0^2 + 1^2]/2 = 1$, the correct answer in this case. The first formula is unstable, while the second formula is stable. There are many other algorithms for computing the sample variance, some of great interest to manufacturers of hand calculators. See Chan et al [4] for a discussion.

The error in the first formula above arises in the rounding errors of $\sum x_i^2$ and $n\bar{x}^2$, and the error is revealed when the difference is taken of the two large and nearly equal quantities. This is often called *subtractive cancellation*, although rounding error occurred not in the subtraction but in the previous summation. It is a general principle that one cannot add a large value to a floating point number and later subtract it without losing accuracy. One concern therefore of numerical analysis is to limit the growth in size of intermediate quantities in calculations. For example, a summation is generally stable if the summands are all of one sign. In this case the partial sums cannot be greater in absolute value than the final sum.

There is often a close relationship between stability in numerical analysis and in statistics. Frequently parameters which are statistically interpretable because they measure some invariant characteristic of a problem appear also in a stable algorithm, because of the need to compute quantities which do not grow without bound. Even the small example above gives an example of this, as the $x_i - \bar{x}$ are the well known residuals, while in the textbook formula $\sum x_i^2$ and $n\bar{x}^2$ are merely intermediate quantities, not statistically useful quantities in their own right.

Let $\tilde{f}(x)$ be the approximation to $f(x)$ which arises from an algorithm. The algorithm is called *backwardly stable* if $\tilde{f}(x)$ can be shown to be equal to the exact evaluation of f at $\tilde{x}$, where $\tilde{x}$ is close to x. In this way, a (backwardly) stable algorithm will compute a well-conditioned function accurately, and will compute an ill-conditioned function as accurately as allowed by its conditioning.

Although proving error bounds is an important part of modern numerical analysis, the specific bounds obtained are usually pessimistic and are seldom used in practice. In general, rounding-error analyses are less valued for their final bounds than for the insight they provide about a numerical algorithm. A thorough treatment of rounding-error analyses can be found in Higham [6].

Floating Point Arithmetic

There are an infinite number of real numbers, but only a finite number can be represented on a computer. There-in lies the fundamental difference between exact and computer arithmetic, alluded to above. Numbers are represented on computers in floating point form, that is $f \cdot \beta^e$ in terms of a base β, fraction f and exponent e. For example

$$2.597 \cdot 10^{-3}$$

is a base-ten floating-point number with four figures of accuracy. Most computers use base 2, and the resulting arithmetic is called binary arithmetic.

Finite computer arithmetic produces three types of errors. When an arithmetic operation which produces a number with an exponent that is too large, the result is said to have *overflowed.* Similarly an arithmetic operation that produces an exponent that is too small is said to have *underflowed.* Even within the limits of the exponent, most numbers cannot be represented exactly on floating-point arithmetic of a fixed word length. The resulting inaccuracy is called *rounding error.* It is a central concern of numerical analysis that rounding errors not accumulate during a long computation. See FLOATING-POINT ARITHMETIC.

Linear Equations and Matrix Computations

The theory and practice of solving a linear system

$$A\mathbf{x} = \mathbf{b}$$

for $\mathbf{x}$, and more generally the whole subject of computations involving matrices, is now very well developed. Here we outline two applications of interest to biostatisticians.

In least squares regression of a response vector $\mathbf{y}$ on a design matrix X, numerical analysts have influenced statisticians to move away from the normal equations for the regression coefficients in favour of methods based on the decomposition

$$X = QR$$

where Q is an orthogonal matrix and R is upper triangular. This is because the QR approach is backwardly stable, while the normal equations are not.

Conditioning for the least squares problem is determined by that of X, which can be analyzed through the singular value decomposition

$$X = UDV^T$$

where U and V are orthogonal and D is diagonal containing the singular values. The condition number of X is usually defined to be ratio of the largest to the smallest singular value. If the columns of X are standardized, say by dividing by the sample standard deviation of the column, then the singular values entirely capture the idea of ill-conditioning and collinearity for the least squares problem. The singular value decomposition therefore gives statisticians the means to quantify collinearity, and there are those who propose its routine use in regression computations for that reason [9, Section 15.4].

Numerical linear algebra is dealt with in more detail in the article MATRIX COMPUTATIONS.

Optimization and Nonlinear Equations

Optimization means to find that value of $\mathbf{x}$ which maximizes or minimizes a given function $f(\mathbf{x})$. This is a central concern in statistics, because statistical estimation principles such as least squares, maximum likelihood, posterior mode and M-estimation are defined in terms of optimizing an appropriate objective function. Numerical optimization strategies come into play when the statistical model is nonlinear and analytic estimators of the parameters are not available.

A closely related problem is that of solving nonlinear equations. Many algorithms for optimizing $f(\mathbf{x})$ are in fact derived from algorithms for solving $\partial f/\partial \mathbf{x} = 0$, where $\partial f/\partial \mathbf{x}$ is the derivative vector of f with respect to $\mathbf{x}$.

Details are given in OPTIMIZATION AND NONLINEAR EQUATIONS.

Interpolation and Approximation

The purpose here is to accurately approximate complex functions with ones which are easy to evaluate. For example, rational function approximations to the standard normal distribution function and its inverse allow it to be computed rapidly within statistical programs. Typical methods include series expansions, rational functions and polynomials. See for example Press et al [9, Chapter 5] and POLYNOMIAL APPROXIMATION. A great many approximation formulae are given in Abramowitz and Stegun [1].

Numerical Integration

After matrix computations, numerical integration is one of the largest areas of numerical analysis. A large number of sophisticated and reliable methods are available for numerical integration in one dimension. Unfortunately for statisticians wanting to evaluate mixture models or Bayesian marginal posteriors, the picture is less clear in high dimensions. Statisticians have made a substantial contribution to high dimensional integration through the development of efficient Monte Carlo methods. A survey of integration methods is given in NUMERICAL INTEGRATION.

Available Software

The final goal of numerical analysis is to make numerical methods generally available through high quality portable software. Numerical analysts were also early users of the Internet, and a wide range of software is available online. Netlib is the most extensive collection of numerical programs. Its URL is

```
http://www.netlib.org
```

Worthy of special mention are the LINPACK library [5] for linear algebra and the EISPACK library [10] for eigenvalue computations, both from the Argonne National Laboratory. These are published, documented and freely available, and have gained wide acceptance by statisticians and other scientists. The two libraries have now been combined and updated as LAPACK [2]. Other libraries of note include the QUADPACK library [8] for numerical integration, and the SLATEC library, an enormous library of fortran programs.

The Guide to Available Mathematical Software (GAMS) at

`http://gams.nist.gov`

provides a virtual database of documented and supported programs, searchable by program and problem type. The journal ACM Transactions of Mathematical Software is a source of refereed software, also searchable by GAMS.

Commercial subroutine libraries include NAG Library (Numerical Algorithms Group) and the IMSL Mathematics and Statistics Libraries. The LINPACK, EISPACK and other routines have also been incorporated into the interactive matrix programming language, MATLAB [7].

Another popular commercial source is Numerical Recipes [9], accessible through

`http://nr.harvard.edu/nr`

Numerical recipes supplies smaller, understandable programs, which may be modified by users for specific applications. Netlib, GAMS, NAG and IMSL provide more sophisticated routines designed for high performance on large problems. Considerable effort has been expended to make the high performance routines efficient, memory compact and capable of trapping most errors.

Programs developed by statisticians, dealing specifically with statistical problems, can be found at Statlib, the statistical database maintained at Carnegie-Mellon University. The URL is

`http://lib.stat.cmu.edu`.

References

[1] Abramowitz, M., and Stegun, I. A. (1962). *Handbook of mathematical functions.* National Bureau of Standards, Washington. Reprinted by Dover, New York, 1965.

[2] Anderson, E., Bai, Z., Bischof, C., Demmel, J., Dongarra, J., DuCroz, J., Greenbaum, A., Hammerling, S., McKenney, A., Ostrouchov, S., and Sorensen, D. (1995). LAPACK *Users' Guide, Release 2.0, 2nd ed.*, SIAM Publications, Philadelphia, PA.

[3] Atkinson, K. E. (1989). *An Introduction to Numerical Analysis. 2nd Edition.* John Wiley, New York.

[4] Chan, T., Golub, G., and Leveque, R. (1983). Algorithms for computing the sample variance: analysis and recommendations. *The American Statistician*, **37**, 242–247.

[5] Dongarra, J. J., et al. (1979). *LINPACK's User's Guide.* SIAM, Philadelphia.

[6] Higham, N. J. (1996). *Accuracy and Stability of Numerical Algorithms.* Society for Industrial and Applied Mathematics, Philadelphia, PA. (A thorough and up-to-date treatment of the effects of rounding errors on numerical algorithms, especially on numerical linear algebra.)

[7] Moler, C., Little, J., and Bangert, S. (1987). *Pro-Matlab User's Guide.* The Math Works, Sherborn, MA.

[8] Piessens, R., De Doncker-Kapenga, E., Überhuber, C. W., and Kahaner, D. K. (1983). *Quadpack, a Subroutine Package for Automatic Integration.* Springer-Verlag, Berlin.

[9] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992). *Numerical recipes in Fortran.* Cambridge University Press. (Also available for C, Basic and Pascal.)

[10] Smith, B. T., Boyle, J. M., Ikebe, Y., Klema, V. C., Moler, C. B. (1970). *Matrix Eigensystem Routines:* EISPACK *Guide, 2nd ed.* Lecture Notes in Computer Science, Volume 6, Springer-Verlag, New York.

[11] Stewart, G. W. (1996). *Afternotes on Numerical Analysis.* Society for Industrial and Applied Mathematics, Philadelphia, PA. (An elegant and elementary introduction to the central ideas of numerical analysis.)

[12] Stoer, J., and Bulirsch, R. (1993). *Introduction to Numerical Analysis, Second Edition.* Springer-Verlag, New York.

[13] Thisted, R. A. (1988). *Elements of Statistical Computing. Numeric Computation.* Chapman and Hall, New York.

Related Articles

Algorithms
Computer Algebra
Floating Point Arithmetic
Matrix Computations
Numerical Integration
Optimization and Nonlinear Equations
Polynomial Approximation