When absolute approximations are optimal in function subroutines

C. B. Dunham

Department of Computer Science, The University of Western Ontario, London 72, Ontario, Canada

We show that in a very common case best absolute approximations are optimal in subroutines for mathematical functions.

(Received September 1972)

Let f be a mathematical function for which we wish to write a subroutine and suppose this can be done by approximating a (possibly different) continuous function g on an interval $[a, b]$ such that $1/\beta \leq g \leq 1$, β the base in floating point. We will first consider the case in which $g < 1$ and then get rid of this restriction later. The best subroutine possible is one in which all answers agree to $\frac{1}{2}$ unit in the last place with the true answer. Since all exponents of the true answer are zero, we have absolute error in this case no more than $\frac{1}{2}\beta^{-t}$, t the number of digits. We are thus led to choose an approximation h such that $|g - h| \leq \alpha\beta^{-t}$, α a chosen number normally a little less than $\frac{1}{2}$. It is not difficult to show analytically that if g takes the value 1 at only a finite number of points, h is to be chosen similarly. We could instead choose h according to a relative error criterion, $|g - h| \leq \alpha\beta^{-t}g$. This is a stronger condition which in marginal cases could require a higher degree approximation. In any case there is no reason to suppose that having absolute error smaller where g is smaller will pay off if h is within machine accuracy. A similar analysis shows that absolute approximations are optimal where $1 \leq g \leq \beta$ and g attains β only finitely many times.

If we examine Hart (1968), we find that most of the elementary functions are most conveniently computed by reduction to such functions g. In particular, if the function we are interested in has a zero of order 1 at the point zero, we approximate the function with the zero divided out. For example, sine and cosine are usually obtained from a joint subroutine using approximations to $\cos(x)$ and $\sin(x)/x$ for $x \in [-\pi/3, \pi/3]$ or a smaller interval.

It is interesting to note that the earliest approximations were absolute approximations and that it has been uniformly considered that relative approximations are better.

Reference

HART, J. F. *et al.* (1968). *Computer Approximations*, New York: Wiley.

Book review

FORTRAN Techniques, by A. Colin Day, 1972; 96 pages. (*Cambridge University Press*, £2·50, hard cover; £1·10 soft cover)

The reviewer agrees in general with the author's assertion that the book fills a need in the computer applications literature to acquaint users with some of the more sophisticated techniques programmable in FORTRAN. The language used to formalise the coding technique in Dr Day's book is American Standard FORTRAN as defined by document X3.9-1966 of the American National Standards Institute which is a subset of what is generally referred to as FORTRAN IV.

This choice has both advantages and disadvantages. As a subset of FORTRAN IV, ASA FORTRAN enjoys a high degree of machine independence which is reflected in the examples given. On the other hand, an extremely readable description in the English language is unavoidably but continuously interrupted by statements in the FORTRAN language which are not ordinarily used for purposes of communication.

Caution must be used when applying some of the techniques since they can produce an exceedingly inefficient object code. This sacrifice is, of course, necessary if FORTRAN is the only high-level language available or unavoidable if other such languages produce just as inefficient code. However, if the decision to use FORTRAN for a particular problem has been dictated by considerations which far outweigh any disadvantages incurred by inefficient object code in a few of the subroutines, these techniques greatly increase the power and scope of the FORTRAN language. Furthermore, they are especially useful for interface list processing procedures with the extensive mathematical techniques available as FORTRAN programs or subroutines thus allowing problem orientated languages to be developed.

In the reviewer's opinion this book should be used as an ancillary text to advanced sections of FORTRAN programming courses and if this indeed becomes the case the book should have a reasonably wide appeal. At £2·50 in hard back it is certainly expensive, especially for students, who should have their own personal copies for continuous reference rather than having to rely on those in the library, but there is also a softback edition at a price that students can afford so that they may become acquainted with the more advanced aspects of FORTRAN programming early in their careers.

E. M. CHANCE (London)