The effect of inadequate convergence criteria in automatic routines

By J. N. Lyness*

The possibility of round off or other statistical error in function values is sometimes not taken into account in the construction of automatic routines. The possible consequences of such an omission are discussed.

(Received February 1969)

'The aim of an automatic integration scheme is to relieve the person who has to compute an integral-[the user] of any need to think.' This is the opening sentence of Chapter 6, 'Automatic Integration' of the standard textbook *Numerical Integration*, by P. Davis and P. Rabinowitz (1967).

Informal discussion indicates that views about this aim vary from extreme to extreme. Some say the user should be made to think; others, that it is impossible to relieve him of all need to think and 'consequently' no attempt should be made to relieve him of any need to think. Another view is that this should be the aim of all numerical analysis or at least the aim of any published algorithm. Occasionally the same individual subscribes to all of these views. However, the present author accepts the opening sentence as a laudable aim, and discusses below the presently available implementations of this aim.

The textbook goes on to describe an automatic integration scheme in general terms (as it appears to the user). The user is to provide the upper and lower limits B and A, a tolerance EP, a subroutine FUN(X) and N, an upper limit on the number of function evaluations to be used. After using N function evaluations the routine gives up and provides the best result available and some indication or message to this effect.

Although in the text it is suggested that N be provided by the user, in the examples in the same book, N is usually provided by the program. In the Romberg Integration Code $N = 2^{15} + 1 = 32{,}769$, and in the Adaptive Simpson Code $N = 2\cdot 3^{30} + 1 \simeq 4 \times 10^{14}$. Presumably experience has shown that it is difficult enough to get the average user to think out what value EP he requires. (This is the author's experience.) When asked in addition for a value of N, mental exhaustion leads him to make some wild guess.

Thus in the present state of the art, the aim (which is to relieve the user of any need to think) has been attained to the following extent. He does not have to think (at least not at this stage) about A, B, or FUN(X). He is forced to give some thought to EP. He has refused to think about N and the programmer has usually done it for him.

An automatic quadrature routine may be considered as containing two essential ingredients. One is the standard rule evaluation part, which is capable of obtaining numerical approximations in terms of function values. The other is the 'strategy component' using which the routine considers the results it has already obtained and decides what further calculation is necessary to obtain a result of the desired accuracy.

An example is the automatic routine ROMBERG (A, B, EP, FUN). The rule evaluation part is capable of calculating successive approximations $R_1f, R_2f, \ldots, R_{15}f$ to the required integral. Here R_jf is the standard Romberg approximation requiring $2^j + 1$ function values (Bauer, Rutishauser, and Stiefel, 1963). The strategy section is particularly simple. After each evaluation R_jf, $2 \geqslant j \geqslant 14$, the number

$$| R_jf - R_{j-1}f | - EP$$

is calculated. If this number is less than or equal to zero the routine returns R_jf as a final answer and the calculation terminates. Otherwise the routine proceeds to calculate the next approximation $R_{j+1}f$. If $j = 15$, the routine need not carry out this check. It simply returns the result $R_{15}f$ a result requiring 32,769 function evaluations.

In the Adaptive Simpson automatic quadrature routine, the procedure is more complicated. However, it is still possible to distinguish between sections of coding devoted to rule evaluation and sections devoted to strategy.

We now turn to the question of the effect of round-off error in these routines. While of course every single operation in the routine could be considered as a source of round-off error, the principal sources are generally agreed to be these:

(i) Errors in the abcissas x_i and weights w_i;
(ii) Errors in function values $f(x_i)$;
(iii) Accumulation Error in the sum $\Sigma w_i f(x_i)$.

In general the effect of errors (i) and (iii) can be removed by calculating abscissas and weights and carrying out the sum using double precision arithmetic. The relative additional cost in computer time is not significant. However, errors in the function values are beyond the reach of the automatic routine.

From the point of view of the rule evaluation, such errors while present are not serious. For example, if a function is only accurate to seven digits the effect of using a rule which is capable of obtaining a twelve digit result is that a seven digit answer is obtained at a twelve digit answer cost. Possibly a factor of 2 or 3 in machine time might have been wasted. This is one reason why round-off error is not considered a practical problem in numerical quadrature.

However, from the point of view of strategy in an automatic routine, moderate errors of a statistical nature in the function values $f(x_i)$ may cause a disaster. Two examples are illustrated in the accompanying tables.

* *Argonne National Laboratory, Argonne, Illinois 60439*
This work was performed under auspices of the U.S. Atomic Energy Commission.

In each the same problem, the evaluation of

$$\int_0^{3\pi/2} \cos x \, dx = -1$$

is attempted to various accuracies *EP*, using essentially the coding

$$\text{FUN}(X) = \text{COS}(X) + 1000000{\cdot}0 - 1000000{\cdot}0$$

(recoded in such a way that the compiler does not notice the cancellation).

In the first example, **Table 1**, using ROMBERG (*A*, *B*, *EP*, FUN) it is possible to see at a glance how the strategy reacted to the information at its disposal; the values $|R_j f - R_{j-1} f|$ are listed. If $EP = 10^{-6}$ the routine concludes after stage $j = 6$ returning a six-figure result based on 65 function evaluations. If $EP = 10^{-7}$, the routine goes on to stage $j = 15$ returning a result of comparable accuracy based on 32,769 function evaluations. The price paid by the user for not giving sufficient thought to the value of *EP* is a factor 500 in machine time, with no significant gain in accuracy.

Table 1

Romberg integration approximations

j	N	$R_j f$	$R_j f - R_{j-1} f$
1	3	−0·4879856539	
2	5	−1·0127508941	−5·25–001
3	9	−0·9998910218	1·29–002
4	17	−0·9999900407	−9·90–005
5	33	−0·9999943948	−4·35–006
6	65	−0·9999951011	−7·06–007
7	129	−1·0000017132	−6·61–006
8	257	−1·0000003520	1·36–006
9	513	−1·0000008370	−4·85–007
10	1,025	−0·9999991700	1·67–006
11	2,049	−0·9999993830	−2·13–007
12	4,097	−1·0000001393	−7·56–007
13	8,193	−0·9999999992	1·40–007
14	16,385	−1·0000001818	−1·83–007
15	32,769	−0·9999997753	4·07–007

In the second example, **Table 2**, using a modification (Lyness, 1969) of the original version (McKeeman, 1962) of the adaptive Simpson Routine, the consequence of asking for too much accuracy is even more catastrophic, a factor of 1,000 in machine time. In the table, only results are given. Here if $EP \leqslant 10^{-7}$ round-off error prevents convergence in any interval unless the difference to be tested is identically zero in that interval. Essentially a machine accuracy calculation is carried out.

Depending on one's point of view, this type of behaviour in these routines may be regarded as a fault, or as a necessary penalty designed to influence the user to give more thought to his problem. The fault, if it is so considered, lies not in the rule evaluation section, but in the strategy section. In general the numerical approximations are all quite adequate. The simple strategy in these routines assume that there will be no significant round-off error and proceed on that basis. If there is round-off error, the strategy sets the routine off on a totally unrealistic sequence of calculations.

Table 2

Some results given by routine
SQUARED (*A*, *B*, *EP*, FUN)

EP	*N*	ACTUAL ERROR*
10^{-4}	29	$-1{\cdot}1 \times 10^{-5}$
10^{-5}	45	$-6{\cdot}9 \times 10^{-6}$
10^{-6}	97	$3{\cdot}9 \times 10^{-6}$
10^{-7}	152,997	$-6{\cdot}6 \times 10^{-7}$
10^{-8}	152,997	$-6{\cdot}6 \times 10^{-7}$

* This is (−1·0-SQUARED) and not (EXACT INTEGRAL-SQUARED)

In this example the round off level was set deliberately to simulate a large cancellation error. Thus, if the problem had been re-coded to avoid cancellation error or if a computer using a longer word length had been used, different results would have been obtained. This is not the only way in which such errors arise. The error in $f(x_i)$ might be discretisation error arising from perhaps a previous numerical quadrature. If an automatic routine is used for the previous calculation, the discretisation error may have a different form for different values of x_i. Thus it would appear to the automatic quadrature routine that the function $f(x)$ it is asked to integrate is smooth only in sections; the errors here could be termed semi-statistical.

These errors may be relatively independent of the machine accuracy. They stem from a desire on the part of the user to economise at an earlier stage in the calculation. They are not due to round off nor are they due to cancellation. Nevertheless they are present and may cause the unwary routine to have difficulty in converging.

In general the function $f(x)$ may be an intricate combination of solutions of differential equations, spline functions and even extrapolated experimental data. The user may be unable or unwilling to perform an extensive error analysis of the entire problem to determine the accuracy of his function. Even if he were, to ask him to do so is completely at variance with the stated aims of automatic integration, namely that he should not have to think at all. Instead of allowing him not to think, these automatic routines demand that either he gives considerable thought to the problem of determining *EP*, or that he risks being penalised by computer time factors of order 500 or 1,000.

The aims of automatic integration may be more completely fulfilled if the strategy is adjusted to take into account possible statistical error in the function values. The intention should be that the routine, which has available large numbers of function values $f(x_i)$, should in some way use this information to determine the round-off level—or 'noise'. The value of *EP* provided by the user should be considered by the routine as a lower bound only, and should be increased to the noise level automatically if this becomes necessary. A user who simply requires the best result available sets $EP = 0{\cdot}0$. At the end of the calculation *EP* is over-written by a number calculated by the routine which represents its own estimate of the accuracy attained.

The detailed method by which such a scheme should be put into effect varies from routine to routine. The author has implemented such a scheme in a modification of the Adaptive Simpson routine described in detail in Lyness (1969). This particular routine calculates in any case various quantities which are very susceptible to round-off error whose difference should generally be positive; in practice this difference is used as an indicator of the round-off level with reasonable success.

In principle a scheme of this nature can be implemented by applying familiar theory concerned with the determination of the noise in a table of numbers. A chapter is devoted to this in Hamming (1962). Or the criterion could be simply that the routine terminates when successive iterates do not appear to be converging.

The author does not feel that there is any fundamental difficulty in arranging for an automatic routine to take into account the possibility of statistical fluctuations in $f(x_i)$ and to terminate if appropriate. The principal difficulty—and the main purpose of this article—is to convince the people who construct such routines that it is a necessary or worthwhile thing to do. Once people are so convinced, they will be quite ingenious enough to invent many schemes and to make thorough tests of these with a view to producing the most efficient procedure for particular routines.

The discussion in this article is set entirely in the context of Automatic Quadrature Routines. This is an appropriate context. There is a general feeling that round-off error in quadrature does not matter because these errors do not become amplified but tend to eliminate each other. The point of this article that in any automatic routine they may be critical is emphasised by this context. The numerical stability of the calculations is a separate question, which is of much less importance in quadrature than in differentiation or the solution of differential equations. Thus the foregoing discussion applies to some extent to any Automatic routine which requires a user-provided function subroutine.

Perhaps it should be emphasised that this sort of procedure does not 'eliminate the round-off error'. The final *result* is as good—or bad—as the given function, the set of quadrature rules and the particular strategy allow. All that this modification attempts to do is to prevent round-off error (of a statistical nature) affecting the strategy section in such a way that an extensive sequence of unnecessary calculations is embarked upon which do not improve the result, but which may use up an excessive amount of computer time.

References

BAUER, F. L., RUTISHAUSER, H., and STIEFEL, E. (1963). New aspects in numerical quadrature, *Proc. Symposia Appl. Math* Vol. 15, pp. 199–218.

DAVIS, P., and RABINOWITZ, P. (1967). *Numerical Integration*, London: Blaisdell Publishing Co.

HAMMING, R. W. (1962). *Numerical Methods for Scientists and Engineers*, New York: McGraw-Hill Co., Inc.

LYNESS, J. N. (1969). Notes on the Adaptive Simpson Quadrature Routine, *JACM*, Vol. 16, No. 3.

MCKEEMAN, W. M. (1962). Algorithm 145, adaptive numerical integration by Simpson's Rule, *Comm. ACM*, Vol. 5, p. 604.