An iterative method for quadratures

By Henry C. Thacher, Jr.

An iterative algorithm is presented for estimating repeated integrals of a function,

$$f^{[n]}(x_0) = \int_0^{x_0} \int_0^{x_1} \cdots \int_0^{x_{n-1}} f(x_n) dx_n \cdots dx_1.$$

The estimate is based upon numerical values of the function at a set of arbitrarily-spaced base points, and is equivalent, except for roundoff, to the repeated integral of the interpolating polynomial.

The convenience of iterative methods of interpolation such as Aitken's (1932) and Neville's (1934) techniques is well known. In addition to allowing introduction of additional data as required by earlier results, they make no requirements on the spacing of the data, and employ a simple, easily programmed algorithm. More recently, D. B. Hunter (1961) has shown that the process can be extended to numerical differentiation. His algorithm turns out to have all the advantages of the interpolation algorithms, but also to be far more flexible in the point of estimation, and in the order of the derivatives, than the standard finite-difference techniques for numerical differentiation. It is the purpose of this note to point out the possibility of applying Hunter's development to the quadrature problem.

Hunter's fundamental equation is

$$y^{(r)}_{i,j,\ldots,p,q} = \frac{1}{x_q - x_i}\left\{\begin{vmatrix} y^{(r)}_{i,j,\ldots,p} & x_i - x \\ y^{(r)}_{j,\ldots,p,q} & x_q - x \end{vmatrix} + r\left[y^{(r-1)}_{j,\ldots,p,q} - y^{(r-1)}_{i,j,\ldots,p}\right]\right\} \quad (1)$$

where $y_{i,j,\ldots,p,q}$ is the polynomial of lowest degree in x which takes on the values y_i at $x = x_i$, y_j at $x = x_j, \ldots,$ y_p at $x = x_p$ and y_q at $x = x_q$. $y^{(r)}_{i,j,\ldots,p,q}$ is the rth derivative of $y_{i,j,\ldots,p,q}$.

Take the integration operator $\int_0^x dx$ to be the inverse of the derivative operator, d/dx, and let $y^{[s]}_{i,j,\ldots,p,q}$ be the s-fold repeated integral of $y_{i,j,\ldots,p,q}$. Then the proof of (1), with $-s = r$, leads to

$$y^{[s]}_{i,\ldots,i+k} = \frac{1}{x_{i+k} - x_i}\left\{\begin{vmatrix} y^{[s]}_{i,\ldots,i+k-1} & x_i - x \\ y^{[s]}_{i+1,\ldots,i+k} & x_{i+k} - x \end{vmatrix} + s\begin{vmatrix} y^{[s+1]}_{i,\ldots,i+k-1} & 1 \\ y^{[s+1]}_{i+1,\ldots,i+k} & 1 \end{vmatrix}\right\} \quad (2)$$

For simplicity in notation, we have adopted Neville's ordering with interpolations based on adjacent sets of k points.

The formula (2) is used as it stands, which means that an iteration involving $n+1$ points for $y^{[s]}$ needs values not only for $y^{[s]}$ involving n points but also for $y^{[s+1]}$ involving n points. To start the iteration, therefore, we need the set of one-point values for $y^{[s]}_i$ with $s = m, m+1, \ldots, n+m-1$, where n is the total number of points to be used in the estimate, and m the order of the integral to be estimated. These starting values are based on zero-order approximations to $y^{[0]}_i$, namely the function values y_i each treated as an independent constant approximation to y. These yield the starting values $y^{[s]}_i = y_i x^s/s!$ for the estimate of $y^{[s]}_i$ corresponding to the upper limit x of the integral sought.

The amount of computation involved is greater and the convergence seems appreciably slower when compared with Hunter's process for finding derivatives.

To illustrate its behaviour, the algorithm was programmed for the Royal-Precision LGP-30 computer, using floating-point arithmetic with 8+ significant decimal digits (24.2). Partial results of a typical experiment are shown in Table 1. The truncation error is, of course, zero for $i \geqslant 5$.

In comparison with standard quadrature methods, this algorithm has several attractive features: (*a*) it may be applied to an arbitrarily spaced set of base points, and thus may be used, for example, in estimating the integrals of the inverse of a tabulated function; (*b*) it may be used to estimate the higher repeated integrals, for which few standard formulas are available; (*c*) it is readily programmed for a digital computer. On the other hand, as the data show, the algorithm may be numerically unstable for large s. Moreover, as the iteration is repeated, the round-off error in the higher integrals is propagated downward, and the estimates diverge. The method must, therefore, be used with caution, in spite of its advantages, particularly if the number of base points, and the order of integral to be estimated are large.

Downloaded from https://academic.oup.com/comjnl/article/5/3/228/424428 by guest on 19 April 2024

Table 1

Iterative quadrature of $y = x^5$, estimated at $x = 0{\cdot}9$

i	x_i	$y_{0,\ldots,i}$	$y_{1,\ldots,i+1}$	$10y^{[1]}_{0,\ldots,i}$	$10y^{[1]}_{1,\ldots,i+1}$	$100y^{[2]}_{0,\ldots,i}$	$100y^{[2]}_{1,\ldots,i+1}$
0	0·0000000	0·0000000	1·0000000	0·0000000	9·000000	0·0000000	40·500000
1	1·0000000	0·9000000	0·8062500	4·050000	−0·5906250	12·150000	−14·428125
2	0·5000000	0·7312500	0·7093750	1·012500	1·4800781	−0·3796875	3·717774
3	0·2500000	0·6525000	0·6037500	1·056797	0·6939844	1·427625	−0·5353596
4	0·7500000	0·5940000	0·5935125	0·8831530	0·8926116	1·102612	1·184511
5	0·1250000	0·5904900	0·5904900	0·8857356	0·8857350	1·138807	1·138789
6	0·3750000	0·5904900	0·5904900	0·8857388	0·8857285	1·138748	1·138812
7	0·6250000	0·5904900	0·5904900	0·8857136	0·8857510	1·138816	1·138868
8	0·8750000	0·5904900	—	0·8857461	—	1·138823	—
True values		0·5904900		0·8857350		1·138802	

Except for rounding, the Hunter algorithm and the present algorithm give, respectively, the derivatives, and the repeated integrals of the unique interpolation polynomial based on the given data. An expression for the discretization error, due to replacing the actual function by a polynomial of finite degree may, therefore, be obtained by appropriate differentiation or integration of the remainder of the Lagrangian interpolation polynomial.

The rounding error may be considered in two parts, that due to errors in the initial data, and that due to rounding in carrying through the procedure. The former is roughly proportional to the sum of the magnitudes of the weights associated with the various data. Although these weights are not normally generated in an iterative procedure of this sort, they may be secured by applying the iteration to the set of functions $f_i(x_j) = \delta_{ij}$. In our example, as might be expected, the sensitivity to inaccuracies in the input data decreases with increasing order of integration.

A more serious source of error is rounding during the iteration. Unfortunately, the algorithm tends to amplify round-off errors, particularly for large values of s. Evidence of this instability can be seen in Table 1. Although the interpolated value itself becomes stabilized at the true value for $i \geqslant 5$, the estimates for the first and second repeated integrals are closest to the true value for $i = 5$, and thereafter become successively poorer.

We may analyse this instability under the assumption that the iteration has been carried far enough so that the discretization error is negligible, but that random residual errors, $\epsilon^{[s]}_{i,\ldots,i+k-1}$, $\epsilon^{[s]}_{i+1,\ldots,k}$, and so on, are associated with each of the estimates. Then, by substitution in (2) we find

$$\epsilon^{[s]}_{i,\ldots,i+k} = \left(\frac{x_{i+k} - x}{x_{i+k} - x_i}\right)\epsilon^{[s]}_{i,\ldots,i+k-1} + \left(\frac{x - x_i}{x_{i+k} - x_i}\right)\epsilon^{[s]}_{i+1,\ldots,i+k} + \frac{s}{(x_{i+k} - x_i)}(\epsilon^{[s+1]}_{i,\ldots,i+k-1} - \epsilon^{[s+1]}_{i+1,\ldots,i+k}). \quad (3)$$

Thus, if the errors for the k-point results are less than ϵ in magnitude,

$$|\epsilon^{[s]}_{i,\ldots,i+k}| \leqslant \frac{|x_{i+k} - x| + |x_i - x| + 2s}{|x_{i+k} - x_i|}\epsilon. \quad (4)$$

The possibilities of error amplification for large values of s are obvious, while the likelihood of amplification for $s = 0$ is also seen to be low under normal conditions of application.

References

AITKEN, A. C. (1932). "On Interpolation by Iteration of Proportional Parts, Without the Use of Differences," *Proc. Edinburgh Math. Soc.*, Series 2, Vol. 3, p. 56.

HUNTER, D. B. (1961). "An Iterative Method of Numerical Differentiation," *The Computer Journal*, Vol. 3, p. 270.

NEVILLE, E. H. (1934). "Iterative Interpolation," *J. Indian Math. Soc.*, Vol. 20, p. 87.

Downloaded from https://academic.oup.com/comjnl/article/5/3/228/424428 by guest on 19 April 2024