UDC 551.509.313

AN *N*-CYCLE TIME-DIFFERENCING SCHEME FOR STEPWISE NUMERICAL INTEGRATION

EDWARD N. LORENZ

Massachusetts Institute of Technology, Cambridge, Mass.

ABSTRACT

A time-differencing scheme consisting of an initializing step and N repetitions of a set of steps is proposed. For linear equations, the scheme is of Nth order. It is easily programmed and uses a minimal amount of storage space. The order may be changed by changing one parameter. An improved scheme is of Nth order even for nonlinear equations, for $N \leq 4$.

1. THE BASIC *N*-CYCLE SCHEME

There are many physical problems in which one may wish to solve a large number of simultaneous ordinary differential equations by numerical means. Among these problems are some where the single independent variable represents time and where one seeks future values of the dependent variables, given the present values.

An obvious example is the many-body problem, but the situation also arises when one is interested in solving a smaller number of partial differential equations where the independent variables represent time and one or more space coordinates. As a first step in the solution, one may replace each function of time and space by a set of functions of time alone. Frequently, these functions are simply the values of the original variables at a set of preselected locations in space or the coefficients in the expansions of the variables in a series of preselected orthogonal functions. When the spatial configuration of the original variables possesses considerable detail that is thought to be relevant, the required number of new variables may be large indeed. This is true, for example, in the case of numerical weather prediction, or numerical simulation of the atmosphere, where three-dimensional grids of 10,000 or more points are not uncommon.

The use of numerical procedures generally demands also that the values of the variables be determined only at a discrete set of times. In some integration schemes used in numerical weather prediction, the values of the variables at certain points are determined at certain times, while those at other points are determined at other times. In other schemes, the values of all the variables are determined at each time. It is the latter type of scheme that concerns us here. Under such a scheme, the choice of a procedure for replacing space derivatives by finite differences effectively replaces each partial differential equation by a set of ordinary differential equations. The choice of a procedure for advancing forward in time then becomes a separate problem.

Consider then the system of M ordinary differential equations

$$dx_i/dt = F_i(x), \qquad i=1, \ldots, M, \tag{1}$$

where by the argument x we mean the multiple argument $x_1, \ldots, x_M$. Let the values of x_i be given at some time t_0. We desire a procedure for approximating the values of x_i at time $t_0 + \Delta t$ where Δt is a reasonably small time increment.

There are numerous well-known methods for accomplishing this end. The simplest, but also one of the less accurate, is the uncentered-difference approximation

$$x_i(t_0+\Delta t) \sim x_i(t_0) + F_i(x(t_0))\Delta t. \tag{2}$$

Among those schemes affording better approximations are the higher order Runge-Kutta schemes (e.g., Henrici 1962). In the Nth-order scheme, the approximation for $x_i(t_0 + \Delta t)$ when expanded in a power series in Δt agrees with the Taylor series expansion for the exact value of $x_i(t_0 + \Delta t)$ through terms in Δt^N. The uncentered-difference scheme, incidentally, is equivalent to a first-order Runge-Kutta scheme.

For the most general set of equations of the form of eq (1), the use of the uncentered-difference scheme requires that at least $2M$ numbers be stored simultaneously in the memory of the computer at some stage of the computation; this is so because all M time derivatives F_i must be evaluated before any of the variables x_i can be modified, since otherwise the wrong values of some variables would be used in evaluating the remaining derivatives. In the straightforward application of some higher order schemes, considerably more than $2M$ numbers may have to be stored simultaneously; and if M is large, the procedure may tax the capacity of the computer. The procedure that we shall offer is the outcome of an attempt to use a higher order scheme with a minimum use of storage space (specifically, without having to store any more numbers than would have to be stored in using the uncentered-difference scheme). Actually, our procedure will prove to have other advantages including simplicity in programming, which will render it attractive even when M is small and storage space is not an important consideration.

Our N-cycle scheme uses $2N$ constants $c_0, \ldots, c_{2N-1}$. In addition to the registers needed to store these constants and the space needed to evaluate but not store the functions F_i, the scheme uses $2M$ registers in which the contents will be denoted by $y_1, \ldots, y_M$ and $z_1, \ldots, z_M$, and an additional register in which the contents will be denoted by k.

Suppose that, initially, the values of y_i are $x_i(t_0)$ while those of z_i are arbitrary. The algorithm for making the values of y_i equal to $x_i(t_0+\Delta t)$ contains an initializing step (0), and six steps (1–6) performed N times:

0. Set $k=0$.
1. Multiply z_i by $c_{2k}/\Delta t$.
2. Add $F_i(y)$ to z_i.
3. Divide z_i by $c_{2k+1}/\Delta t$.
4. Add z_i to y_i.
5. Add 1 to k.
6. If $k<N$, return to step 1; if $k=N$, the procedure is completed. (Except for the simplest systems of equations, most of the computation occurs in step 2 when evaluating F_i.)

The problem we now face if we are to put the scheme to use is the choice of suitable constants $c_0, \ldots, c_{2N-1}$. The simplest way of insuring that the final result will be independent of the initial arbitrary values of z_i is to let $c_0=0$, and we shall not consider any other possible choice. The remainder of this work is concerned with the choice of the remaining constants.

2. THE SIMPLEST SCHEME

Consider first the case where the differential eq (1) are linear. They may then be written

$$d\mathbf{X}/dt=\mathbf{AX} \tag{3}$$

where $\mathbf{X}$ is a matrix of M rows and one column with elements x_i and $\mathbf{A}$ is a square matrix of order M in which the elements are constants. We may likewise let $\mathbf{Y}$ and $\mathbf{Z}$ denote matrices of M rows and one column with elements y_i and z_i. Because eq (3) is linear, the values of $\mathbf{Y}$ and $\mathbf{Z}$ after k repetitions of steps 1–6 in the algorithm will be polynomials of degree k in Δt. We wish to choose the constants c_j so that, after N repetitions,

$$\mathbf{Y}=\sum_{k=0}^{N}\mathbf{A}^k\mathbf{X}(t_0)\Delta t^k/k! \tag{4}$$

Carrying out the algorithm and equating coefficients of Δt^k, we find that the $2N-1$ constants c_j must satisfy the N equations

$$\sum_j c_j=N, \tag{5a}$$

$$\sum_{j<k-1} c_jc_k=N(N-1), \tag{5b}$$

$$\sum_{j<k-1<l-2} c_jc_kc_l=N(N-1)(N-2), \tag{5c}$$

$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots,$$

$$c_1c_3\ldots c_{2N-1}=N! \tag{5N}$$

The summation in the kth equation in eq (5) is over all those products of k distinct constants in which no two successive constants enter as factors.

Since there are more unknowns than equations, we may expect a multiplicity of solutions. Hence, we may seek to extend the usefulness of the scheme by choosing solutions that possess further desirable properties. We therefore let $\delta t=\Delta t/N$ and seek values of the constants for which the value of $\mathbf{Y}$ after k repetitions of steps 1–6 will afford a fair approximation to $\mathbf{X}(t_0+k\delta t)$. Specifically, we demand that the linear term in Δt be $kN^{-1}\mathbf{AX}\Delta t$. We find that this will be so if

$$c_{2k+1}=N+c_{2k}. \tag{6}$$

In particular, $c_1=N$.

We now have more equations than constants so that conceivably there could be no solution, but actually a simple solution is

$$c_{2k}=-k \tag{7a}$$

and

$$c_{2k+1}=N-k. \tag{7b}$$

Although it is not immediately obvious, at least to us, that eq (7a) and (7b) actually satisfy eq (5), it is readily verified by induction that, after k repetitions of steps 1–6

$$\mathbf{Z}=\sum_{l=1}^{k}\frac{(k-1)!\,(N-l)!}{(l-1)!\,(k-l)!\,N!}\mathbf{A}^l\mathbf{X}\Delta t^l \tag{8}$$

and

$$\mathbf{Y}=\sum_{l=0}^{k}\frac{k!\,(N-l)!}{l!\,(k-l)!\,N!}\mathbf{A}^l\mathbf{X}\Delta t^l \tag{9}$$

whereupon eq (4) follows when $k=N$.

We thus obtain a very simple scheme. The constants c_j need not be determined in advance. It is sufficient to reserve two registers in which the contents will be denoted by a and b and to store the values of $1/\Delta t$ and $1/\delta t$. The algorithm may then be restated as:

0. Let $a=0$, $b=1/\delta t$.
1. Replace z_i with az_i.
2. Replace z_i with $z_i+F_i(y)$.
3. Replace z_i with z_i/b.
4. Replace y_i with y_i+z_i.
5. Replace a with $a-(1/\Delta t)$ and b with $b-(1/\Delta t)$.
6. If $b>0$, return to step 1; if $b=0$, the procedure is completed.

Since the constants c_j need not be stored, all that is required to change the order of the scheme is a change in $1/\delta t$. The one-cycle scheme, incidentally, is nothing more than the uncentered-difference approximation, and the two-cycle scheme is identical to the so-called "modified Euler method" (Henrici 1962).

3. AN ALTERNATIVE INTERPRETATION OF THE *N*-CYCLE SCHEME

We have introduced Δt as the fundamental time increment and have presented the preceding algorithm as an N-cycle scheme for advancing from t_0 to $t_0+\Delta t$. Because the value of y_i after k cycles is an approximation to $x_i(t_0+k\delta t)$, it is also possible to look upon δt as the fundamental time increment. The algorithm then becomes effectively a one-cycle scheme for advancing from $t_0+k\delta t$ to $t_0+(k+1)\delta t$, which varies from step to step and gives its most satisfactory approximations every Nth step.

There are certain advantages to this interpretation. First, the amount of computation required to advance from t_0 to $t_0+\Delta t$ is proportional to N, but that required to advance from t_0 to $t_0+\delta t$ is independent of N. With δt as the time step, the total amount of computation is determined by the number of time steps. One may then switch to a higher order scheme without altering the amount of computation simply by increasing Δt.

Perhaps more important, the maximum allowable value of Δt which will insure reasonable accuracy, and in particular will not introduce computational instability, tends to increase with increasing N; whereas the maximum allowable value of δt tends to be independent of N. To justify this claim, we restrict our attention to equations in which true solutions neither approach infinity nor settle down to a steady state as time increases. This happens, for example, in the problem of numerical weather forecasting.

Over a finite time interval, the solution may then be approximated by a superposition of periodic oscillations. It is generally recognized that, for a specified scheme, computational instability will result if Δt is too large a fraction of the shortest period of oscillation. Consider, for simplicity, a solution oscillating with the single period T; it may be governed by the single complex equation

$$dx/dt=2\pi iT^{-1}x, \tag{10}$$

although x will, of course, appear as two real variables in the computer. At time $t_0+\Delta t$, the power series expansion in Δt will be

$$x(t_0+\Delta t)=\sum_{k=0}^{\infty}(2\pi i/T)^k x(t_0)\Delta t^k/k! \tag{11}$$

If the N-cycle scheme is used to approximate $x(t_0+\Delta t)$, the absolute value of the leading term in the error will be

$$E_N=(2\pi\Delta t/T)^{N+1}x(t_0)/(N+1)! \tag{12}$$

Using Stirling's formula to approximate $(N+1)!$, we find that, in terms of δt,

$$E_N\sim(2\pi(N+1))^{-1/2}(N/N+1)^{N+1}(2\pi e\delta t/T)^{N+1}x(t_0). \tag{13}$$

Noting that the factor $(N/N+1)^{N+1}$ approaches $1/e$ as $N\rightarrow\infty$, we find that

$$\lim_{N\rightarrow\infty}E_N=\begin{cases}0 \text{ if } \delta t\leq T/2\pi e\\ \infty \text{ if } \delta t>T/2\pi e.\end{cases} \tag{14}$$

Unless δt is at least comparable to $T/2\pi$ and thus much larger than $T/2\pi e$, the leading term in the error will be the dominating term. It follows that, if δt is less than about 1/17 of the period of oscillation, the N-cycle scheme becomes more and more accurate as N is increased. However, if δt is greater than this amount, the scheme becomes more and more inaccurate.

For a more general system, the scheme becomes increasingly accurate with increasing N if δt is less than 1/17 of the shortest period of oscillation. This conclusion is most readily justified when the system is linear.

A basic time step δt may therefore be chosen independently of any considerations of N. Once it has been established that the value of δt is suitable, further accuracy may be gained without affecting the total amount of computation by increasing Δt.

In actual practice, the maximum allowable value of N may be limited by the computer's own round-off error, which has not been considered in the preceding development. When N is very large, the round-off error introduced at certain stages of the N-cycle scheme may be amplified manyfold during subsequent stages; and the results may be rendered unacceptable. In the case of eq (10), using fixed-point arithmetic, we have obtained excellent results with $N=16$ and worthless results with $N=32$.

4. AN IMPROVED SCHEME

When the differential eq (1) are nonlinear, the N-cycle scheme proves not to be of Nth order, for $N>2$. That is, the approximation for $x_i\ (t_0+\Delta t)$, when expanded in a power series in Δt, fails to agree with the Taylor series expansion

$$x_i(t_0+\Delta t)=\sum_{k=0}^{\infty}(d^kx_i(t_0)/dt^k)\Delta t^k/k! \tag{15}$$

through terms in Δt^N.

To examine the discrepancy, we note first that

$$dx_i/dt=F, \tag{16a}$$

$$d^2x_i/dt^2=F_{i,j}F_j, \tag{16b}$$

$$d^3x_i/dt^3=F_{i,j,k}F_jF_k+F_{i,j}F_{j,k}F_k, \tag{16c}$$

$$d^4x_i/dt^4=F_{i,j,k,l}F_jF_kF_l+3F_{i,j,k}F_{j,l}F_kF_l+F_{i,j}F_{j,k,l}F_kF_l+F_{i,j}F_{j,k}F_{k,l}F_l, \tag{16d}$$

etc. Here, in the symbols $F_{i,j}$, $F_{i,j,k}$, etc., each subscript except the first denotes partial differentiation with respect to the variable having a similar subscript; and summation over repeated subscripts is understood. When the equations are linear, the functions $F_{i,j}$ are constants; and $F_{i,j,k}$, etc., vanish.

Upon carrying out the algorithm and comparing with eq (16a), (16b), etc., we find that the approximation given by the three-cycle scheme is

$$x_i(t_0+\Delta t)\sim x_i+\frac{dx_i}{dt}\,\Delta t+\frac{1}{2}\,\frac{d^2x_i}{dt^2}\,\Delta t^2 +\frac{1}{6}\left(\frac{d^3x_i}{dt^3}+\frac{1}{6}\,F_{i,j,k}F_jF_k\right)\Delta t^3+\ .\ .\ .\ . \tag{17}$$

Likewise, the approximation given by the four-cycle scheme is

$$\begin{aligned}x_i(t_0+\Delta t)\sim x_i&+\frac{dx_i}{dt}\,\Delta t+\frac{1}{2}\,\frac{d^2x_i}{dt^2}\,\Delta t^2\\ &+\frac{1}{6}\left(\frac{d^3x_i}{dt^3}+\frac{1}{8}\,F_{i,j,k}F_jF_k\right)\Delta t^3+\frac{1}{24}\left(\frac{d^4x_i}{dt^4}\right.\\ &+\frac{1}{4}\,F_{i,j,k,l}F_jF_kF_l+\frac{1}{2}\,F_{i,j,k}F_{j,l}F_kF_l\\ &\left.+\frac{1}{4}\,F_{i,j}F_{j,k,l}F_kF_l\right)\Delta t^4+\ .\ .\ .\ .\end{aligned} \tag{18}$$

It thus appears that the three- and four-cycle schemes are actually only of second order.

It must not be concluded on this account that the three- and four-cycle schemes are no better than the two-cycle scheme nor that they are necessarily greatly inferior to third-order and fourth-order Runge-Kutta schemes. One should not look upon the order of a scheme as the sole measure of its suitability. First of all, different schemes of the same order may differ greatly in accuracy. Having determined that a scheme is of Nth order, one also needs to know how greatly the terms in Δt^{N+1} differ from the corresponding terms in the Taylor series expansion for $x_i\ (t_0+\Delta t)$.

Moreover, it is not obvious that, among schemes of different order, the higher order scheme is always preferable, especially if the equations are nonlinear. Certainly, the higher order scheme is more accurate if Δt is small enough. However, in such applications as numerical weather prediction, one does not ordinarily try to make Δt very small but is more likely to use as large a Δt as possible without introducing computational instability. In this event, the terms in Δt^{N+1} could well contribute less than those in Δt^{N+2}, or higher powers, to the total departure of the approximation to $x_i(t_0+\Delta t)$ from the exact value.

Our experience with the three- and four-cycle schemes applied to rather simple nonlinear systems suggests that they are much more accurate than the two-cycle scheme. Nevertheless, for some purposes, it is desirable to have an N-cycle scheme that is known to be of Nth order. We, therefore, seek other solutions of eq (5).

Among those solutions that also satisfy eq (6) appears to be the solution

$$c_0=0,\ c_{2k}=-(N-k) \qquad \text{if } k>0 \tag{19a}$$

and

$$c_1=N,\ c_{2k+1}=k \qquad \text{if } k>0. \tag{19b}$$

Again, it is not obvious that eq (19a) and (19b) satisfy eq (5) for all values of N; and we have actually verified that this is the case only for $N\leq 8$.

The algorithm for the new N-cycle scheme is nearly as simple as that for the old one. Steps 0–4 are the same; and steps 5 and 6 are replaced by:

5a. If $a=0$, replace a with $a-\ (1/\delta t)$ and b with $b-(1/\delta t)$; if $a<0$, do nothing.

5b. Replace a with $a+\ (1/\Delta t)$ and b with $b+(1/\Delta t)$.

6. If $a<0$, return to step 1; if $a=0$, the procedure is completed.

Applying the new algortihm with $N=3$, we find that

$$x_i(t_0+\Delta t)\sim x_i+\frac{dx_i}{dt}\,\Delta t+\frac{1}{2}\,\frac{d^2x_i}{dt^2}\,\Delta t^2 +\frac{1}{6}\left(\frac{d^3x_i}{dt^3}-\frac{1}{6}\,F_{i,j,k}F_jF_k\right)\Delta t^3+.\ .\ .\ . \tag{20}$$

Thus, the new three-cycle scheme is also only of second order. However, comparing eq (20) and (17), we see that the error in the coefficient of Δt^3 made by the new three-cycle scheme is precisely the negative of that made by the old one. Likewise, the errors in the coefficients of both Δt^3 and Δt^4 made by the new four-cycle scheme prove to be the negatives of the errors made by the old one.

It is evident then that one can obtain a true third-order or fourth-order scheme by carrying out both three- or four-cycle schemes separately and then averaging the results. We are not inclined to recommend such a procedure, however, since it would require twice the amount of computation, not to mention additional storage space. Nevertheless, it is possible that the errors in the coefficients of Δt^3 and also Δt^4 will tend to cancel when the old and new N-cycle schemes are used in *alternate* steps. We find, in fact, that a true third-order scheme for advancing from t_0 to $t_0+2\Delta t$ may be obtained by advancing from t_0 to $t_0+\Delta t$ with one three-cycle scheme and from $t_0+\Delta t$ to $t_0+2\Delta t$ with the other.

When the two four-cycle schemes are used in alternate steps, the result is a third-order rather than a fourth-order scheme for advancing two steps forward. However, the error in the coefficient of Δt^4 made by following the new four-cycle scheme with the old one proves to be precisely the negative of the error made by following the old four-cycle scheme with the new one. There is the further possibility of following one combination of four-cycle schemes by the other combination; and we find that a true fourth-order scheme for advancing from t_0 to $t_0+4\Delta t$ is obtained by advancing from t_0 to $t_0+\Delta t$ with the old four-cycle scheme, from $t_0+\Delta t$ to $t_0+\ 2\Delta t$ and then from $t_0+2\Delta t$ to $t_0+3\Delta t$ with the new one, and from $t_0+3\Delta t$ to $t_0+4\Delta t$ with the old one.

There are some problems where one may not desire the most accurate attainable solution of the ordinary differential equations being solved. This may be the case, for example, when the solutions contain high-frequency oscillations of limited physical interest, which may be conveniently suppressed by a suitably chosen time-differencing scheme.

There are also problems where a highly accurate solution is desired. In this event, for an easily programmed procedure, which for fixed δt requires no more total computation time than the uncentered-difference approximation and no more storage space, we recommend the first N-cycle scheme with a moderately large value of N, particularly if the equations are essentially linear. For improved results when the equations are nonlinear, again without additional computation or storage, we suggest using the two four-cycle schemes, switching from one to the other after completing each odd-numbered step.

ACKNOWLEDGMENTS

This research has been supported principally by the Atmospheric Sciences Section, National Science Foundation, under NSF Grant GA-10276. Some of the earlier results were obtained with support from the Weather Bureau, Environmental Science Services Administration, under Contract E-73-69(N).

REFERENCE

Henrici, Peter, *Discrete Variable Methods in Ordinary Differential Equations*, John Wiley & Sons, Inc., New York, N.Y., 1962, 407 pp.

[*Received September 20, 1970; revised February 8, 1971*]