On Numerical Integration of Ordinary Differential Equations

By Arnold Nordsieck

Abstract. A reliable efficient general-purpose method for automatic digital computer integration of systems of ordinary differential equations is described. The method operates with the current values of the higher derivatives of a polynomial approximating the solution. It is thoroughly stable under all circumstances, incorporates automatic starting and automatic choice and revision of elementary interval size, approximately minimizes the amount of computation for a specified accuracy of solution, and applies to any system of differential equations with derivatives continuous or piecewise continuous with finite jumps. ILLIAC library subroutine #F7, University of Illinois Digital Computer Laboratory, is a digital computer program applying this method.

1. Introduction. A typical common scientific application of automatic digital computers is the integration of systems of ordinary differential equations. The author has developed a general-purpose method for doing this and explains the method here. While it is primarily designed to optimize the efficiency of large-scale calculations on automatic computers, its essential procedures also lend themselves well to hand computation. The method has the following characteristics, all of which are requisite to a satisfactory general-purpose method:

a. Thorough stability with a large margin of safety under all circumstances. (Instabilities in the subject differential equations themselves are, of course, reflected in the solution, but no further instabilities are introduced by the numerical procedures.)

b. Any integration is started with only the essential initial conditions, i.e. there is a built-in automatic starting procedure.

c. An optimum elementary interval size is automatically chosen, and the choice is automatically revised either upward or downward in the course of an integration, to provide the specified accuracy of solution in the minimum number of elementary steps.

d. The derivatives need be computed just twice per elementary step, which is the minimum consistent with controlling accuracy.

e. Any system of equations

$$\frac{dy_i}{dx} = f_i(x, y_1, y_2 \cdots) \qquad i = 1, 2, \cdots n \tag{1}$$

$$\left(\text{often written } \frac{dy}{dx} = f(x, y) \text{ for short}\right)$$

can be treated for which the f_i are either continuous or piecewise continuous functions with finite jumps.

f. The solution is computed at (although not necessarily only at) equally spaced values of the independent variable x, with specifiable spacing.

Received May 6, 1961. The research presented in this paper was supported in part by the U. S. Army, Navy and Air Force.

Further useful though perhaps not indispensable characteristics of the method are:

g. Enough numerical information is developed to make interpolation or evaluation of functions (e.g., roots) of the solution possible with accuracy equivalent to the solution accuracy.

h. The sense of integration can be reversed.

Characteristic a) is essential for getting trustworthy results in lengthy automatic computations because the number of elementary steps may be as large as 10^5 or 10^6 or more, and disturbances in unstable methods typically grow exponentially with the number of steps. Characteristic b) is not only a convenience but also insures that in the integration of intrinsically unstable equations, in which early errors tend to be strongly magnified, the starting errors do not dominate. Characteristic c) relieves the human being of the often difficult task of determining the correct interval in advance. Where the human being must specify the interval for a computation not to be performed by himself he tends to make up for uncertainty by a conservatively small interval choice. Characteristics c) and d) together thus make for efficient use of computer time, and the saving in computer time can easily be a factor of 10 or even much more in the handling of problems in which the interval should vary.

In regard to the question of relating our method to previously available methods, we wish to make clear at the outset that it is equivalent to a reformulation of the method of Adams [1, p. 53–55], [2, p. 81–82] for it uses effectively the same quadrature formula as does Adams. However, the formulation and the point of view are so different that it is instructive and seems appropriate to explain the method starting from first principles, as we shall do below, rather than starting from Adams' quadrature formula.

Presently available methods may be divided into two classes: those involving no memory and those involving some memory, of the past behavior of the solution. The Runge-Kutta methods [1, p. 72–75], [2, p. 59–74] are typical of the first class, the Milne methods [1, p. 64–70], [2, p. 84] and the Adams methods of the second. It has been clear for some time that the methods with memory are superior in accuracy for a given elementary interval size and a given amount of computational labor since they permit a better approximating curve to be fitted over the elementary interval. Our method involves such memory. In return for this superiority of the methods with memory we must cope with two problems quite foreign to the memoryless methods: how to start off, since at the beginning there is nothing to remember; and how to prevent the remembered numerical information from behaving unstably.

Two further problems must be dealt with in order to implement the automatic choice and revision of the elementary interval, namely, choosing which quantities to remember in such a way that the interval may be changed rapidly and conveniently and developing an appropriate set of rules for controlling the interval size. Thus the four major problems are: automatic starting, stability, choice of quantities to remember and interval control logic. The last of these four is the most intricate.

As with most methods, there exist lower and higher "order" versions of this method. The author prefers to use the term "degree" rather than "order", since all methods are ultimately equivalent to finding a polynomial of some given *degree* approximating the solution of the system of equations, and since the term "order"

is already standardized usage for the number of equations n in (1). We have chosen and recommend degree 5, which corresponds to a truncation error $0(h^7)$ per elementary step of length h, for large-scale digital computer operations. This represents an advantageous return in accuracy per step with quite large steps, while still not overdoing the accuracy when the choice of h is limited to inverse powers of 2, as is natural in a binary computer.

The order n of the system (1) is immaterial to a large part of our discussion, so that we can advantageously use the simpler notation $dy/dx = f(x, y)$ for (1), regarding y and f as vector-like objects with n real numbers as components. The independent variable x is, of course, a single real number. Whenever the multicomponent character of y and f makes a significant difference in the discussion we shall so note.

In Section 2 the choice of quantities to be remembered is discussed, in 3 the numerical procedure and the associated stability theory are developed, in 4 certain parameters of the method are adjusted for optimum stability and accuracy, in 5 the procedure for modifying the interval is given, in 6 the characteristic behavior of the remembered quantities is described, in 7 error estimation is discussed, in 8 the automatic interval control logic is developed, in 9 automatic starting is described and finally in Section 10 the results of certain test problems done by this method are exhibited. In Appendix A are collected the working formulas and error estimates for degrees 3 through 6 of the approximating polynomial. Appendix B contains a schematic flow chart for programming the method for a digital computer, with computing time estimates. Appendix C is a discussion of control of roundoff errors in iterative numerical procedures.

2. Choice of Quantities to Remember. It is immediately clear that quantities like differences $y(x) - y(x - h)$, $y(x - h) - y(x - 2h)$, etc., and/or higher differences would constitute a poor choice to remember, for changing the interval in terms of these is a cumbersome process involving much interpolation and/or extrapolation. (Ignoring the remembered quantities whenever the interval is to be changed and starting again "from scratch" would entail serious loss of accuracy and of time).

We take our cue from the remark above to the effect that all methods of numerical integration are equivalent to finding an approximating polynomial for $y(x)$. Of the many ways of specifying a polynomial of degree m by $m + 1$ constants there is one way which is interval-independent, namely: to specify the 0th to mth derivatives of the polynomial evaluated at the current value of x. These particular $m + 1$ quantities specify the same polynomial no matter what the interval is, being in fact defined with no reference to an interval at all. They would be ideal from the point of view of interval modification. However, they are not suitable for automatic computation because the higher derivatives may vary enormously in magnitude and are thus not conveniently stored in a "fixed-point" arithmetic operation.*

* The discussion in the present paper is limited to "fixed-point" arithmetic procedures. The question whether a "floating-point" version of the method could be made safe against loss or illusory gain of significance of the quantities in the course of a long computation, and otherwise trustworthy, is for future investigation. The possible freedom to store just the higher derivatives of the approximating polynomial and the increased freedom from scaling problems certainly suggest that one investigate the floating-point possibility.

In order to see how to modify our choice so as to cure the latter difficulty, we consider how the $m + 1$ derivatives would actually be used in the computation. A typical important use is, in the first phase of the integration step from x to $x + h$, to "predict" a trial value of $y(x + h)$ from the formula:

$$y^p(x+h) = y(x) + h\left\{f(x, y(x)) + \frac{h}{2!}P_5''(x) + \frac{h^2}{3!}P_5'''(x) + \frac{h^3}{4!}P_5''''(x) + \frac{h^4}{5!}P_5'''''(x)\right\} \tag{2}$$

where m has been made 5 and $P_5(x) = y(x)$, $P_5'(x) = f(x, y(x))$, $P_5'' \cdots P_5'''''$ are the 6 aforementioned derivatives of the approximating polynomial evaluated at x. Formula (2) is written in the special way shown, with one factor h external to the $\{\ \}$, because we may expect f to be computed to full register accuracy on occasion, which suggests that the remaining terms in the $\{\ \}$ be kept to the same accuracy; and because for the case of small h and many steps (many successive applications of formulas like (2)) we can minimize the accumulation of roundoff errors in y by keeping log $(|h|^{-1})$ more places in $h\{\ \}$ than we keep in the $\{\ \}$ itself. Formula (2) in the form written then suggests that the appropriate quantities to store in the computer registers are, besides the always necessary $y(x)$ and $f(x, y(x))$, the four quantities

$$\begin{aligned} a(x) &= \frac{h}{2!}P_5''(x) & b(x) &= \frac{h^2}{3!}P'''(x) \\ c(x) &= \frac{h^3}{4!}P_5''''(x) & d(x) &= \frac{h^4}{5!}P'''''(x) \end{aligned} \tag{3}$$

We may reasonably expect these quantities to stay within register capacity since an appropriate choice of h will just cause the successive terms in the $\{\ \}$ to decrease in magnitude no matter how large the $P_5^{(i)}$ themselves become. Although the quantities (3) are not completely interval-independent, they depend on the interval in such a simple way that interval change involves merely multiplying each by a constant, and in the important practical case of a binary computer and intervals restricted to inverse powers of 2 the change is achieved simply by shifting the numbers. Formula (3) seems accordingly to be essentially the unique sensible choice, at least for a fixed point arithmetic procedure.

We emphasize that the quantities y, f, a, b, c, d as they exist in the computer registers and appear in our discussion are formally defined from successive derivatives of an *approximating polynomial*, so that they always exist since an approximating polynomial always exists, whether or not the exact solution of the original problem (1) has five derivatives. If the original problem involves a discontinuous f, the quantities $a \cdots d$ tend to get large because of that, but concurrently tend to get small because of interval decrease, with the overall result that they stay within register capacity. While the existence of an approximating polynomial is assured, its quality as an approximation of the exact solution of (1) depends on how it is developed; in subsequent sections we discuss how to develop it in an optimum way.

3. Taylor's Theorem Procedure Modified for Stability. In order to have a completely defined integration procedure we must have rules for determining all of the

quantities $y(x+h)$, $f(x+h)$, $a(x+h) \cdots d(x+h)$ when given $y(x)$, $f(x)$, $a(x) \cdots d(x)$ and the differential equation (1). (The starting problem, namely to determine y, f, a, b, c, d at $x+h$ given only $y(x)$ and $f(x)$ and the differential equation, is discussed below in Section 9). Consider first the ordinary Taylor's series formulas terminated at h^6, which in terms of $a, b \cdots$ read:

$$
\begin{aligned}
y(x+h) &= y(x) + h\{f(x) + a(x) + b(x) + c(x) + d(x) + e(x)\} \\
f(x+h) &= f(x) + 2a(x) + 3b(x) + 4c(x) + 5d(x) + 6e(x) \\
a(x+h) &= a(x) + 3b(x) + 6c(x) + 10d(x) + 15e(x) \\
b(x+h) &= b(x) + 4c(x) + 10d(x) + 20e(x) \\
c(x+h) &= c(x) + 5d(x) + 15e(x) \\
d(x+h) &= d(x) + 6e(x)
\end{aligned} \tag{4}
$$

Here we have introduced one more quantity $e(x)$ analogous to $a \cdots d$, which we eliminate forthwith by using the differential equation. The system (4) as it stands is incomplete, having one less equation than it involves quantities. But by identifying the second formula of (4) with $f(x+h, y(x+h))$ calculated from the differential equation, we can eliminate $e(x)$ and get:

$$
\begin{aligned}
y(x+h) &= y(x) + h\{f(x) + a(x) + b(x) + c(x) + d(x) \\
&\qquad + \tfrac{1}{6}[f(x+h, y(x+h)) - f^p]\} \\
f(x+h) &= f(x) + 2a(x) + 3b(x) + 4c(x) + 5d(x) \\
&\qquad + 1[f(x+h, y(x+h)) - f^p] \\
a(x+h) &= a(x) + 3b(x) + 6c(x) + 10d(x) \\
&\qquad + \tfrac{15}{6}[f(x+h, y(x+h)) - f^p] \\
b(x+h) &= b(x) + 4c(x) + 10d(x) \\
&\qquad + \tfrac{20}{6}[f(x+h, y(x+h)) - f^p] \\
c(x+h) &= c(x) + 5d(x) \\
&\qquad + \tfrac{15}{6}[f(x+h, y(x+h)) - f^p] \\
d(x+h) &= d(x) \\
&\qquad + 1[f(x+h, y(x+h)) - f^p]
\end{aligned} \tag{5}
$$

where $f^p \equiv f(x) + 2a(x) + 3b(x) + 4c(x) + 5d(x)$, the "predicted" value of $f(x+h)$.

Now the system (5) augmented by the differential equation is complete, for the first equation of (5) and the differential equation together constitute an implicit system determining $y(x+h)$ and $f(x+h)$; the second equation of (5) is an identity and the next four then determine $a(x+h) \cdots d(x+h)$ straightforwardly.

Having arrived at the scheme (5) quite directly from Taylor's theorem we entertain the possibility of using it for numerical integration. A small amount of hand computation using (5) establishes that it is: a) very accurate indeed, and b) very unstable indeed, with small disturbances growing approximately as $(-10)^s$ in s steps.

These two phenomena are closely related. The high accuracy derives from basing the scheme directly and exactly on Taylor's theorem; however, just because it is so based it has another property, namely reversibility. If we apply (5) to go from x to $x + h$ and reapply (5) with reversed h to retrace from $x + h$ to x, we recover the original quantities $y, f \cdots d$ precisely. Now a process reversible in this sense cannot be stable, for it cannot damp out small disturbances (i.e., "forget" or "lose information") as it must to be stable. Stated in terms of the eigenvalues of the stability matrix M discussed later, reversibility implies that the matrix for backward integration is the inverse of the matrix for forward integration, which is inconsistent with the condition for stability, namely that for both these matrices all eigenvalues except one must lie inside the unit circle. (The only exception to the last statement occurs when the stability matrix is 1×1, which corresponds to the trapezoidal method $m = 1$ with no "memory.")

We search then for such a modification of (5) as will provide stability with minimum degradation of accuracy. The following discussion will establish that a usable and in fact essentially optimum modification of (5) consists of replacing the series of six coefficients 1/6, 1, 15/6, 20/6, 15/6, 1 multiplying the [ ] by new constant coefficients $Y = 95/288, 1, A = 25/24, B = 35/72, C = 5/48, D = 1/120$ respectively and leaving (5) otherwise unaltered. It is interesting to note that the ratios of the new coefficients to the old form a rather strongly decreasing sequence: 1.98, 1, 0.42, 0.15, 0.042, 0.0083, which reminds one of the well known technique for stabilizing electrical filters involving feedback by somewhat enhancing the low frequency gain and strongly depressing the high frequency gain.

In searching for an appropriate modification of (5) it is inadvisable to tamper with the coefficients *not* pertaining to the [ ], and this will be borne out by later analysis, for these coefficients are clearly just such as to make the integration of a 5th degree polynomial $y(x)$ come out exact (the [ ] will vanish for $y(x)$ a 5th degree polynomial). However, the coefficients multiplying the [ ] have no such unique significance and we are free to modify them to suit our purpose.

To dispose of the possibility of generalizing the coefficient 1 in the second equation of (5): So long as this coefficient remains 1 we can delete the second equation entirely from the considerations as being merely an identity, and we ultimately do just that. In the interests of generality the author has experimented some with modifying this particular coefficient numerically and has indeed found that any value other than 1 for it, beside costing an additional multiplication, degrades both the accuracy and the stability.

The remaining 5 equations of (5) with the coefficients 1/6, 15/6, $\cdots$ 1 replaced by arbitrary constants Y, A, B, C, D, may then be studied for stability by introducing a small variation of each of the 5 independent quantities (y, ha, hb, hc, hd), namely (δy, δha, δhb, δhc, δhd), and studying how this latter quintuple changes as we integrate from x to $x + h$ [3]. The quantity f is to be regarded as not independent but a function of y in virtue of the differential equation. After some calculation we find that the quintuple (δy, $h\delta a$, $h\delta b \cdots h\delta d$), regarded as a 5-component vector $V(x)$, obeys the equation

$$V(x + h) = MV(x) \tag{6}$$

where M is a 5×5 matrix:

$$M = \tag{7}$$

$$\begin{Vmatrix} 1+\dfrac{p}{1-Yp} & 1+\dfrac{Y(p-2)}{1-Yp} & 1+\dfrac{Y(p-3)}{1-Yp} & 1+\dfrac{Y(p-4)}{1-Yp} & 1+\dfrac{Y(p-5)}{1-Yp} \\ \dfrac{Ap^2}{1-Yp} & 1+\dfrac{A(p-2)}{1-Yp} & 3+\dfrac{A(p-3)}{1-Yp} & 6+\dfrac{A(p-4)}{1-Yp} & 10+\dfrac{A(p-5)}{1-Yp} \\ \dfrac{Bp^2}{1-Yp} & \dfrac{B(p-2)}{1-Yp} & 1+\dfrac{B(p-3)}{1-Yp} & 4+\dfrac{B(p-4)}{1-Yp} & 10+\dfrac{B(p-5)}{1-Yp} \\ \dfrac{Cp^2}{1-Yp} & \dfrac{C(p-2)}{1-Yp} & \dfrac{C(p-3)}{1-Yp} & 1+\dfrac{C(p-4)}{1-Yp} & 5+\dfrac{C(p-5)}{1-Yp} \\ \dfrac{Dp^2}{1-Yp} & \dfrac{D(p-2)}{1-Yp} & \dfrac{D(p-3)}{1-Yp} & \dfrac{D(p-4)}{1-Yp} & 1+\dfrac{D(p-5)}{1-Yp} \end{Vmatrix}$$

with

$$p \equiv h\,\frac{\partial f(x, y)}{\partial y}\,. \tag{8}$$

We note that the 5-dimensional vector space of V and M is a different space from the n-dimensional space of y, f, a, etc.

We have treated p as though it were a scalar quantity even though for $n > 1$ it is really an $n \times n$ matrix $h(\partial f_i/\partial y_j)$; but it is only the *smallness* of p, insurable by appropriate choice of h, which is important in our argument, not its matrix character. The difference between $p(x + h)$ and $p(x)$ has also been neglected, for it gives rise to errors involving one factor h more than we need consider.

The characteristic equation $0 = |\lambda\delta_{rs} - M_{rs}|$ of M turns out to be:

$$\begin{aligned} 0 = {} & (1 - Yp)(\lambda - 1)^5 \\ & + [2A + 3B + 4C + 5D - (1 + A + B + C + D)p](\lambda - 1)^4 \\ & + [6B + 24C + 70D - (2A + 6B + 14C + 30D)p](\lambda - 1)^3 \\ & + [24C + 180D - (6B + 36C + 150D)p](\lambda - 1)^2 \\ & + [120D - (24C + 240D)p](\lambda - 1) - (120D)p. \end{aligned} \tag{9}$$

One root of this equation, which may be found by substituting a power series in p into it, and which we shall call the principal root λ_0, is essentially a function of p only, depending but slightly on Y, A, B, C and D:

$$\begin{aligned} \lambda_0 = e^p + {} & \frac{6Y - 3 + A - (1/5)C}{D}\,\frac{p^6}{6!} \\ & + \frac{-49 + 105Y + 14A + (7/5)B - (14/5)C - D}{D}\,\frac{p^7}{7!} + 0(p^8). \end{aligned} \tag{10}$$

This is a consequence of retaining the coefficients in (5) not pertaining to the [ ]. The root λ_0 is thus essentially a property of the differential equation system (1), and whether or not it lies inside the unit circle in the complex λ plane determines

whether the subject system, as distinguished from our numerical method, is stable or not. On the other hand, the four further roots of (9), which we shall call "extraneous" roots, depend strongly on A, B, C, D and only weakly on p and Y; their location relative to the unit circle determines the stability of the integration method itself. These roots must lie inside the unit circle for stability of the method, and the nearer they are to the origin the more stable the method will be.

4. Determination of Parameters. The parameters $Y \cdots D$ are now to be chosen, primarily to optimize the stability of the method and secondarily, if any freedom is left over, to optimize the accuracy within the restriction of optimum stability. The author regards optimum stability as essential to an automatic general-purpose method, for the rapid elimination of disturbances characteristic of good stability not only makes an automatic starting process feasible and permits accurate integration across finite discontinuities of f, as we shall see below, but also minimizes the error due to interaction of disturbances with non-linearities of the differential equations.† Since there are four extraneous eigenvalues whose locations in the complex plane we wish to control and we have five parameters free, we can expect to have considerable control over stability and accuracy. What actually happens is that A, B, C, D determine stability and Y is left free to optimize accuracy. Thus we can arrange for a truncation error of $0(h^7)$ even though we are using 5th degree polynomials, the explanation being that in each integration step we use both the 5th degree polynomial available at the beginning and the one available at the end of the step.

Now it is easy to bound $|p|$ (bound the magnitudes of its eigenvalues if it is a matrix) by control of h during the numerical integration process, while it is much more difficult actually to compute p for $n > 1$. Therefore it seems best and is certainly simplest to choose Y, A, B, C, D independent of p, i.e. as absolute constants, in such a way that stability is guaranteed for as large a range of p as possible. This is substantially accomplished by considering (9) with $p = 0$ (whereupon Y drops out, indicating that it has little influence on the stability of the method) and then choosing A, B, C, D so that the four extraneous roots coincide at 0. Thus, we require (9) for $p = 0$ to take the form $(\lambda - 1)\lambda^4 = 0$, and it does that for $A = 25/24$, $B = 35/72$, $C = 5/48$, $D = 1/120$. The choice of Y is then made to nullify the coefficient of p^6 in (10), which has no effect on the stability but optimizes the accuracy. This determines $Y = 95/288$. For stability for $p \neq 0$ we then depend on the fact that the extraneous roots are continuous functions of p, so that they cannot move very far from the origin provided p is appropriately limited.

In order to get a better picture of the behavior of the extraneous roots as functions of p, we first note that for small p they are the roots of

$$\lambda^4 = -\frac{3}{160}\, p \tag{11}$$

as can be read off from (9) with the chosen values of the parameters inserted. It is

† A report by E. Fehlberg [4], has just come to the author's attention. Fehlberg exhibits other choices of parameters which produce smaller truncation error than Adams' and the author's choice, *but* at the expense of much poorer stability, cf. Fehlberg's tables 3 and 4. For $m = 5$ the gain in computing speed for the same error is greatest and is $(1/0.0801)^{1/7} = 1.43$, which the author considers not worth the risks incurred with the much poorer stability.

fortunate that the numerical coefficient in (11) is so small, for the $p^{1/4}$ dependence of the roots is a rather strong dependence. The roots have also been computed for p a real number between -1 and $+1$, and these are shown in Figure 1. We see that stability will be guaranteed with a comfortable margin of safety if the interval is so chosen that p lies effectively inside the dashed curve. This boundary corresponds to $| Yp | \leqq 1/8$, which is a convenient form of test for a computer.

The author has done considerable searching for other favorable choices of A, B, C, D with the thought in mind that if the extraneous roots never coincided they might move away from the origin more slowly as $| p |$ increased, than they do according to (11). However, all other choices tried were inferior in point of both stability and accuracy.

The choice of parameters made above seems optimum among choices restricted to constants independent of p. The potential advantage of a more elaborate procedure in which the matrix p is numerically computed at every step and $Y, A \cdots, D$ are made chosen functions of p, implying a nonlinear process tailored to the subject differential equation system, is an interesting topic for future investigation, for it might lead to faster (though less accurate) methods of solving some classes of equations.

The working equations of the method have now been determined completely and they are summarized in Appendix A, equations ($A4$).

The working equations having been determined, the precise connection with

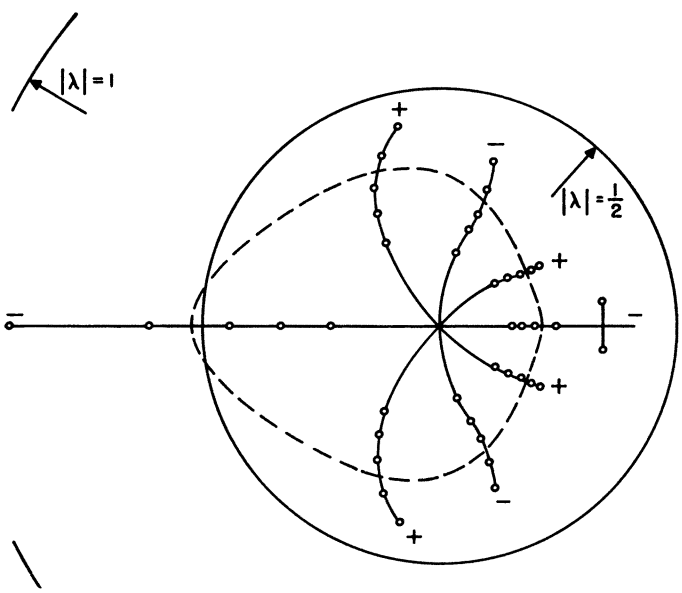

FIG. 1.—The extraneous roots of the characteristic equation as functions of p for real p, plotted in the complex λ plane. As p departs from zero these roots depart from the origin along the loci shown. Loci marked $+$ correspond to positive p and loci marked $-$ to negative p. Counting outward from the origin along each locus, the points plotted represent in order, $|p| = \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1$. The real positive extraneous root coalesces with the principal root at $p = -0.88$, producing a conjugate pair. The dashed curve encloses all extraneous root values permitted by the interval control tests, which limit $|p|$ to values $\leqq .38$.

other methods can be deduced by ascertaining the equivalent quadrature formula for the method. This can be done by expressing the part $h\{\quad\}$ of the first working equation in terms of past values $f(x-h)$, $f(x-2h)$, etc., by repeated application of the working equations. We find that the equivalent quadrature formula is

$$y(x+h) - y(x) = \frac{h}{1440}\{475f(x+h) + 1427f(x) - 798f(x-h) + 482f(x-2h) - 173f(x-3h) + 27f(x-4h)\} \tag{12}$$

which agrees exactly with the Adams formula of corresponding degree. By way of confirmation of this conclusion we observe that the characteristic equation for small variations in the Adams method coincides with (9) when the chosen values of the parameters are inserted into (9).

5. Change of Interval. We indicate how to perform the three useful changes of interval: $h' = -h$, $h' = \beta h$ and $h' = \beta^{-1}h$ (where in binary computer operations β is preferably taken equal to 2):

(13)

Reversal	Increase	Decrease	
$-h$	βh	$\beta^{-1}h$	replaces h
y	y	y	" y
f	f	f	" f
$-a$	βa	$\beta^{-1}a$	" a
b	$\beta^2 b$	$\beta^{-2}b$	" b
$-c$	$\beta^3 c$	$\beta^{-3}c$	" c
d	$\beta^4 d$	$\beta^{-4}d$	" d

The rules for changing a, b, c, d are clear from (3).

The simplicity of the rules for changing the interval is evident here.

Every change of interval of any of the three types induces a disturbance in the system, but the disturbance affects mainly the higher derivatives and clears out in a few steps because of the choice of parameters. These transient phenomena will be described in more detail in the next following section.

6. Behavior of a, b, c, d. A qualitative understanding of the behavior of the quantities constituting the method's "memory" is required in order correctly to design the interval control logic and the starting procedure.

We first describe the "normal" or steady behavior which prevails when no transients have been induced by interval change or f-discontinuity or otherwise, within the preceding 4 to 8 steps or so. Then the quantities a, b, c, d "lag" behind the current value of x, a a little, b more, c still more, and d most, in the sense that they equal the "true" higher derivatives of y evaluated at points $x - \theta h$, where $0 < \theta \leqq 2$. This lagging behavior is related to, and is in fact a necessary consequence of stability. A close analogy exists between this and the "stable physically realizable filter" of electrical engineering theory, and likewise the causality discussions in physics. The indicated behavior may be established (and incidentally some formulas of later use for deriving the truncation error found) by assuming that a 7th degree polynomial $y = P_7(x)$ satisfies the differential equation exactly and that f, a, b, c, d are corresponding polynomials of 6th $\cdots$ 2nd degree, and solving the working

equations ($A4$) for the coefficients by some rather lengthy algebra. The result is

$$
(14)\qquad
\begin{aligned}
a(x) &= \frac{h}{2!}\,y''(x) - 72\,\frac{h^5}{6!}\,y^{VI}(x) + 840\,\frac{h^6}{7!}\,y^{VII}(x)\\
b(x) &= \frac{h^2}{3!}\,y'''(x) - 100\,\frac{h^5}{6!}\,y^{VI}(x) + 1110\,2/3\,\frac{h^6}{7!}\,y^{VII}(x)\\
c(x) &= \frac{h^3}{4!}\,y''''(x) - 52\,1/2\,\frac{h^5}{6!}\,y^{VI}(x) + 525\,\frac{h^6}{7!}\,y^{VII}(x)\\
d(x) &= \frac{h^4}{5!}\,y'''''(x) - 12\,\frac{h^5}{6!}\,y^{VI}(x) + 91\,\frac{h^6}{7!}\,y^{VII}(x).
\end{aligned}
$$

These formulas are then in error by $0(h^7)$ for any general y which is differentiable sufficiently many times. The last of the four formulas shows that $d(x) = \frac{h^4}{5!}\,y'''''(x - 2h) + 0(h^6)$, so that d lags by very nearly two steps.

We may describe this "normal" behavior in another way, namely, by observing that the polynomial evaluated at x is always essentially the polynomial fitted to the values of y at x, $x - h$, $x - 2h$, $x - 3h$, $x - 4h$. The 5th derivative of this polynomial naturally agrees best with the 5th derivative of the true solution y at the mid-point of the fitting interval, which explains the last equation of (14). The close relation of our method to the Adams method also becomes clear from this point of view. When we advance from x to $x + h$ the working equations in effect change the old polynomial fitted at $x - 4h \cdots x$ into one fitted at $x - 3h \cdots x + h$. In the approximation that the 4th powers of the extraneous characteristic roots may be neglected, all disturbances clear out in precisely 4 steps, corresponding to the memory of the method having a "time-span" of just 4 steps. Thus, we have arranged *effectively* to keep and use what Adams actually keeps and uses, namely the last four previous ordinates, whereas actually we keep quantities much more suitable for interval modification.

As for "abnormal" behavior of the remembered quantities, the simplest important case of this occurs upon reversal. The quantities exhibit a hysteresis after reversal, most pronounced in the case of $d(x)$ which has the most lag. The behavior of d in reversal is illustrated in Figure 2, which shows essentially that d stays quite strictly constant for four steps after a reversal, then abruptly resumes normal behavior. Since what was a backward-fitted polynomial before reversal becomes a forward-fitted polynomial after reversal, we may say that d and indeed the polynomial as a whole "freezes", remains the same and marks time until enough steps have been executed for it to become normal for the current point x, then behaves normally.

The other important type of abnormal behavior is the response to shock excitation. Shock excitation occurs severely in starting, when the normal $a \cdots d$ are not known; mildly enough to be harmless in increasing or decreasing the interval, when the main terms but not the "lag" terms in (14) are correctly modified by the simple rules (13); and more or less severely when f has a discontinuity so that the change in the polynomial is large in one step. Here again $d(x)$ shows the most violent behavior and its behavior in all shock-excited transients is essentially an oscillation lasting just four steps.

TABLE 1

x	y	f	$24a$	$72b$	$48c$	$120d$
x_0	0	0	0	0	0	0
$x_0 + h$	$\left(\frac{1}{2} - \frac{245}{1440}\right)h\Delta$	Δ	25Δ	35Δ	5Δ	Δ
$x_0 + 2h$	$\left(\frac{3}{2} + \frac{217}{1440}\right)h\Delta$	Δ	-23Δ	-69Δ	-13Δ	-3Δ
$x_0 + 3h$	$\left(\frac{5}{2} - \frac{119}{1440}\right)h\Delta$	Δ	$+13\Delta$	$+45\Delta$	11Δ	3Δ
$x_0 + 4h$	$\left(\frac{7}{2} + \frac{27}{1440}\right)h\Delta$	Δ	-3Δ	-11Δ	-3Δ	$-\Delta$
$x_0 + 5h$	$\frac{9}{2}h\Delta$	Δ	0	0	0	0

In order to become familiar with the detailed behavior of such a transient, we treat a simple case which approximates the general case of an isolated discontinuity of f with finite jump: Let $y \equiv f \equiv 0$ for $x \leqq x_0$ and assume that $a \cdots d$ have their normal values of zero for $x \leqq x_0$. Let $f \equiv \Delta =$ constant for $x > x_0$ and apply the working equations (14) five times in succession and Table 1 results.

Evidently $\frac{2!}{h}a$, $\frac{3!}{h^2}b$, $\frac{4!}{h^3}c$ and $\frac{5!}{h^4}d$ are behaving like numerical approximations to the "δ-function" of x and its first, second and third derivatives respectively. Meanwhile the transient in y, represented by the terms with denominator 1440, is a decreasing oscillation also lasting just four steps, and the ultimate value of y is exactly what one would get by connecting the last point sampled at which $f = 0$ with the first point sampled at which $f = \Delta$ by a straight line segment. This essentially best performance in integrating across a discontinuity is unique to our choice of parameters. A reasonable upper bound for the magnitude of the error in y due to such a discontinuity is $\frac{1}{2} \mid h\Delta \mid$ where Δ is the jump in f. One can hardly do better without sampling in between these two points, i.e. decreasing h; but by controlling h one can bound this error.

7. Estimation of Errors. In discussing errors in the solution $y(x)$ we must distinguish between the error present at the beginning of an elementary step and the error contributed by the execution of that step. The error present at the beginning of a step, sometimes called inherited error, is the net result of all the errors contributed by all the previous steps, each modified according to the action of the differential equation between the point of origin x' of the error and the current point x. Letting $E(x')$ represent the error contributed by a step of length h taken at x', we can write for the inherited error at x if the integration began at x_0:

$$E_i(x) = \sum_{x'=x_0}^{x} E(x') \prod_{x''=x'}^{x} \lambda_0(x'') \tag{15}$$

where $\lambda_0 \cong e^p$ is the principal root (10). The sum involves the summand once for every elementary step taken and similarly the product. Equation (15) illuminates the relation between inherited error and error contributed by an individual step.

The product in (15) may also be written in adequate approximation as:

$$\prod_{x''=x'}^{x} \lambda_0(x'') = \exp\left\{\int_{x'}^{x} \frac{\partial f}{\partial y}(x'')\, dx''\right\} \tag{16}$$

which shows that the product is a property of the differential equation, independent of integration method and of interval choice. It is clear then that although by careful design of the method and choice of interval we may be able to reduce $E(x')$ down to about half the least count in the register (but no further because of inevitable rounding), nevertheless such measures have no effect on (16). Consequently if Λ is the largest eigenvalue of the matrix (16) the error at the conclusion of the integration will be in general at least about $\frac{1}{2} \mid \Lambda \mid$ times the least count. The number of correct significant digits may at most be preserved through the calculation if the magnitude of the solution increases by Λ or more; if not, the significance (i.e. the number of correct significant digits) will decrease. If a problem has $\Lambda > \beta^L$, where L is the number of base β digits in the register, then it is useless to attempt the problem at all by fixed point arithmetic, for there will be no correct significant digits left at the end of the calculation. Floating point could help if the magnitude of the solution increases meantime; if not, nothing will help except increased register length.

We have dwelt on the above points because they show that the best that can be done with any method is approximately to preserve the number of correct significant digits in the solution, and this essentially defines a best or optimum method. Some of the test examples exhibited in Section 10 below show nearly complete preservation of significance through as many as 10^5 steps and with Λ as large as 10^6 or so.

Turning now to discussion of $E(x)$, we assert that the contributions to $E(x)$ are: a) truncation error incurred by terminating the formulas (A1) to (A5) with a given power of h; b) discontinuity error incurred in integrating past a discontinuity of f (cf. Section 6); c) iteration error resulting from incomplete iterative solution of the implicit equations for $y(x + h)$; and d) roundoff error resulting from using registers of finite length to perform the arithmetic.

The truncation error may be found by making the same assumptions $y = P_7(x)$ etc. as were made in deriving equations (14) and calculating $y(x + h) - P_7(x + h) - y(x) + P_7(x)$, using the first and second working equations and (14). We find that the truncation part of E, which we call E_t, is given by:

$$E_t(x + h/2) = 72 \frac{h^7}{7!} y^{VII}(x) + 0(h^8). \tag{17}$$

It is interesting to note that the truncation error is closely related to the principal root of the stability matrix. In fact, if we replace p arbitrarily by the operator $h\frac{d}{dx}$, $\left(\text{because the proof that } p \text{ is precisely equivalent to } h\frac{d}{dx} \text{ is not apparent}\right)$, then e^p becomes the "true" displacement operator $e^{h(d/dx)}$ and $\lambda_0(p)$ becomes the approximate displacement operator of the method. Thus the difference $\lambda_0(p) - e^p$ with p replaced by $h\frac{d}{dx}$, and applied to $y(x)$, would seem to yield the truncation error. The term in h^8 in the truncation error was determined by exploiting this relationship,

yielding:

$$E_t(x + h/2) = 72 \frac{h^7}{7!} y^{VII}(x) - 440 \frac{h^8}{8!} y^{VIII}(x) + 0(h^9) \tag{18}$$

The discontinuity error, called E_d, is bounded by the inequality

$$| E_d(x)| \leqq \tfrac{1}{2} | h(f(x_+) - f(x_-))| \tag{19}$$

as we saw in Section 6.

The iteration error, called E_i, depends on how we solve the implicit equation system, and we choose to solve it by doing just two iterations, or more precisely: We calculate a first trial value $y^{(1)}(x + h)$ from equation (1) of (A4) with the [ ] term left off (the "predicted" $y(x + h)$ in Milne's terminology); calculate $f^{(1)}(x + h) = f(x + h, y^{(1)}(x + h))$ and insert it on the right of the complete equation (1) of (A4) to give an improved $y^{(2)}(x + h)$; and repeat the procedure just once more, so that by definition in this method the final values of $y(x + h)$ and $f(x + h)$ are $y^{(3)}(x + h)$, respectively $f(x + h, y^{(2)}(x + h))$. The reasons for choosing so are that f need be calculated only twice, that the convergence of the iterative procedure and the (related) bounds on p can be estimated from two iterations but not from less than two, and that the iteration error is sufficiently small. For the special case $n = 1$ (a single first-order differential equation) one can do better by solving the implicit system by interpolative methods with the same number of computations of the derivative; for general n, however, one would have to compute the derivatives $2n$ times at least in order to apply interpolative methods, which we regard as uneconomical. The convergence is determined by the equation

$$y^{(3)} - y^{(2)} = Yp(y^{(2)} - y^{(1)}); \qquad Y = \frac{95}{288} \tag{20}$$

and the "iteration error" in $y(x + h)$ by

$$E_i = y^{(3)} - y^{(\infty)} = (Yp)^2(y^{(1)} - y^{\infty}) \cong -Y^3p^2h^6y^{VI}(x) \tag{21}$$

which is proportional to h^8 with a small coefficient so long as $| Yp | \leqq \frac{1}{8}$ as we shall require, and is therefore overshadowed in general by the truncation error. Equations (20) and (21) follow from iterative treatment of equation 1 of (A4).

The roundoff error E_r, finally, is determined by the care with which both the computation of derivatives and the computations of (A4) are done, and with sufficient care can be as small as about $\frac{1}{2}$ the least count in the effective register used and approximately statistically independent from step to step. The author has found it best to keep $\log_\beta(| h |^{-1})$ extra "guard" digits in y, above and beyond the number kept in $f, a, \cdots d$, in order to minimize the accumulation of roundoff errors in y when the number of elementary steps is large.

8. Automatic Interval Control Logic. In order to describe the interval control we must first outline the 3 stages in which a step $x \to x + h$ is performed. Stage 1 consists of "predicting" all six quantities $y, f, \cdots d$ at $x + h$, i.e. applying equations (A4) without the [ ] terms, using a tentative value of h. The first tentative value of h actually tried is the value which was accepted in the last previous step or the next larger value if the conditions (given below) for increasing h were fulfilled. Note that Stage 1 is exactly reversible in a digital machine, so that if h later turns out to

be wrong the beginning values of $y \cdots d$ can be exactly recovered without the need for additional registers for saving them. Stage 2 consists of solving the implicit equation system for $y(x + h)$ and $f(x + h)$ by iterating twice as explained in the preceding section. This stage is not exactly reversible and $2n$ registers are therefore provided for saving the beginning values of y and f. At the conclusion of Stage 2 enough information has been developed to decide whether the interval tentatively being used is small enough; if it turns out to be not small enough the beginning values of $y \cdots d$ are recovered, the interval is reduced (by a factor $\beta^{-1} = \frac{1}{2}$ in a binary computer) and Stage 1 is again entered. If the tentative interval is found adequate we proceed to Stage 3, which consists of "correcting" a, b, c, d by adding the [ ] terms.

Two tests are made at the conclusion of Stage 2 and failure of either signifies that h is too large; the two tests are respectively

$$| y_i^{(3)} - y_i^{(2)} |_{\max} \leqq \tfrac{1}{8} | y_i^{(2)} - y_i^{(1)} |_{\max} \tag{22a}$$

and

$$| f_i(x + h) - f_i^p |_{\max} \leqq \beta^{-e} / | h | \tag{22b}$$

where e is a specifiable positive integer and "max" means the largest of the n components $i = 1, 2, \cdots n$. It is clear that these tests are first possible at the end of Stage 2, since they involve quantities developed only in that stage. While the tests are being made it is also determined whether both tests are "over-satisfied", i.e. so well satisfied that the next larger h would likely also satisfy them, and if so the interval may be tentatively increased for the next following step.

Satisfying test (22a) insures that the largest eigenvalue of p does not exceed 0.38 in magnitude (cf. equation (20)) and, therefore, that the stability is good (cf. Figure 1) and also that the iteration error is small enough to be overshadowed by the truncation error (cf. equation (21)). The test is not formulated in the ideal way, which would be to require the Euclidean norm of the difference vector to decrease by $\frac{1}{8}$; instead we require that the largest component of the difference vector decrease by at least $\frac{1}{8}$, which is equally effective in insuring convergence, works for any order n, and requires less computation and less registers.

Satisfying test (22b) then has the effect of roughly bounding the truncation error and the discontinuity error in such a way that the *accumulated* error in integrating a standard distance Δx (which we take equal to 1) is independent of the elementary step-lengths used and about equal to β^{-e}. In effect, instead of having to specify the elementary step-lengths to be used, the programmer tells the computer he wants the eth digit in y to be correct after integrating a unit distance along the x axis and the computer is expected to choose the elementary intervals to achieve this result most economically. Note, however, that in this connection the discussion of preservation of significance for unstable equations given at the beginning of Section 7 must be kept in mind.

Test (22b) is derived from equation (17) by the following rough argument. We divide the interval $(x_0, x_0 + 1)$ into subintervals in such a way that within each subinterval h is constant. Then summing (17) over the kth subinterval gives:

$$\mathcal{E}_k \equiv \sum_{x=x_k}^{x_{k+1}} E_t(x) \cong \frac{h^6}{70} \int_{x_k}^{x_{k+1}} y^{VII}\, dx \cong \frac{h^6}{70} (y_{k+1}^{VI} - y_k^{VI}) \tag{23}$$

Now the computation provides an estimate of h^6y^{VI}, namely, $h[f(x + h) - f^p]$, as may be deduced from the 6th equation of (A4). We use this estimate to bound h^6y^{VI} for all x by requiring satisfaction of test (22b) in every elementary step. Thus $h^6 \mid y^{VI} \mid \leqq \beta^{-e}$ and the accumulated error is, roughly speaking, bounded by

$$\left| \sum_k \varepsilon_k \right| \leqq \frac{\text{no. of subintervals}}{70} \cdot \beta^{-e} \tag{24}$$

which is not likely to be much greater than β^{-e}. We see also that the general effect of bounding h^6y^{VI} is to cause each part of the total integration interval to contribute to the error in proportion to its length, which tends to minimize the total number of steps to achieve a given accumulated error. The argument is necessarily somewhat crude, for we cannot do what one would ideally like to do, namely, bound h^6y^{VII}, because there is no estimate of it available (without increasing the degree of the method). Test (22b) also bounds the discontinuity error, equation (19), for a discontinuity if f clearly appears directly in $[f - f^p]$, so that bounding $h[f - f^p]$ just bounds (19).

In addition to availability of an estimate there is a further practical reason for formulating test (22b) in just the way shown, at least in a fixed point arithmetic operation, namely, that it permits the widest possible range of choices of h without either member of the inequality falling outside register range. If one wants to integrate across large discontinuities of f and still be free to demand accuracy of the order of the least count, it is clear from (19) that h must be reducible to or near the least count; on the other hand, for maximum size steps when f varies slowly and smoothly h must be increasable to or near the greatest count of the register. In practice the author has had the interval vary all the way from 2^{-2} to 2^{-39} in a 39 binary digit machine.

In the main then, the interval is selected by requiring it to be the largest interval satisfying both tests (22a) and (22b). However, four minor modifications of this basic rule are introduced in order to improve the usefulness and efficiency of the method and the smoothness of the automatic interval control, as follows:

Since the programmer cannot predict what intervals will be used he is given the privilege of specifying a maximum interval h_0, so that he has assurance that the solution will be available at least at the points $x_0 + (\text{integer})h_0$. The automatic interval control then includes a feature preventing an increase in the interval whenever such increase would result in skipping over one of the above points $x_0 + (\text{integer})h_0$.

Next, when any considerable amplitude of shock excitation has occurred it seems best, judging from Table 1, to choose the interval at the onset of the shock, then leave it unchanged until the transient due to the shock has subsided. In fact, if the interval is changed while the strong transient is still present this interval change itself results in a new shock excitation, and the interval control tends to become erratic in the sense that the interval is reduced too much and for too long, a phenomenon which the author has observed experimentally. The interval control itself contains feedback loops, we may say, which can cause erratic behavior, although not genuine instability because the computer takes refuge in *reducing* the interval in response to any uncomfortably large disturbance. The main rule, if not modified, leads to just such behavior because, as we see from the last column of Table 1

the change in d is 4 to 6 times greater in steps subsequent to the first step after onset of the disturbance than in the first. To avoid this misbehavior the computer is programmed to recognize the characteristic Δ, -4Δ, $+6\Delta$, $-4\Delta \cdots$ pattern and to leave the interval unchanged on the 2nd, 3rd and 4th steps provided they conform to this pattern within certain tolerances. This effectively prevents the interval control from interfering with the expeditious elimination of transients and results in preserving the ideal accuracy and speed represented by Table 1.

Another form of undesirable interference from interval control occurs in connection with reversal. Suppose that reversal has just occurred and that test (22b) is dominant in determining the interval, as it often will be. From Figure 2 we see that just after reversal d stays constant for 4 steps. This means that (22b) will be oversatisfied and the interval will be increased, whereas it should clearly not be increased since we are retracing steps for which the interval was presumably already correctly chosen earlier. The subsequent behavior would involve an unusually large shock when the "slack" in d is eventually "taken up" and an unnecessarily large interval decrease, again a phenomenon the author has observed in practice. The remedy for this misbehavior is simple: we program in a rule preventing interval increase for the first four steps after any reversal.

Finally a rather interesting type of misbehavior can occur when f tends toward a constant or indeed toward any 4th degree polynomial after an earlier more violent behavior which required a small interval. In these circumstances we want and expect the interval to increase rapidly, but if the parameter β^{-e} of test (22b) is very small, say only a few times the least count, then such increase may be prevented entirely by persistent roundoff noise in the "remembered" quantities. If f tends asymptotically to a 4th degree polynomial d should tend to a constant and $[f(x+h) - f^p]$ should tend to 0. What happens then is that so long as roundoff noise persists either (22b) is barely satisfied and the interval is not increased, or if (22b) is oversatisfied and an interval increase is attempted the roundoff noise in d is magnified by a factor β^4 according to (13) and causes test (22b) to fail on the next step. Now, unless special measures are taken, the roundoff noise can indeed persist and prevent interval increase indefinitely. Thus we may get into (and the author has actually got into) the absurd situation of taking 4000 steps to integrate $\frac{dy}{dx} = 0$ from $x = \frac{1}{2}$ to $x = 1$ (provided f was non-zero for $x < \frac{1}{2}$). The remedy for this mis-

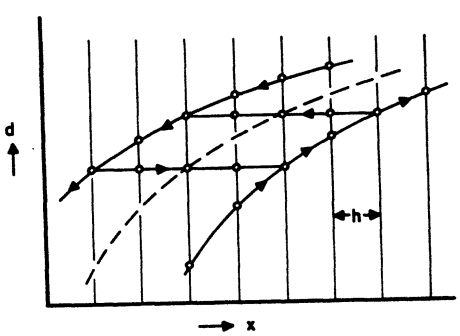

FIG. 2.—Hysteresis behavior of the "remembered" quantity d. The dashed curve is the true value of $d(x)$; the solid curves show the behavior of d in the computation.

behavior is not modification of the rules for interval choice, but a peculiar, carefully chosen rounding procedure for the multiplications by $A, \cdots D$ involved in the working equations so as to guarantee that roundoff noise will disappear in a finite (and minimum) number of steps just as other transients must and do because of the stability. The discussion of choice of rounding is rather long and is also of interest for other iterative procedures in numerical analysis, therefore, it is given separately in Appendix C.

The main rule amended by the four modifications just described provides a stable, non-erratic and generally reasonable behavior of the interval size in all cases which have been investigated, and the cases investigated were purposely chosen extremes in which the interval had to vary rapidly and widely. The interval still does not increase as fast when it should increase as it decreases when it should decrease, but this is hardly avoidable since both the finite rate of clearing of transients and the requirement of not skipping over the points $x_0 +$ (integer)h_0 act to delay interval increase.

If, when h has been reduced to the least count, test (22b) still fails, a programmed stop is encountered. Almost any major malfunction of program such as overflow in the computation of derivatives or elsewhere leads quite immediately to this programmed stop because of the extreme sensitivity of test (22b).

9. Automatic Starting. The essential idea which makes automatic starting feasible is that if we set off with entirely abnormal values of a, b, c, d, say putting 0 for each of them in the absence of any evidence as to their normal initial values, then upon integrating a few steps they will assume approximately their normal values *if* the stability is sufficiently good. Such a method of starting has the advantage of using mostly the normal integrating program, which has to be supplied in any case, requires very little extra programming of special nature, is of use only during starting. Since a modern computer can execute at least about one step per second in even rather complicated differential equation problems, the start can be accomplished blunder-free and accurately in a matter of seconds or at most minutes.

Several complications must be dealt with in providing a satisfactory automatic start: the proper interval for the first step forward from x_0 is not known in advance any more than are a, b, c, d. There is a certain degree of incompatibility between automatic starting and interval changing since the starting essentially involves eliminating a very large transient and, as we saw in the preceding section, changing the interval during a large transient can lead to erratic interval behavior. In any case, application of test (22b) during the first few steps of the starting process would be meaningless since the test was derived on the assumption that $a \cdots d$ had nearly normal values; this is illustrated by the fact that when a, b, c, d are zero the quantity $[f(x + h) - f^p]$ is $0(h)$, not $0(h^5)$ as it normally is. Finally, although one would ideally like to use points to the left of x_0 for starting, corresponding to fitting a polynomial to the left of x_0 and thus obtaining what we have called normally lagging values of $a(x_0) \cdots d(x_0)$, this cannot be done because it would imply that f is defined to the left of x_0, as it may not be.

The detailed schedule of the starting procedure will now be described and in the process the way in which the complications listed above are dealt with will become clear. The overall objective of the starting procedure is to fit a 5th degree poly-

nomial for y to the points x_0, $x_0 + h/2$, $x_0 + h$, $x_0 + 3h/2$, $x_0 + 2h$, thus determining $a(x_0)$, $b(x_0)$ $\cdots$ $d(x_0)$, where h is the correct interval (also to be determined) for the first step $x_0 \to x_0 + h$.

First we set the initial values $y(x_0) = y^0$ aside for safekeeping, set $a \cdots d$ equal to zero and do a tentative step forward $x_0 \to x_0 + h_0$, where h_0 is the maximum interval permitted. Test (22a) (but not (22b)) may now be applied since its operation is essentially independent of whether $a \cdots d$ have their normal values. If (22a) fails the interval is reduced, the beginning values at x_0 are recovered and a shorter tentative step forward from x_0 is taken, the program used here being just the same as in normal integration. This process continues until an h has been found which satisfies (22a).

When (22a) has been satisfied three more steps forward are taken, followed by a reversal and four steps back to x_0, all eight steps being taken at a constant interval. The reason for taking just four steps either way is that it provides just enough information to determine a 5th-degree polynomial.

We are now back at x_0 with a value of y somewhat in error but with first approximations for $a \cdots d$ which are already good to a fraction of a percent because of the high degree of stability of the method. The correct value of $y(x_0)$ is reinserted, the sense of integration again changed to forward and another four steps forward and four steps back to x_0 are taken, all at the same constant interval.

During the last backward step listed (the 16th step of the starting process) test (22b) is activated, for now the quantities $a \cdots d$ are so nearly normal that this test is significant. Test (22b) must be made neither too early during the starting process, for then $[f(x + h) - f^p]$ is not yet $0(h^5)$; nor too late, for as the process of integrating four steps back and forth is continued, $[f(x + h) - f^p]$ tends to *zero* in any case (refer to the hysteresis behavior of d described in Section 6). Thus there is a sort of psychological moment for doing test (22b) during the starting process. The author has found by "experimental mathematics" that $[f - f^p]$ is 2 to 3 times larger on the 16th starting step, for all equations and all h's, than it is in the ultimate normal integration process. Thus, applying the test at this point results in a slightly conservative initial choice of h.

If (22b) is not satisfied the interval is reduced and we go back to the very beginning of the starting process. If (22b) is satisfied, y^0 is reinserted, the sense of integration is changed to forward and the starting process may be considered almost completed. In fact, for all cases except those with very high accuracy requirements and very unstable equations the above process provides a satisfactory start. In the exceptional cases one can do a little better, typically a factor of six in the initial truncation error, by extending the starting schedule to include four more steps forward and four back at *half* the interval eventually to be used (making 24 starting steps in all after both tests are satisfied) and we actually take these extra eight steps in order to be quite sure that errors attributable to starting are less than the normal running truncation error. More precisely, after test (22b) is satisfied during the 16th starting step, we reinsert y^0, change the sense to forward, halve h, integrate forward four steps, reverse, integrate back four steps to x_0, reinsert y^0, change the sense to forward, double h and *now* regard the starting process as complete.

The chief effect of performing the last eight starting steps at a reduced interval is to reduce the amount of lead in a, b, c, d, which is beneficial because they should

lag, as they would if we had used a backward fitted polynomial. In any event, the truncation error in the first step after starting as above is less than normal.

Note that we have avoided ill effects due to changing interval during a transient by insisting that if any starting step at all is taken with a given interval h, at least eight are taken without changing h.

The transients during the early stages of the starting process are often large enough to cause overflow of the computer registers, and it is interesting to observe that such overflow will do no harm, for test (22b) is a very sensitive test and will almost certainly be violated if there are any previously occurring overflow errors. When this test is violated the computer simply discards all its previous computations, including any overflow errors, and starts afresh with reduced interval. The author has observed this effect many times, always without ultimate consequences. Persistent overflow caused by incorrect scaling of x, y or f is of course another matter, but one which comes to light very quickly in the form of the programmed stop mentioned earlier.

10. Test Problems Done by This Method. The differential equation problems used to develop the program and to rectify programming errors were those for the sine function and the exponential function. The normal truncation error for these "well-behaved" problems was found to agree with (18).

A test problem to exercise the automatic variable interval feature thoroughly and to verify the behavior for discontinuous f was then devised as follows:

$$\frac{dy}{dx} = \begin{cases} 0 & \text{for} \quad |x - \frac{1}{2}| \geqq 2^{-31} \\ 2^5 & \text{for} \quad |x - \frac{1}{2}| < 2^{-31} \end{cases} \tag{25}$$

to be integrated from 0 to 1 with h_0 specified as 2^{-8} and β^{-e} specified as 2^{-34}. This involves having the computer search the x-axis efficiently for an extremely narrow region in which $f \neq 0$, finding the area under the curve in this narrow region very

TABLE 2

Steps	h	x	$y \cdot 2^{20}$	"*Correct*" $y \cdot 2^{20}$
0	2^{-8}	0	0	0
157	2^{-38}	$1/2 - 2^{-31}$	0	0
169	2^{-32}	1/2	.015 564	.015 564
176	2^{-38}	$1/2 + 2^{-31}$	.031 149	.031 128
370	2^{-8}	1	.031 128	.031 128

TABLE 3

Steps	h	x	$y \cdot 2^{20}$	*Error*
0	2^{-8}	$-1/2$	0	
202	2^{-31}	-2^{-30}	.098 177	.000 002
214	2^{-36}	0	.196 352	.000 003
227	2^{-32}	2^{-30}	.294 527	.000 003
505	2^{-8}	1/2	.392 700	.000 001

accurately and then searching the rest of the x-interval at high speed again. The performance on problem (25) is shown in Table 2.

The "correct" y means the exact area of the figure obtained by joining the consecutive pair of points sampled, with $h = 2^{-38}$, at which f changes, by a straight line. The interval actually increased 64-fold temporarily between corner and center of the curve. The somewhat slower recovery of the interval on the increasing-interval side is exhibited in the difference between 194 steps from $x = 1/2 + 2^{-31}$ to $x = 1$ and 157 steps from $x = 0$ to $x = 1/2 - 2^{-31}$. The recovery of the interval to $h_0 = 2^{-8}$ *at all* is evidence that roundoff noise does not persist in the "remembered" quantities.

A test problem similar to the above with a very narrow but smooth analytic curve was also treated:

$$\frac{dy}{dx} = 2^7 \frac{(2^{-30})^2}{x^2 + (2^{-30})^2} \tag{26}$$

to be integrated from $x = -1/2$ to $x = +1/2$ with h_0 specified as 2^{-8} and β^{-e} specified as 2^{-32}. The result of this computation is given in Table 3.

The interval evidently did not have to decrease so much in this case because of the smoother curve to be integrated. The same comments in regard to increasing h apply here as in the previous example. The accumulated error is much less than 2^{-32} because of the simple symmetrical character of the curve being integrated.

Next, a typical unstable differential equation was treated:

$$\frac{dy}{dx} = \frac{20y}{x}; \qquad y^0 = \tfrac{1}{2} \quad \text{at} \quad x_0 = \tfrac{1}{2} \tag{27}$$

to be integrated from $x = 1/2$ to $x = 1$ with $h_0 = 2^{-4}$ and $\beta^{-e} = 2^{-25}$. Results are given in Table 4.

This illustrates the quality of the starting process in keeping the early truncation error small, a very important consideration in this case because such early errors are ultimately magnified one millionfold. Six significant decimals are preserved correct through 63 steps, in each of which the solution increases by 25 per cent on the average. The final error exceeds 2^{-25} because significance cannot increase.

Each of the above tests required only 3 to 15 seconds of computer time, and some sort of longer test seemed appropriate. As such the author chose Bessel's differential equation of order 16, and in particular, to find $J_{16}(z)$ by integrating from $z = 6$ to $z \cong 6000$. In this range the function begins very small, increases monotonically and rapidly over 200,000-fold, and then makes almost 1000 complete oscillations. We put $z = 2^{13}x$ and $h_0 = 2^{-13}$ and $\beta^{-e} = 2^{-23}$ and 2^{-28} respectively, for two tests. Tables 5 and 6 show the results of this computation.

TABLE 4

Steps	x	y	*Error*
0	.50	.000 000 476 837	
1	.507 812 5	.000 000 650 187	$.0^{11}1$
63	1.0	.500 000 546 694	.000 000 55

TABLE 5
($\beta^{-e} = 2^{-23}$)

Steps	z	$J_{16}(z)$	*Error*	$J'_{16}(z)$	*Error*
0	6.0	$.0^{51}$ 201 950		$.0^{52}$ 986 480	
1	6.125	$.0^{51}$ 633 713	$.0^{111}$	$.0^{53}$ 963 765	$.0^{111}$
77	16.0	.177 453 370	$.0^{50}$ 177	.062 487 955	$.0^{50}$ 066
93	18.0	.261 082 210	$.0^{50}$ 266	.003 519 524	$.0^{50}$ 005
109	20.0	.145 179 990	$.0^{50}$ 150	−.116 956 059	$-.0^{50}$ 118
49 005	6132.0	.004 126 972	$-.0^{53}$ 498	.009 311 583	
49 021	6134.0	.006 748 858	$-.0^{50}$ 809	−.007 627 018	
49 037	6136.0	−.009 741 657	$.0^{54}$ 174	−.002 961 758	
49 053	6138.0	.001 359 819	$-.0^{52}$ 666	.010 089 144	

TABLE 6
($\beta^{-e} = 2^{-28}$)

Steps	z	$J_{16}(z)$	*Error*	$J'_{16}(z)$	*Error*
0	6.0	$.0^{51}$ 201 950		$.0^{52}$ 986 480	
1	6.125	$.0^{51}$ 633 713	$.0^{111}$	$.0^{53}$ 963 765	$.0^{111}$
99	16.0	.177 453 297	$.0^{50}$ 104	.062 487 925	$.0^{50}$ 036
126	18.0	.261 082 096	$.0^{50}$ 152	.003 519 520	$.0^{50}$ 001
156	20.0	.145 179 923	$.0^{50}$ 083	−.116 956 010	$-.0^{50}$ 069
98 700	6132.0	.004 130 418	$-.0^{50}$ 052	.009 314 069	
98 741	6134.0	.006 749 685	$.0^{50}$ 018	−.007 631 186	
98 773	6136.0	−.009 745 792	$.0^{50}$ 039	−.002 960 774	
98 805	6138.0	.001 362 434	$-.0^{50}$ 051	.010 092 495	

Some of the properties of the automatic interval control are well illustrated by these two tables. In spite of our asking for less than full register accuracy, the computer starts accurately enough and with a small enough interval in both cases so that the initial truncation error is half the least count, for it recognizes via test (22a) that early errors may be magnified by the instability of the differential equation itself. The ultimate error is somewhat but not much larger than asked for, as it must be expected to be because of significance considerations. The interval is halved over most of the range and the error drops by just about 2^{-6} as between Table 5 and Table 6 (due allowance being made for the change of phase of the error between the two calculations). In the calculation of Table 6 we end up with almost as many correct significant figures as were given initially. A further increase in e (and in computing time) would presumably improve the preservation of significance a little more.

We emphasize that the above treatment of the Bessel equation is not claimed to be a good way of calculating Bessel functions, but was chosen purposely to illustrate how the method handles a rather "ill-behaved" problem.

Appendix A. The working formulas and truncation errors for degrees $m = 2$ through 6 are collected here.

$m = 2$

$$
\text{(A1)}\quad
\begin{aligned}
y(x+h) &= y(x) + h\{f(x) + a(x) + \tfrac{5}{12}[f(x+h) - f^p]\}\\
f^p &= f(x) + 2a(x)\\
a(x+h) &= a(x) + \tfrac{1}{2}[f(x+h) - f^p]
\end{aligned}
$$

$$E_t = 1 \cdot \frac{h^4}{4!} y^{IV}$$

$m = 3$

$$
\text{(A2)}\quad
\begin{aligned}
y(x+h) &= y(x) + h\{f(x) + a(x) + b(x) + \tfrac{3}{8}[f(x+h) - f^p]\}\\
f^p &= (x) + 2a(x) + 3b(x)\\
a(x+h) &= a(x) + 3b(x) + \tfrac{3}{4}[f(x+h) - f^p]\\
b(x+h) &= b(x) + \tfrac{1}{6}[f(x+h) - f^p]
\end{aligned}
$$

$$E_t = (25/6)\frac{h^5}{5!} y^{V}$$

$m = 4$

$$
\text{(A3)}\quad
\begin{aligned}
y(x+h) &= y(x) + h\{f(x) + a(x) + b(x) + c(x) + \tfrac{251}{720}[f(x+h) - f^p]\}\\
f^p &= f(x) + 2a(x) + 3b(x) + 4c(x)\\
a(x+h) &= a(x) + 3b(x) + 6c(x) + \tfrac{11}{12}[f(x+h) - f^p]\\
b(x+h) &= b(x) + 4c(x) + \tfrac{1}{3}[f(x+h) - f^p]\\
c(x+h) &= c(x) + \tfrac{1}{24}[f(x+h) - f^p]
\end{aligned}
$$

$$E_t = (27/2)\frac{h^6}{6!} y^{VI}$$

$m = 5$

$$
\text{(A4)}\quad
\begin{aligned}
y(x+h) &= y(x) + h\{f(x) + a(x) + b(x) + c(x) + d(x) + \tfrac{95}{288}[f(x+h) - f^p]\}\\
f^p &= f(x) + 2a(x) + 3b(x) + 4c(x) + 5d(x)\\
a(x+h) &= a(x) + 3b(x) + 6c(x) + 10d(x) + \tfrac{25}{24}[f(x+h) - f^p]\\
b(x+h) &= b(x) + 4c(x) + 10d(x) + \tfrac{35}{72}[f(x+h) - f^p]\\
c(x+h) &= c(x) + 5d(x) + \tfrac{5}{48}[f(x+h) - f^p]\\
d(x+h) &= d(x) + \tfrac{1}{120}[f(x+h) - f^p]
\end{aligned}
$$

$$E_t = (863/12)\frac{h^7}{7!} y^{VII}$$

$m = 6$

$$
\text{(A5)}\quad
\begin{aligned}
y(x+h) &= y(x)\\
&\quad + h\{f(x) + a(x) + b(x) + c(x) + d(x) + e(x)\\
&\qquad + \tfrac{19087}{60480}[f(x+h) - f^p]\}\\
f^p &= f(x) + 2a(x) + 3b(x) + 4c(x) + 5d(x) + 6e(x)\\
a(x+h) &= a(x) + 3b(x) + 6c(x) + 10d(x) + 15e(x)\\
&\qquad + \tfrac{137}{120}[f(x+h) - f^p]\\
b(x+h) &= b(x) + 4c(x) + 10d(x) + 20e(x)\\
&\qquad + \tfrac{5}{6}[f(x+h) - f^p]\\
c(x+h) &= c(x) + 5d(x) + 15e(x)\\
&\qquad + \tfrac{17}{96}[f(x+h) - f^p]\\
d(x+h) &= d(x) + 6e(x)\\
&\qquad + \tfrac{1}{40}[f(x+h) - f^p]
\end{aligned}
$$

$$e(x+h) = e(x) + \tfrac{1}{720}\,[f(x+h) - f^{P}]$$

$$E_t = 513\,\frac{h^8}{8!}\,y^{VIII}$$

Appendix B. The flow chart (Figure 3) presented here is probably in terms which are general enough to apply to most stored-program computers. As shown, it pro-

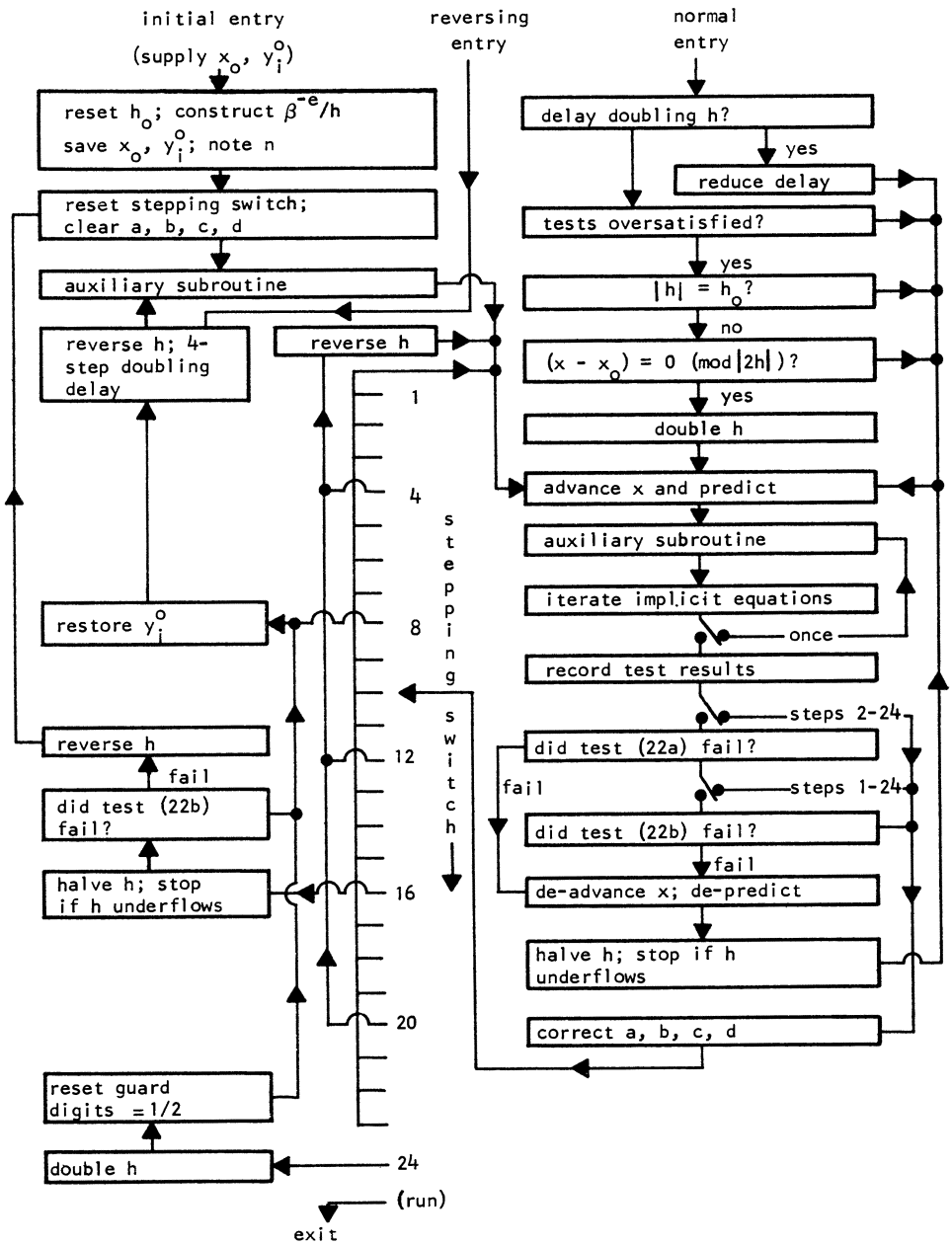

FIG. 3.—Flow Chart for one elementary step of integration.

vides for one elementary step of integration per entry into the routine, so that a master program can supervise the general course of the computation with complete flexibility. It also appeals to an "auxiliary subroutine" (closed) to calculate $f(x, y)$ given x and y, for complete flexibility as to what system of differential equations is being treated. The parameters which must be supplied are: the order n, the location of the auxiliary subroutine h_0, the accuracy parameter e, and the location of a working storage of $2 + 10n$ memory locations. The working storage contains a location for x_0, one for x and, for each $i(i = 1, 2, \cdots n)$, 10 locations containing respectively $y_i, f_i, a_i, b_i, c_i, d_i$, guard digits for y_i, f_i^p, y_i^p, and y_i at the beginning of the current step. The location normally containing guard digits is used for preserving the initial y_i^0 during starting. At the conclusion of the starting process this location is set to $\frac{1}{2}$ so that when a double precision increment-addition is made to y_i, the normally *rounded* y_i will appear in the first of the 10 registers for the use of the auxiliary subroutine.

The computing time per normal elementary step in this method is about $30n$ multiplication times ($21n$ milliseconds on the Illiac) plus twice the time required to calculate the derivatives. There are $6n$ actual multiplications performed, the remainder of the $30n$ being accounted for by additions and "housekeeping". Abortive integration steps, i.e. those partially done and then undone because of test failures, require only $2n$ actual multiplications but about $20n$ multiplication times plus twice the derivative calculation time. The starting process is clearly the equivalent in time consumed of not less than 24 normal step times.

These figures are for a computer without special address modification features, and the housekeeping time may be expected to be rather less where address modification features are available.

Appendix C. Here we discuss the choice of rounding procedures to guarantee against persistent noise induced by rounding, in an otherwise stable iterative arithmetic process, i.e. a process producing a convergent sequence when applied in the real number domain. Although we are not able to state a general recipe guaranteed to work in all cases, we can cite a qualitative principle which clearly always tends to improve the persistent noise behavior and which leads to a guaranteed solution in our particular problem of rounding the multiplications in (A4).

If multiplications and divisions are rounded in the normal way, namely, by replacing any number which is a fraction in terms of the least count, by the nearest integer in terms of the least count, we do *not* in general get the resulting sequence of integers converging to a unique limit, as can be seen in terms of a simple example. Consider the process $x_{n+1} = mx_n + b$, where the x's are real numbers and m and b are constants with $|m| < 1$. The sequence $\{x_n\}$ obviously converges and converges to $b/(1 - m)$. When iterative processes of this sort are done numerically the limiting value is not generally known in advance, the objective of the process being in fact usually to find the limiting value. Accordingly we reformulate the problem in such a way that b does not appear: let $y_n = x_n - x_{n-1}$, so that y_n obeys $y_{n+1} = my_n$ and tends to zero. Then we further reformulate so that the quantity eventually to be rounded in some sense, is the change in magnitude of y in an iteration. Spe-

cifically, we write

$$\text{(C1)} \qquad y_{n+1} = \pm y_n \mp (1 - |m|)y_n = \pm y_n \mp \mu y_n$$

according as $m > 0$ or $m < 0$. Observe that $0 < \mu < 1$.

The digital (integer) process corresponding to the real number process (C1) involves rounding the product μy_n to an integer according to some rule. Using an asterisk to denote a quantity integral in terms of the least count, we have for the digital process:

$$\text{(C2)} \qquad y^*_{n+1} = \pm(y_n^* - [\mu y_n^*])$$

where [ ] means some sort of rounding.

Normal rounding causes most of the sequences generated by (C2) to misbehave. If $\mu = \frac{1}{4} - \epsilon$ for example, then it is easy to verify that under normal rounding rules an initial $y_0^* = 0$ leads to the sequence $0, 0, 0, \cdots$; initial $y_0^* = 1$ leads to $1, \pm 1, 1, \pm 1, \cdots$; all other positive initial y_0^* lead to $2, \pm 2, 2, \pm 2 \cdots$; and similarly for negative initial y^*. This general sort of misbehavior is not peculiar to the value of μ chosen for illustration, but is typical of most μ's. In the formulation (C2), however, the source of the difficulty is easy to discern: it is merely that the term $[\mu y_n^*]$ normally rounded may often vanish when y_n^* does not, so that the magnitude of y_n^* may "get stuck" at a non-zero value.

The difficulty is entirely removed in this simple example by redefining the rounding process so that

$$\text{(C3)} \quad [x] \equiv \begin{cases} x & \text{for } x \text{ exactly integral} \\ \text{integer nearest } (x + \frac{1}{2}) & \text{for } x \text{ positive non-integral} \\ \text{integer nearest } (x - \frac{1}{2}) & \text{for } x \text{ negative non-integral} \end{cases}$$

We term this special kind of rounding "rounding away from zero," for it consists of moving the number x *away* from the origin just far enough to make it integral. So defined, $[\mu y_n^*]$ does not exceed y_n^* in magnitude, is of the same sign as y_n^* and does not vanish unless y_n^* vanishes. Thus, all integer sequences generated by (C2) must now converge to 0.

The general principle is accordingly that if we can formulate an iterative digital process so that the quantity to be rounded is a correction subtracted from the previous value of an integer variable intended to converge to zero, as in (C2), then the quantity to be rounded should be rounded generally away from zero. In more complicated cases where several integer variables are involved the correction (in the above sense) to each may be a function of all the variables; but still it should be rounded away from zero.

Our particular problem consists of rounding the multiplications $A[\ \ ]$, $B[\ \ ]$, $C[\ \ ]$, $D[\ \ ]$ in the working equations (A4). Suppose that f tends asymptotically to a constant and consider what may happen when $a \cdots d$ have become small. Then $f(x + h) - f(x)$ will cancel out of (A4) at some stage, and thereafter the relevant equations of the process will be:

$$
\text{(C4)}\quad
\begin{aligned}
a^*_{n+1} &= a_n^* + 3b_n^* + 6c_n^* + 10d_n^* \\
&\qquad - [(25/24)(2a_n^* + 3b_n^* + 4c_n^* + 5d_n^*)] \\
b^*_{n+1} &= b_n^* + 4c_n^* + 10d_n^* \\
&\qquad - [\ 35/72\ (2a_n^* + 3b_n^* + 4c_n^* + 5d_n^*)] \\
c^*_{n+1} &= c_n^* + 5d_n^* \\
&\qquad - [\ 5/48\ \ (2a_n^* + 3b_n^* + 4c_n^* + 5d_n^*)] \\
d^*_{n+1} &= d_n^* \\
&\qquad - [\ 1/120\ (2a_n^* + 3b_n^* + 4c_n^* + 5d_n^*)]
\end{aligned}
$$

where the asterisk signifies a quantity integral in terms of the least count, and the [ ] symbolizes rounding. Note that these equations are in just the form we require to apply the "rounding away from zero" principle, since the terms $3b_n^*$, $4c_n^*$ etc. are integral and have no effect on the behavior of the rounding.

Normal rounding in equations (C4) leads to persistent roundoff noise. The rounding process is so non-linear that we have no analytical theory and must work out specific numerical examples. Two examples of indefinitely persisting (cyclic) roundoff noise are:

n	a^*	b^*	c^*	d^*
0	1	0	0	0
1	−1	−1	0	0
2	1	1	1	0
3	1	1	0	0
4	−1	−1	−1	0
5	−1	−1	0	0
6	1	1	1	0
		etc.		

n	a^*	b^*	c^*	d^*
0	0	1	0	1
1	5	7	4	1
2	6	8	4	1
3	5	6	3	1
4	4	6	3	1
5	5	7	4	1
6	6	8	4	1
		etc.		

As we saw in Section 8, any behavior like this (and there are many cases of it) can frustrate the interval control in its attempts to increase the interval when the interval obviously ought to be increased. Curiously enough, the persistent cycles of roundoff noise contribute practically no error to y, for the contribution to y, averaged over a repetitive noise cycle, is no more than about $h/60$ times the least count. However, proper behavior of the interval control alone is enough reason for rectifying the roundoff behavior.

The simplest change in rounding which suggests itself is rounding all four multiplications in (C4) away from zero. However, such a simple remedy does not work, for it represents too drastic a modification of the fourth equation of (C4). It implies in fact that d^* must change unless $(2a^* + 3b^* + 4c^* + 5d^*)$ is zero, and persistent oscillation of d^* results inevitably. After some experimentation the author has concluded that the best rule is: round the first three multiplications in (C4) away from zero according to (C3), but for the fourth multiplication move the multiplicand $(2a^* + 3b^* + 4c^* + 5d^*)$ away from zero by 16 units and then multiply by $\frac{1}{120}$, rounding *normally*. The treatment of the fourth multiplication is a "partial" rounding away from zero or a less drastic modification of normal rounding, but

clearly in the same spirit. The rounding rules thus finally fixed upon will cause every initial quadruple of integers to converge to (0, 0, 0, 0), as was verified by letting the computer treat every case. Actually, all initial quadruples of integers between -2 and 2 inclusive were examined, and all tend to (0, 0, 0, 0). The average number of steps to arrive at (0, 0, 0, 0) is $\frac{13}{2}$ and the maximum is 14. If we move the multiplicand of the last multiplication only 12 units instead of 16, one persistent cycle appears. If we move it 14, 16, respectively 18 units all quadruples converge to (0, 0, 0, 0) but the average number of steps to clear begins to increase. Thus 16 seems a safe compromise.

These principles may be of help in deciding how to round the arithmetic in other iterative digital processes, such as solving systems of implicit equations. In our present state of knowledge of the subject a certain amount of experimenting of the sort described above will probably have to be done in every individual case more complicated than the one-variable case. The general reason for stabilizing roundoff noise in these ways is to improve the functioning of tests-for-end, for such tests are subject to the same difficulties as test (22b) in our procedure.

University of Illinois
Urbana, Illinois

1. W. E. Milne, *Numerical Solution of Differential Equations*, John Wiley & Sons, New York, 1953.
2. L. Collatz, *Numerische Behandlung von Differentialgleichungen*, Springer-Verlag, Berlin, 1955.
3. H. Rutishauser, "Über die Instabilität von Methoden zur Integration gewöhnlicher Differentialgleichungen," *Z. Angew Math. Phys.*, v. 3, 1952.
4. E. Fehlberg, "Numerically stable interpolation formulas with favorable error propagation for first and second order differential equations," NASA Technical Note D-599, March 1961.