A Polyalgorithm for the Numerical Solution of Ordinary Differential Equations

G.D. BYRNE
University of Pittsburgh
and
A.C. HINDMARSH
Lawrence Livermore Laboratory, University of California

Two variable-order, variable-step size methods for the numerical solution of the initial value problem for ordinary differential equations are presented. These methods share a common philosophy and have been combined in a single program. The two integrators are for stiff and nonstiff ordinary differential equations, respectively. The former integrator is based on backward differentiation formulas of orders one through five, each of which is stiffly stable, while the latter is based on formulas of Adams-Moulton type of orders one through twelve. Both use a Nordsieck history array; generating polynomials to compute coefficients; and Milne-like estimates of the principal part of the local truncation error to control error, step size, and order. Some numerical results are given.

Key Words and Phrases: ordinary differential equations, initial value problem
CR Categories: 3.20, 5.16, 5.17

1. INTRODUCTION

We are concerned with the numerical solution of the ordinary initial value problem

$$\begin{aligned} \dot{Y}(t) &= F(Y(t), t), \quad t \in [a, b] \\ Y(a) &= Y_0 \end{aligned} \tag{1.1}$$

where $\dot{Y}(t) = dY(t)/dt$, $Y(t) = [Y^1(t), Y^2(t), \ldots, Y^N(t)]^\mathsf{T}$, $F(u, t) = [F^1(u, t), \ldots, F^N(u, t)]^\mathsf{T}$, and the superscript T denotes transpose. The existence of a unique, continuous solution is assured if there is a constant K such that for some vector norm $\|\cdot\|$ the Lipschitz condition

$$\| F(u_1, t) - F(u_2, t) \| \leq K \| u_1 - u_2 \| \tag{1.2}$$

A version of this paper was presented at Mathematical Software II, a conference held at Purdue University, West Lafayette, Indiana, May 29–31, 1974.
This work was performed under the auspices of the U.S. Atomic Energy Commission.
Authors' addresses: G.D. Byrne (present), Applied Mathematics Division, Argonne National Laboratory, Argonne, IL 60439; A C. Hindmarsh, Numerical Mathematics Group, L-310, Lawrence Livermore Laboratory, P.O. Box 808, Livermore, CA 94550.

holds for (u_1, t), $(u_2, t) \in \mathbf{R}^N \times [a, b]$, and if F is continuous on $\mathbf{R}^N \times [a, b]$. We note that weaker conditions can be imposed [6, 20]. Problems of type (1.1) arise in the analysis of electrical networks, chemical systems, and mechanical systems, as well as in the solution of control problems and in the solution of parabolic partial differential equations by the method of lines [2, 5, 29].

The methods to be treated here are discrete-variable methods. That is, approximations to the exact solution Y are provided only at certain (discrete) points. Step sizes h_i will be introduced, and if we let $t_n = a + \sum_{i=1}^{n} h_i$, then we produce approximations y_n to $Y(t_n)$ for $n = 1, 2, \ldots$. The methods are aimed at two types of ordinary differential equations (including systems of such equations), stiff and nonstiff equations.

Stiffness in ordinary differential equations is difficult to characterize precisely. While attempts have been made to define the concept formally [11], we refrain from doing so here. (See also [13, pp. 207–217; 29, pp. 267–270].) However, it is typical of stiff ordinary differential equations that the Jacobian matrix $J(u, t)$, whose element in row i and column j is

$$J^{ij}(u, t) = (\partial F^i/\partial Y^j)(u, t), \tag{1.3}$$

has one or more eigenvalues whose real parts are negative and large in modulus. As a result, the solution has one or more terms which decay very rapidly in comparison to other terms present. A simple stiff system is [13, p. 209]

$$\dot{Y}(t) = \begin{bmatrix} 998 & 1998 \\ -999 & -1999 \end{bmatrix} Y(t), \qquad Y(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{1.4}$$

for which the exact solution is

$$Y(t) = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} \exp(-t) \\ \exp(-1000t) \end{bmatrix},$$

and the eigenvalues of the Jacobian are -1000 and -1. An example with $N = 1$ is the following diurnal chemistry problem:

$$\begin{aligned} \dot{Y}(t) &= \dot{H}(t) - B[Y(t) - H(t)], \quad 0 \le t \le T, \\ Y(0) &= D/B, \\ H(t) &\equiv [D + AE(t)]/B, \\ E(t) &= \begin{cases} 0, & \sin \omega t \le 0, \\ \exp(-C\omega/\sin \omega t), & \sin \omega t > 0, \end{cases} \end{aligned} \tag{1.5}$$

with solution $Y(t) = H(t)$. The values of the constants which arose from a study of the oxygen singlet [10] are $A = 10^{-18}$, $B = 10^{8}$, $C = 4$, $D = 10^{-19}$, $\omega = \pi/43{,}200$, and $T = 432{,}000$. For a discussion of stiffness and its relation to numerical methods, see [13, pp. 209–223]. It should be noted that stiff differential equations frequently arise from the use of the method of lines, analysis of electronic networks, the design of chemical reactors, etc. [2, 3, 29, 30, 31, 41].

The stiff method described here was designed for several reasons. A large atmospheric modeling problem had led to a nonlinear vector-valued parabolic partial differential equation (a diffusion-convection problem) [5]. The method of lines had been applied to obtain a system of the type (1.1) with N on the order of a few to several thousand, $b - a \sim 10^{10}$, and

$$\max\{|\operatorname{Re}(\lambda_i)|\}/\min\{|\operatorname{Re}(\lambda_i)|\} \sim 10^8,$$

where the λ denote eigenvalues of the Jacobian matrix (1.3) of F. This quotient is, in some sense, a measure of the stiffness of (1.1): the larger the quotient the stiffer the system, if it is understood that for at least some λ_i, $\operatorname{Re}(\lambda_i) < 0$. This type of equation was to be solved repeatedly. Thus, in light of much recent testing on automatic integrators [8, 11, 26, 42], a Lawrence Livermore Laboratory version of Gear's method, due to Hindmarsh [22], was used to solve the problem. It was observed that certain instabilities arose; e.g. the order and step size selected would occasionally be decreased drastically, then quickly built up again. For the diurnal chemistry problem, which was related to the atmospheric model, and of which (1.5) is a mockup, Gear's method failed altogether.

Some recent work [3, 17, 18, 47] has indicated the source of these instabilities. Gear's methods, when operating at order q, require that $q + 1$ steps be taken at a fixed step size h. Then, if it is deemed appropriate, the step size is changed to a new value h'. The new data required to continue are obtained from the existing data by interpolation. Thus the methods, while allowing the step size to vary throughout the problem, are basically fixed-step methods. (See also [12] for another fixed-step interpolation method.)

An alternative approach to changing step sizes developed in recent years [3, 27, 28, 32, 38, 40, 42, 43] is based on an alternate interpolating polynomial which underlies most of the linear multistep formulas. If instead of requiring that the interpolation knots be equally spaced, it is required that they be distinct but otherwise arbitrarily spaced, then the step size can be selected more or less arbitrarily. Such methods are called variable-step methods. It has been proved theoretically and demonstrated empirically that the variable-step Adams-type methods are more stable than the fixed-step Adams methods with interpolatory step changing [17, 18, 47]. For stiff methods based on backward differential formulas, there is computational evidence that this result holds there as well. (Here the stability property in question means that errors in initial data do not lead to unbounded errors as the integration proceeds.) The instability of the fixed-step approach is further aggravated by the possibility that the step size may be changed (e.g. because of error controls) at any time during the $q + 1$ steps for which it is intended to be fixed. The variable-step methods involve no such ambivalence in step change strategy.

The stiff method described in Section 2 is a variable-step method based on backward differentiation formulas. It bears similarities to that in [3], but there are several important differences. One of these is in the adoption of a Nordsieck type of history array for the storage of past data.

The nonstiff method described in Section 3 grew out of a need for a program having a broad scope of applicability, as was the case for the programs of Gear [13] and Hindmarsh [22]. Thus the Adams formulas were generalized to variable-step form, also with the use of a Nordsieck history array. Because they share many of the same features, the stiff and nonstiff methods were then easily combined into a single code, called EPISODE: *E*xperimental *P*ackage for *I*ntegration of *S*ystems of *O*rdinary *D*ifferential *E*quations. This code is an implementation of the concept of a polyalgorithm developed by Rice [39], in that it contains many algorithms for the same problem (1.1), from which the user can easily select the one most appropriate.

In Section 4 some numerical results are given for EPISODE to illustrate its effectiveness in comparison with the program in [22]. A summary is given in Section 5.

2. THE INTEGRATOR FOR STIFF ORDINARY DIFFERENTIAL EQUATIONS

This integrator is based on the use of the implicit backward differentiation formulas of orders one through five. Curtiss and Hirschfelder [9] used the lower order formulas; Henrici [21, pp. 206–209] gave the formulas for all orders; Gear [13, 14, 15] implemented them by using interpolation to change step size; Hindmarsh developed a revised version of Gear's programs, called the GEAR package [22, 23, 24]; and Brayton et al. [3] implemented a version which allows the step size to vary after each successful step. There are some features of the method in [3] which are unclear because of proprietary rights to it. There also appear to be some errors.

2.1 Predictor and Corrector Formulas

The fundamental idea is simple. Consider $N = 1$ and let π_{n-1} denote an interpolatory polynomial of degree q or less, $1 \leq q \leq 5$, which satisfies the $L \equiv q + 1$ conditions

$$\pi_{n-1}(t_{n-i}) = y_{n-i}, \; i = 1, 2, \ldots, q, \quad \text{and} \quad \dot{\pi}_{n-1}(t_{n-1}) = \dot{y}_{n-1} \equiv F(y_{n-1}, t_{n-1}). \tag{2.1}$$

Here y_j is an approximation to $Y(t_j)$, the exact solution of (1.1) evaluated at t_j, and $t_j = a + \sum_{i=1}^{j} h_i$. (The obtaining of the required past values will be discussed later.) Of course, h_i denotes the step size, need not be uniform, and satisfies $0 < \min\{h_i\}$, $\max\{h_i\} \leq H$. Thus, $a = t_0 < t_1 < \cdots < t_{\text{final}} = b$ defines a strict partition of $[a, b]$. The strategy is to construct the polynomial π_n, of degree q or less, which satisfies the $L + 1$ conditions

$$\pi_n(t_{n-i}) = y_{n-i}, \; i = 0, 1, \ldots, q, \quad \dot{\pi}_n(t_n) = \dot{y}_n \equiv F(y_n, t_n), \tag{2.2}$$

and, in the process, to compute a value of y_n which makes this possible. This set of conditions can be rephrased in the classical linear multistep form

$$h_n \dot{y}_n = -\sum_{i=0}^{q} \alpha_{ni} y_{n-i}, \qquad \dot{y}_n = F(y_n, t_n), \tag{2.3}$$

such that the solving of (2.3) for y_n is necessary and sufficient for the existence of π_n in (2.2). In this latter form, the present method is seen to generalize the backward differentiation formulas of Henrici and Gear, as is shown in [3]. However, we shall not use this formulation here, but rather the interpolatory formulation (2.2).

The solution of (2.2) begins with the extrapolation to t_n of π_{n-1}. Thus we define

$$y_{n(0)} = \pi_{n-1}(t_n), \qquad \dot{y}_{n(0)} = \dot{\pi}_{n-1}(t_n) \tag{2.4}$$

as first approximations to y_n and $\dot{y}_n$. However, the quantity

$$\dot{y}_{n(0)} - F(y_{n(0)}, t_n) \tag{2.5}$$

will not generally vanish, as it must for y_n. Thus it is used as a residual with which we can correct $y_{n(0)}$ and obtain $y_{n(1)}$. This correction process, to be described shortly, is repeated, until a sufficiently accurate estimate of y_n, which solves (2.2), is obtained.

For general N, the basic ideas given above are unchanged, but are applied to each of the N components of the system. The polynomials π_{n-1} and π_n are now vector valued.

Rather than store the vectors y_{n-i} and $\dot{y}_{n-1}$ called for in (2.1), we store an equivalent $N \times L$ array of data based on an invention of Nordsieck [36]. The array is written

$$z_{n-1} = [y_{n-1}, h\dot{y}_{n-1}, h^2\ddot{y}_{n-1}/2, \ldots, h^q y_{n-1}^{(q)}/q!] \tag{2.6}$$

with the understanding that the derivatives $y_{n-1}^{(j)}$ are not generally exact values of $Y^{(j)}(t_{n-1})$, but instead are the approximations $\pi_{n-1}^{(j)}(t_{n-1})$. Here h is $h_n = t_n - t_{n-1}$. These scaled derivatives are correct to within $O(H^{q+1})$ if the data in (2.1) is correct to that order.[1] The problem now becomes that of constructing z_n from z_{n-1}. For obvious reasons we will index the columns of these arrays from 0 to q rather than from 1 to L.

The first step of the algorithm consists of approximating z_n by the predicted array

$$z_{n(0)} = z_{n-1}A(q) \tag{2.7}$$

where $A(q)$ is the $L \times L$ Pascal triangle matrix with elements

$$A^{ij}(q) = \left\{\begin{matrix} 0, & i < j \\ \binom{i}{j} = i!/j!(i-j)!, & i \geq j \end{matrix}\right\}, \quad i, j = 0, 1, \ldots, q. \tag{2.8}$$

For example,

$$A(5) = \begin{bmatrix} 1 & & & & & \\ 1 & 1 & & & & \\ 1 & 2 & 1 & & & \\ 1 & 3 & 3 & 1 & & \\ 1 & 4 & 6 & 4 & 1 & \\ 1 & 5 & 10 & 10 & 5 & 1 \end{bmatrix}.$$

[1] If the data is exact, this result is given in [35, pp. 155–156]. The interpolation formula with remainder is $Y(t) = \pi_{n-1}(t) + R(t)$; $R(t) = P(t)\,[t, t_{n-1}, t_{n-1}, t_{n-2}, \ldots, t_{n-q}]$; $P(t) = (t - t_{n-1})^2 (t - t_{n-2}) \ldots (t - t_{n-q})$, in terms of Newton divided differences. For $j = 0, 1, \ldots, q$ and $t \geq t_{n-1}$ one has $R^{(j)}(t) = P^{(j)}(t)\, Y^{(q+1)}(\xi)/(q+1)!$ for some $\xi \in (t_{n-q}, t)$, provided $Y \in C^{q+1}$. Further, the Newton formula for π_{n-1} [35, pp. 2–3] shows that $h^j \pi_{n-1}^{(j)}(t)$ has a bounded dependence on each piece of data involved.

The columns of $z_{n(0)}$ are defined to be $h^j\pi_{n-1}^{(j)}(t_n)/j!$, $j = 0, 1, \ldots, q$, while the columns of $z_{n-1}A(q)$ are the Taylor series extrapolations to t_n of $h^j\pi_{n-1}^{(j)}/j!$, to $q + 1 - j$ terms. Thus (2.7) reduces to the statement that the $(d + 1)$ term Taylor series of a polynomial of degree d is the polynomial itself. The multiplication in (2.7) can be done with repeated additions only, as observed by Gear [13].

Consider now the corrected quantity y_n, the polynomial π_n associated with it by (2.2), and the corrected Nordsieck array

$$z_n = [y_n, h\dot{y}_n, \ldots, h^q y_n^{(q)}/q!], \quad y_n^{(j)} \equiv \pi_n^{(j)}(t_n). \tag{2.9}$$

While postponing the discussion of the algorithm for obtaining y_n, we now derive a fundamental relation between $z_{n(0)}$ and z_n. Define a polynomial of degree q or less by

$$\Delta_n(t) = \pi_n(t) - \pi_{n-1}(t). \tag{2.10}$$

Inspection of (2.1) and (2.2) yields

$$\Delta_n(t_{n-i}) = 0, \quad i = 1, 2, \ldots, q. \tag{2.11}$$

If we define a correction quantity

$$e_n = y_n - y_{n(0)} = \pi_n(t_n) - \pi_{n-1}(t_n), \tag{2.12}$$

then we have also $\Delta_n(t_n) = e_n$, and so Δ_n is given uniquely by

$$\Delta_n(t) = \prod_{i=1}^{q} [(t - t_{n-i})/(t_n - t_{n-i})]e_n. \tag{2.13}$$

We make a change of variables given by

$$x = (t - t_n)/h, \qquad \xi_i = (t_n - t_{n-i})/h \tag{2.14}$$

and define the scalar polynomial

$$\Lambda_n(x) = \prod_{i=1}^{q} (1 + x/\xi_i). \tag{2.15}$$

As a result we have

$$\Delta_n(t) = \Delta_n(t_n + hx) = \Lambda_n(x)e_n. \tag{2.16}$$

If we now compute the coefficients l_j in (2.15),

$$\Lambda_n(x) = \sum_{j=0}^{q} l_j x^j, \tag{2.17}$$

then we see from (2.7), (2.9), and (2.16) that for $j = 0, 1, \ldots, q$, column j of $z_n - z_{n(0)}$ is

$$h^j\pi_n^{(j)}(t_n)/j! - h^j\pi_{n-1}^{(j)}(t_n)/j! = h^j\Delta_n^{(j)}(t_n)/j! = \Lambda_n^{(j)}(0)e_n/j! = l_je_n.$$

That is, if we define the $1 \times L$ vector

$$l = [l_0, l_1, \ldots, l_q], \tag{2.18}$$

then

$$z_n = z_{n(0)} + e_nl. \tag{2.19}$$

This simple relation is used to correct the entire $z_{n(0)}$ array as soon as y_n is computed.[2] (See also [1], where essentially the same result is independently given, but for uniform step size.)

We note in passing some facts about the vector l. From (2.15) and (2.17) we obviously have

$$l_0 = 1, \quad l_1 = \sum_1^q \xi_i^{-1}, \quad l_q = 1/\xi_1\xi_2 \ldots \xi_q . \tag{2.20}$$

Also, since the partition $\{t_i\}$ is strict and $h = h_n$, it follows from (2.14) that $1 = \xi_1 < \xi_2 < \cdots < \xi_q$. As a result, all $l_j > 0$. Of course, l depends on n and q, and we will sometimes write $l_j(n)$ or $l_j(q)$ to show this. We compute l at every step, using (2.15) and (2.17), by building up the coefficients of the partial products in (2.15) in succession, leading to the two-term recursion relation $l_j(q) = l_j(q-1) + l_{j-1}(q-1)/\xi_q$.

We can now observe the following (somewhat surprising) fact. The algorithm given by (2.7) and (2.19), with y_n calculated as described above, is a qth-order method regardless of the values of the vector l, assuming only that $l_0 = 1$ and that l_1 is bounded away from zero. To see this, suppose that z_{n-1} has errors that are at most $O(H^L)$, relative to an exact solution $Y \in C^{q+1}$. If $z(t)$ is the corresponding exact Nordsieck array evaluated at t, and $\delta = [\delta_0, \delta_1, \ldots, \delta_q] = z_n - z(t_n)$, then from (2.19) we have

$$\delta = z_{n(0)} + e_n l - z(t_n) = e_n l + O(H^L),$$

because the predictor (2.7) is of order q. We then have $\delta_0 = e_n + O(H^L)$, and $\delta_1 = l_1 e_n + O(H^L) = l_1\delta_0 + O(H^L)$. But $\delta_1 = h\dot{y}_n - h\dot{Y}(t_n) = hF(y_n, t_n) - hF(Y(t_n), t_n)$, and so, using any suitable vector norm, and the Lipschitz condition (1.2), we obtain $\| \delta_1 \| \leq hK \| y_n - Y(t_n) \| = hK \| \delta_0 \|$. Combining these results, we have

$$\| l_1\delta_0 \| = \| \delta_1 \| + O(H^L) \leq hK \| \delta_0 \| + O(H^L),$$

or

$$\| \delta_0 \| \leq [1/(| l_1 | - hK)]O(H^L) = O(H^L),$$

as $H \rightarrow 0$. This proves that y_n, and in fact all of z_n, is accurate to order q.

2.2 Corrector Iteration

We come now to the calculation of y_n, and we start from the relation given by column 1 in (2.19),

$$h\dot{y}_n = h\dot{y}_{n(0)} + (y_n - y_{n(0)})l_1 . \tag{2.21}$$

[2] We do not use the Milne error estimate to extrapolate y_n or z_n to the next higher order of accuracy, for three reasons. First, extrapolation to order $q + 1$, while controlling the $O(H^{q+1})$ error for an order q approximation, seems inconsistent. Second, the extrapolation of an order q implicit backward differentation formulas does not yield the order $q + 1$ backward differentiation formula, as can be seen from the set of data involved. (For example, extrapolation of the implicit Euler method yields essentially the trapezoid rule.) Third, the effort required to extrapolate at the end of an order q step is greater than that of taking an order $q + 1$ step to begin with. See also [44], where extrapolation is recommended.

Here, $y_{n(0)}$ and $h\dot{y}_{n(0)}$ are the first two columns of (2.7), l_1 is also known (and is $-\alpha_{n0}$ in (2.3)), and $\dot{y}_n = F(y_n, t_n)$. We therefore define a mapping G from $\mathbf{R}^N$ to itself by

$$G(u) = (u - y_{n(0)}) - (h/l_1)[F(u, t_n) - \dot{y}_{n(0)}], \tag{2.22}$$

whose zero is the desired vector y_n. We compute this zero by a modified Newton iteration method, given by

$$u_0 = y_{n(0)}, \qquad P_\nu(u_{m+1} - u_m) = -G(u_m), \; m = 0, 1, \ldots . \tag{2.23}$$

Here P_ν is an $N \times N$ matrix evaluated at step $\nu = \nu(n) \leq n$ and given by

$$P_\nu = I - [h_\nu/l_1(\nu)]J_\nu . \tag{2.24}$$

The matrix J_ν is an approximation to the Jacobian (1.3) evaluated at the predicted value as of $t = t_\nu$:

$$J_\nu \approx J(y_{\nu(0)}, t_\nu), \tag{2.25}$$

and I denotes the $N \times N$ identity matrix.

At present, four different options are available to the user in (2.25). One is to use an analytic (user-supplied) value for the Jacobian. A second is an approximation computed internally with finite differences. A third, which is helpful when $J(u, t)$ is diagonally dominant, uses a diagonal approximation to the Jacobian in which the diagonal resembles a directional derivative of F with respect to y. In this case, the scalar h/l_1 in P_ν is updated at every step. Finally, the user can let J_ν be the null matrix, but this is not recommended for stiff problems. The first three iteration methods are called chord methods, by analogy with the one-dimensional picture. The first two require the solution of an $N \times N$ linear algebraic system (2.23), for which an LU decomposition is computed and saved. The fourth option is referred to as a functional iteration, because it reduces to iterative evaluation of the function $u - G(u) = y_{n(0)} + (h/l_1)[F(u, t_n) - \dot{y}_{n(0)}]$.

In any case, the matrix P_ν is used for $n = \nu, \nu + 1, \ldots$ until an update of it is forced by the failure of (2.23) to converge or by a direct detection of inaccuracy in P_ν. Corrector convergence is decided on the basis of successive differences $u_{m+1} - u_m$, in relation to a user-supplied error tolerance parameter.

We recognize that there are many more sophisticated methods for solving nonlinear algebraic systems, some of which are directly applicable to (2.21). Future development of EPISODE, and variants of it, is anticipated with this fact in mind. For the first option alone, variants are being developed in which special sparse structures for J are utilized (cf. [25]).

2.3 Error Estimation

The algorithm described so far is of little or no use without an accompanying algorithm for the selection of order q and step size h throughout the integration. This selection algorithm is based on estimates of the local discretization error.

We begin by defining this local error. In terms of the classical linear multistep form (2.3), we define the local error for the kth-order method to be

$$E_n(k) = Y_n + \frac{1}{\alpha_{n0}}\left[h\dot{Y}_n + \sum_{i=1}^{k} \alpha_{ni} Y_{n-i}\right], \tag{2.26}$$

where the Y_i and $\dot{Y}_n$ are *exact* values $Y(t_i)$ and $\dot{Y}(t_n)$ for the true solution $Y \in C^{q+1}$ of the differential equation in (1.1) with $Y(t_{n-1}) = Y_{n-1}$. The α_{ni} here are the coefficients for the order k method as in (2.3). We will let k be any of q, $q - 1$, or $q + 1$.

To compute (2.26) we will use instead the machinery of interpolatory polynomials. Define[3] a polynomial $\pi_{n,k}$ of degree k or less, by

$$\pi_{n,k}(t_{n-i}) = Y_{n-i}, \quad i = 0, 1, \ldots, k. \tag{2.27}$$

Then $h\dot{\pi}_{n,k}(t_n)$ is the estimated value of $h\dot{Y}_n$ given by the right-hand side of (2.3) with y_{n-i} replaced by Y_{n-i}, and q replaced by k. Therefore,

$$E_n(k) = [h\dot{Y}_n - h\dot{\pi}_{n,k}(t_n)]/\alpha_{n0}. \tag{2.28}$$

Now for a fixed value of k, we define also a polynomial $\pi_{n,k+1}$ of degree $k + 1$ or less, by

$$\pi_{n,k+1}(t_{n-i}) = Y_{n-i}, \; i = 0, 1, \ldots, k, \qquad \dot{\pi}_{n,k+1}(t_n) = \dot{Y}_n. \tag{2.29}$$

Then if $\Delta \equiv \pi_{n,k+1} - \pi_{n,k}$, we have

$$E_n(k) = h\dot{\Delta}(t_n)/\alpha_{n0}. \tag{2.30}$$

We also have $\Delta(t_{n-i}) = 0$, $i = 0, 1, \ldots, k$, and therefore

$$\Delta(t) = (t - t_n)(t - t_{n-1})\cdots(t - t_{n-k})c$$

for some constant vector c. Thus

$$\dot{\Delta}(t_n) = (t_n - t_{n-1})\cdots(t_n - t_{n-k})c = h^k\xi_1\xi_2\cdots\xi_k c, \qquad E_n(k) = h^{k+1}\xi_1\xi_2\cdots\xi_k c/\alpha_{n0}. \tag{2.31}$$

Since c is the leading coefficient of both Δ and $\pi_{n,k+1}$, we deduce from the interpolation properties (2.29) that

$$c = \pi_{n,k+1}{}^{(k+1)}(t_n)/(k + 1)! = Y_n{}^{(k+1)}/(k + 1)! + O(H).$$

Using this and the identity $l_1 = -\alpha_{n0}$, we therefore have

$$E_n(k) = -\xi_1\xi_2\cdots\xi_k h^{k+1} Y_n{}^{(k+1)}/[(k + 1)!\,l_1(k)] + O(H^{k+2}). \tag{2.32}$$

If the current order of the method is q, the first task of the error control algorithm is to estimate $E_n(q)$ in terms of existing quantities. Only the principal (asymptotic) part will be estimated, and this will be done by way of the difference $e_n = y_n - y_{n(0)}$ between the predicted and corrected values, in the manner of Milne [21, 33, 34], but with modifications given by Gear [16].

We start by deriving the asymptotic value of $Y_n - y_n$, where y_n is calculated from (2.2) (or, equivalently, (2.3)) at order q, with *exact* past values $y_{n-i} = Y_{n-i}$, $i \geq 1$. Using (2.26) with $k = q$, we have the relations

$$hF(y_n, t_n) + \alpha_{n0}y_n + \sum_1^q \alpha_{ni}Y_{n-i} = 0,$$

$$hF(Y_n, t_n) + \alpha_{n0}Y_n + \sum_1^q \alpha_{ni}Y_{n-i} = \alpha_{n0}E_n(q). \tag{2.33}$$

[3] Symbols $\pi_{n,k}$, Δ, and the like are used generically in this and later sections and are defined specifically only for the context in which they are used.

From these, since F is Lipschitz and $E_n(q) = O(H^{q+1})$, we can solve for $Y_n - y_n$ to get (recall that $L = q + 1$)

$$\begin{aligned} Y_n - y_n &= E_n(q) + O(H^{q+2}) = C_{n,q+1}h^{q+1}Y_n^{(q+1)} + O(H^{q+2}), \\ C_{n,q+1} &= -\xi_1\xi_2\cdots\xi_q/l_1L!. \end{aligned} \tag{2.34}$$

To get a similar statement about $y_{n(0)}$, we define a polynomial $\pi_{n-1,q}$ of degree q or less, as in (2.1), by

$$\pi_{n-1,q}(t_{n-i}) = Y_{n-i}\,,\ i = 1, 2, \ldots, q, \qquad \dot\pi_{n-1,q}(t_{n-1}) = \dot Y_{n-1}\,.$$

Then by definition $y_{n(0)} = \pi_{n-1,q}(t_n)$. We introduce also the polynomial $\pi_{n-1,q+1}$ of degree $q + 1$ or less, given by

$$\pi_{n-1,q+1}(t_{n-i}) = Y_{n-i}\,,\ i = 0, 1, \ldots, q, \qquad \dot\pi_{n-1,q+1}(t_{n-1}) = \dot Y_{n-1}\,.$$

Then

$$Y_n - y_{n(0)} = \pi_{n-1,q+1}(t_n) - \pi_{n-1,q}(t_n) = \Delta(t_n),$$

where $\Delta \equiv \pi_{n-1,q+1} - \pi_{n-1,q}$. From the construction,

$$\Delta(t_{n-i}) = 0,\ i = 1, 2, \ldots, q, \qquad \dot\Delta(t_{n-1}) = 0,$$

and so

$$\Delta(t) = (t - t_{n-1})^2(t - t_{n-2})\cdots(t - t_{n-q})c$$

for a constant vector c. As before, we find

$$c = \pi_{n-1,q+1}^{(q+1)}(t_n)/L! = Y_n^{(q+1)}/L! + O(H).$$

Thus

$$Y_n - y_{n(0)} = (t_n - t_{n-1})^2(t_n - t_{n-2})\cdots(t_n - t_{n-q})c = h^{q+1}\xi_2\xi_3\cdots\xi_q c,$$

$$Y_n - y_{n(0)} = \bar C_{n,q+1}h^{q+1}Y_n^{(q+1)} + O(H^{q+2}), \qquad \bar C_{n,q+1} = \xi_1\xi_2\cdots\xi_q/L!. \tag{2.35}$$

We could now subtract (2.34) from (2.35) to get an asymptotic relation for e_n, on the assumption of exact past values. However, the errors in the past values can be taken into account [16][4] if we assume that the global errors $Y_n - y_n$ at order q satisfy the expansion

$$y_n - Y_n = d_q(t_n)h^q + d_{q+1}(t_n)h^{q+1} + O(H^{q+2}), \tag{2.36}$$

with functions d_q and d_{q+1} which satisfy differential equations of the form $\dot d_k = F_y d_k + \phi_k$. This assumption is valid for constant h [46] and, at least under some circumstances, for nonconstant h [17]. Here ϕ_q is the "principal error function," for which $\alpha_{n0}^{-1}h^{q+1}\phi_q(t_n)$ is the principal term of the local error $E_n(q)$. The result of including these effects is that the asymptotic form of $e_n = y_n - y_{n(0)}$ is given by the difference between (2.34) and (2.35) but with an extra factor inserted into the coefficient $C_{n,q+1}$. That factor, in the present notation, is (cf. [16, eq.

[4] In applying the results of [16], we have modified the notation slightly. Here h is the current step size h_n, not the maximum value H, and asymptotic expansions are given in terms of h. The local error is defined by the $E_n(k)$ of (2.26), rather than by $\alpha_{n0}\, E_n(k)$ as used in [16].

(33)]) $\alpha_{n0}/\bar{\alpha}_{n0}$, where $\bar{\alpha}_{n0}$ is the coefficient analogous to α_{n0} in the predictor formula, in the form

$$\bar{\alpha}_{n0} y_{n(0)} + \sum_{i=1}^{q} \bar{\alpha}_{ni} y_{n-i} + h\dot{y}_{n-1} = 0. \tag{2.37}$$

Thus the result is

$$e_n = y_n - y_{n(0)} = (\bar{C}_{n,q+1} - C_{n,q+1}\alpha_{n0}/\bar{\alpha}_{n0}) h^{q+1} Y_n^{(q+1)} + O(H^{q+2}). \tag{2.38}$$

To compute $\bar{\alpha}_{n0}$, we refer to the interpolatory polynomial π_{n-1} in (2.1), and it is sufficient to consider the simple case $y_{n-1} = y_{n-2} = \cdots = y_{n-q} = 0$. Then we clearly have $\pi_{n-1}(t) = \prod_{i=1}^{q} (t - t_{n-i})c$ for some constant vector c. We can determine c from the final interpolation condition, $\dot{y}_{n-1} = \dot{\pi}_{n-1}(t_{n-1}) = \prod_{i=2}^{q} (t_{n-1} - t_{n-i})c$. Hence, on identifying $y_{n(0)}$ in (2.37) with $\pi_{n-1}(t_n)$, we find that

$$\bar{\alpha}_{n0} = -h \prod_{2}^{q} (t_{n-1} - t_{n-i}) \Big/ \prod_{1}^{q} (t_n - t_{n-i}) = -\prod_{2}^{q} [(t_{n-1} - t_{n-i})/(t_n - t_{n-i})]. \tag{2.39}$$

Inserting (2.34), (2.35), (2.39), and $\alpha_{n0} = -l_1$ into (2.38), we find that

$$e_n = c_n h^{q+1} Y_n^{(q+1)} + O(H^{q+2}), \quad c_n = \xi_1 \xi_2 \cdots \xi_q \left[1 + \prod_{2}^{q} \left(\frac{t_n - t_{n-i}}{t_{n-1} - t_{n-i}}\right)\right] \Big/ (q+1)!. \tag{2.40}$$

Thus, from (2.32), an estimate of $E_n(q)$ is given by

$$E_n(q) = -l_1^{-1} \left[1 + \prod_{2}^{q} \left(\frac{t_n - t_{n-i}}{t_{n-1} - t_{n-i}}\right)\right]^{-1} e_n + O(H^{q+2}). \tag{2.41}$$

Since $h^q Y_n^{(q)}/q!$ is approximated to within $O(H^{q+1})$ by the last column of the z_n array, we can write, from (2.32),

$$E_n(q-1) = -[\xi_1 \xi_2 \cdots \xi_{q-1}/l_1(q-1)](h^q y_n^{(q)}/q!) + O(H^{q+1}). \tag{2.42}$$

With these results the local error at order q and $q - 1$ can be estimated, when the current order is q.

It remains only to estimate $E_n(q+1)$. To do this, we use e_n and e_{n-1} in an appropriate linear combination. From (2.40), we see that, in order that this combination be asymptotically $O(H^{q+2})$, it must be proportional to $e_n - Q_n e_{n-1}$, where

$$Q_n = (c_n/c_{n-1})(h/h_{n-1})^{q+1}. \tag{2.43}$$

If we neglect the $O(H^{q+2})$ terms in (2.40), and if $Y \in C^{q+2}$, then we get from (2.40) (and from (2.40) with n replaced by $n - 1$) the relation

$$h^{q+2} Y_n^{(q+2)} = h^{q+1} Y_n^{(q+1)} - h^{q+1} Y_{n-1}^{(q+1)} + O(H^{q+3}) = c_n^{-1}(e_n - Q_n e_{n-1}) + O(H^{q+3}).$$

This in turn gives

$$E_n(q+1) = \frac{-\xi_{q+1}}{(q+2) l_1(q+1) \left[1 + \prod_{2}^{q} \left(\frac{t_n - t_{n-i}}{t_{n-1} - t_{n-i}}\right)\right]} (e_n - Q_n e_{n-1}) + O(H^{q+3}). \tag{2.44}$$

The neglected error terms in (2.40) make this somewhat inaccurate. The method of [16] can be used to derive the more correct asymptotic formula for $e_n - Q_n e_{n-1}$, based on (2.36), and hence a more accurate coefficient in (2.44). However, we have observed empirically that this correction has no significant beneficial effect, and so we have used (2.44) as it stands to estimate $E_n(q+1)$.

Many of the assumptions used to derive these error estimates are in practice unjustified and unjustifiable. The past values do not generally satisfy (2.36), the neglected remainder terms are not generally negligible, and the smooth differentiability ($q + 2$ times) of Y is often not present. However, the resulting formulas easily allow for the construction of a selection algorithm for step size and order. That algorithm, to be described shortly, works well in practice, and has a basis that is at least heuristically valid and is in part rigorously valid. Moreover, the extent to which the algorithm has a rigorous basis is greater than that of other error control algorithms that have been used [3, 23, 40]. In this connection, it should be pointed out that the result (2.32) is essentially given in [3] and, in the constant step case in [23], but that the estimators (2.41) and (2.44) differ substantially from those used elsewhere, even in the constant step case.[5]

2.4 Error Control and Selection of Step Size and Order

The algorithm for selecting h and q can now be summarized. On any given step, the user is required to provide an error tolerance parameter ϵ, a vector $w = (w_1, w_2, \ldots, w_N)^T$ of weights with which errors are to be compared, and an interval size S over which errors of size ϵ are to be allowed. After the step is taken, at order q, the quantity $D_q = \max_i \{| E_{ni}(q) |/w_i\}$ can be computed, where $E_{ni}(q)$ denotes the ith component of the vector $E_n(q)$, as estimated by (2.41). Then we require that

$$D_q \leq \bar{\epsilon} \equiv \epsilon h/S \tag{2.45}$$

for the step to be successful. Note that this controls error per interval of length S on the t axis, and therefore it controls error per unit step if $S = 1$, and error per step if h/S is replaced by 1. All of these options are easily available to the user. If the test (2.45) fails, then a new value of h which will make (2.45) hold is given approximately by $h' = \eta_q h$ with

$$\eta_q = (\bar{\epsilon}/D_q)^{1/(q+1)}, \tag{2.46}$$

on the basis of the asymptotic behavior of D_q. The step from t_{n-1} is then retaken,[6] but with the new step size h'. If the test succeeds, we still use (2.46) in an attempt to choose a larger step size h' for the next step. In either case, the necessary modification to the z_n array upon a change of h by a factor of η is given by multiplying by a diagonal matrix:

$$z_n \leftarrow z_n \operatorname{diag}(1, \eta, \eta^2, \ldots, \eta^q). \tag{2.47}$$

[5] In [3] the analog of (2.41) given is $E_n(q) \approx \xi_{q+1}^{-1} e_n$, and the calculation of l is done by solving an $L \times L$ system of linear equations.

[6] The retaking of a step requires that z_{n-1} be recovered from $z_{n(0)}$. This calls for a multiplication by $A(q)^{-1}$, which is done by repeated subtractions in a manner analogous to the repeated additions used to compute (2.7).

A change of order is contemplated after running at order q for $q + 1$ steps, and, for the sake of simplifying the algorithm, the change in q is restricted to ± 1, if any.[7] This strategy is that used by Gear [14] and suggested in [3] and has only a heuristic and empirical basis. It guarantees that all of the data involved in the step down from t_n to t_{n+1} at the new order have been computed at the same order.

The selection of the new order begins with the calculation of the two quantities

$$D_{q\pm 1} = \max_i \{| E_{ni}(q \pm 1) |/w_i\},$$

where (2.42) and (2.44) are used to estimate the components of $E_n(q \pm 1)$. Then, by analogy with (2.46), we estimate acceptable step sizes at orders $q - 1$ and $q + 1$ by $h' = \eta_{q-1}h$ and $h' = \eta_{q+1}h$ with

$$\eta_{q-1} = (\bar{\epsilon}/D_{q-1})^{1/q}, \qquad \eta_{q+1} = (\bar{\epsilon}/D_{q+1})^{1/(q+2)}. \tag{2.48}$$

The new order and step size used are determined by maximizing the next step size:

$$\eta = \max\{\eta_{q-1}, \eta_q, \eta_{q+1}\} = \eta_{q'}, \quad h \leftarrow \eta h, \quad q \leftarrow q'.$$

It remains only to describe the adjustments on the z_n array necessary when the order is changed. If $q' = q - 1$, the array z_n must be modified to an array z_n^* which represents the reduced set of data $\{y_n, y_{n-1}, \ldots, y_{n-q+2}, \dot{y}_n\}$ (q elements). Thus, while z_n is defined by the polynomial π_n of degree q or less in (2.2), z_n^* is defined by a polynomial π_n^* of degree $q - 1$ or less defined by $\pi_n^*(t_{n-i}) = y_{n-i}$, $i = 0, 1, \ldots, q - 2$, and $\dot{\pi}_n^*(t_n) = \dot{y}_n$. The difference $\Delta = \pi_n - \pi_n^*$ thus satisfies $\Delta(t_{n-i}) = 0$, $i = 0, 1, \ldots, q - 2$, $\dot{\Delta}(t_n) = 0$, while its leading coefficient is that of π_n, namely, $y_n^{(q)}/q!$, as in the last column of z_n. Therefore,

$$\Delta(t) = (t - t_n)^2 \left[\prod_1^{q-2} (t - t_{n-i})\right] y_n^{(q)}/q!.$$

The change of variables given by (2.14) yields

$$\Delta(t) = \Delta(t_n + hx) = d(x)h^q y_n^{(q)}/q!, \quad d(x) \equiv x^2 \prod_1^{q-2} (x + \xi_i). \tag{2.49}$$

The coefficients d_j of $d(x)$ are computed in the same way as those of $\Lambda_n(x)$, and $d_j h^q y_n^{(q)}/q!$ is subtracted from column j of z_n ($j = 2, 3, \ldots, q - 1$) to get z_n^*. Column q is ignored for the subsequent steps at order $q - 1$.

If $q' = q + 1$, the adjusted Nordsieck array z_n^* is to correspond to a polynomial π_n^* of degree $q + 1$ or less which interpolates the data set $\{y_n, y_{n-1}, \ldots, y_{n-q}, \dot{y}_n\}$ ($q + 2$ elements). But (2.2) says that π_n itself fits this set of data, and therefore $\pi_n^* = \pi_n$. Thus columns 0 to q of z_n need no adjustment, and column $q + 1$ of z_n^* is $h^{q+1}\pi_n^{(q+1)}/(q + 1)! = 0$. That is, because of the way the order q method for step n forces a degree q polynomial to fit $q + 2$ pieces of data, the order $q + 1$ method for step $n + 1$ begins with that same data set, and thus with the same Nordsieck array (augmented by a zero column) as produced on step n.

Finally, regardless of the value of q', the Nordsieck array is rescaled according to (2.47) for the new step size selected.

[7] In [40], the new order can be of any $1, 2, \ldots, q + 1$ when the current order is q. This strategy appears somewhat extreme, but in some cases may be superior.

Given this algorithm for step and order variability, the problem of startup is easily solved by setting $q = 1$ initially, thereby requiring no past history for the first step.

The actual programmed implementation of the above error control algorithm contains slight differences from the present description. Among these is the use of factors inserted into (2.46) and (2.48) which produce slightly smaller values of η, in order to allow for inaccuracies in the error estimates used.

2.5 Stability

The numerical stability properties of the algorithm described here are difficult or impossible to describe in the general setting of arbitrary step sizes, even for the simple equation usually used for this purpose, $\dot{y} = \lambda y$. There is some computational evidence that the numerical stability has been improved considerably in comparison with codes based on fixed-step formulas, especially in cases where the error control results in frequent changes of step size. That the numerical stability of Gear's methods with respect to stiffness has not been degraded with the generalization to variable-step formulas is indicated by the following result, easily verified by induction from (2.3). If the algorithm is applied at fixed order to $\dot{y} = \lambda y$, with initial values satisfying $| y_i | \leq M$, $i = 0, 1, \ldots, q - 1$, and with step sizes h_n such that $| \alpha_{ni} | \leq \alpha$ for all n and i and $| h_n \lambda | \geq D \geq (q + 1)\alpha$ for all n, then the computed values y_n satisfy

$$| y_{q+n} | \leq (\alpha q/(D - \alpha))^{n+1} M, \quad n = 0, 1, \ldots . \tag{2.50}$$

This says that at fixed order the computed y_n tend to zero in an exponential manner as long as $h\lambda$ stays outside some bounded region. In particular, the algorithm is stiffly stable in the sense of Gear (this sense being appropriately generalized as stated for unequal time steps).

The regions in question here are known precisely in the case of constant h [13]. For orders 1 to 6 the region of stability includes in particular all of the negative real axis in the complex h plane. For $q > 6$ it does not, and for this reason those orders are excluded from the implemented algorithm. For $q = 6$, the stability region is quite restricted in vertical extent near (but to the left of) the origin. Hence, this order is also excluded, and the available orders for the stiff methods are $q = 1$, $2, \ldots, 5$. (See also [37] and [40].)

2.6 Output Values

An important point in the implementation of any variable-step method is the calculation of solution values at prescribed output times t. As with other codes involving the Nordsieck history array (2.9), this calculation is very easily handled as follows: If z_n exists at $t = t_n$, and output is desired at $t = t^*$ with $t_{n-1} < t^* \leq t_n$, then a sum which approximates a finite Taylor series is computed from z_n, as

$$y^* = \sum_{j=0}^{q} [(t^* - t_n)/h^j](h^j y_n^{(j)}/j!), \tag{2.51}$$

where q is the current order and h the current step size. This is equivalent to the evaluation at t^* of the interpolation polynomial π_n represented by z_n.

3. THE INTEGRATOR FOR NONSTIFF ORDINARY DIFFERENTIAL EQUATIONS

This integrator makes use of the implicit variable-step Adams formulas of orders one through twelve. Although other Adams integrators do exist [13, 27, 38, 42, 43], this one is unique in that it shares much of the philosophy of the method in Section 2. The Nordsieck matrix is retained. Consequently, the predicted value $z_{n(0)}$ is again obtained via (2.7), and output values are computed by (2.51). A generating polynomial analogous to (2.15) is used to provide the coefficients l_i, which are required for computing the zero of G as in (2.22)–(2.25) and correcting the Nordsieck matrix (2.19). Error estimates similar to those in (2.41)–(2.44) are used to change order and step size with the same general technique as was described in Section 2.4.

The differences arise from the fact that, while the stiff methods come from the differentiation of an interpolatory polynomial of degree q, the Adams methods come from the integration of an interpolatory polynomial of degree $q - 1$ (q being the order of the method in both cases).

The fundamental ideas here are best stated, as before, in terms of these interpolatory polynomials. We begin with the polynomial π_{n-1} of degree q or less given by the $L = q + 1$ conditions

$$\pi_{n-1}(t_{n-1}) = y_{n-1}, \qquad \dot{\pi}_{n-1}(t_{n-i}) = \dot{y}_{n-i},\ i = 1, 2, \ldots, q, \tag{3.1}$$

in terms of the calculated values y_j and $\dot{y}_j$ approximating $Y(t_j)$ and $\dot{Y}(t_j)$, respectively. Step n is then to be taken by constructing a polynomial π_n of degree q or less which satisfies the $L + 1$ conditions

$$\begin{gathered}\pi_n(t_n) = y_n,\ \pi_n(t_{n-1}) = y_{n-1},\ \dot{\pi}_n(t_{n-i}) = \dot{y}_{n-i},\ i = 0, 1, \ldots, q-1, \\ [\dot{y}_n \equiv F(y_n, t_n)]\end{gathered} \tag{3.2}$$

and to compute a value of y_n which makes this possible. This set of conditions can be expressed equivalently in the classical linear multistep form

$$y_n = y_{n-1} + h_n \sum_{i=0}^{q-1} \beta_{ni}\dot{y}_{n-i}. \tag{3.3}$$

Here the coefficients β_{ni} are functions of step sizes $h_j = t_j - t_{j-1}$, as well as the order q, generalizing the classical implicit Adams formulas.

The solution of (3.2) for y_n begins with the prediction of y_n and $\dot{y}_n$ by use of π_{n-1}, as follows:

$$y_{n(0)} = \pi_{n-1}(t_n), \qquad \dot{y}_{n(0)} = \dot{\pi}_{n-1}(t_n). \tag{3.4}$$

(While these predicted values involve both y_{n-1} and y_{n-2} as well as past values of $\dot{y}_j$, they can be written in terms of y_{n-1} and past $\dot{y}_j$ only, as in the classical Adams predictor, by use of (3.3) with n replaced by $n - 1$.) Subsequent iterations are then taken to solve for y_n in the equation

$$h\dot{y}_n = hF(y_n, t_n) = h\dot{y}_{n(0)} + (y_n - y_{n(0)})l_1 \tag{3.5}$$

where $h = h_n$ and $l_1 = 1/\beta_{n0}$, this equation being an equivalent form of (3.3). The same set of four corrector iteration methods is available here as described in Section 2.2, (2.22)–(2.25). The choice of functional iteration yields the variable-step generalization of the Adams-Bashforth-Moulton method.

As with the stiff methods, past history information is stored in the form of the Nordsieck history array z_{n-1} in (2.6) whose columns are $h^j \pi_{n-1}^{(j)}(t_{n-1})/j!$ at step $n-1$, $j = 0, 1, \ldots, q$.[8] With this, the prediction is done by multiplying z_{n-1} by the Pascal triangle matrix $A(q)$ as in (2.7) and (2.8).

The updating of the array z_n after y_n has been calculated is based on formulas analogous to those in Section 2.1. If we define $\Delta_n = \pi_n - \pi_{n-1}$, we see from (3.1), (3.2), and (3.4) that

$$\Delta_n(t_n) = y_n - y_{n(0)} \equiv e_n, \quad \Delta_n(t_{n-1}) = 0, \quad \dot{\Delta}_n(t_{n-i}) = 0, \; i = 1, 2, \ldots, q-1. \tag{3.6}$$

This uniquely determines Δ_n to be

$$\Delta_n(t) = \Delta_n(t_n + hx) = \Lambda_n(x)e_n, \quad [x = (t - t_n)/h], \tag{3.7}$$

where Λ_n is a scalar polynomial determined uniquely by

$$\Lambda_n(0) = 1, \; \Lambda_n(-1) = 0, \; \dot{\Lambda}_n(-\xi_i) = 0, \; i = 1, 2, \ldots, q-1, \; \xi_i \equiv (t_n - t_{n-i})/h. \tag{3.8}$$

Explicitly, we can construct Λ_n as

$$\Lambda_n(x) = \int_{-1}^{x} \prod_{1}^{q-1} (u + \xi_i)\, du \Big/ \int_{-1}^{0} \prod_{1}^{q-1} (u + \xi_i)\, du. \tag{3.9}$$

The coefficients l_j of Λ_n in

$$\Lambda_n(x) = \sum_{j=0}^{q} l_j x^j \tag{3.10}$$

can be computed in a straightforward manner from (3.9). When they are known, we have, as derived in Section 2.1,

$$z_n = z_{n(0)} + e_n l, \quad l \equiv [l_0, l_1, \ldots, l_q]. \tag{3.11}$$

The relation from column 1 in (3.11) (counting from 0 to q) is just (3.5). The first two cases are given by $\Lambda_n(x) = 1 + x$ $(q = 1)$, $\Lambda_n(x) = (1 + x)^2$ $(q = 2)$.

In order to complete the algorithm, the method of selecting h and q, on the basis of the control of local truncation errors, must be described. The local error for the method of order k is defined (following (2.1)) as

$$E_n(k) = Y_n - \left[Y_{n-1} + h \sum_{i=0}^{k-1} \beta_{ni} \dot{Y}_{n-i} \right], \tag{3.12}$$

where the Y_j and $\dot{Y}_j$ are function and derivative values for an exact solution Y for which $Y(t_{n-1}) = Y_{n-1} = y_{n-1}$. The β_{ni} are the coefficients for the order k method as in (3.3), and k is fixed at one of the values q, $q-1$, or $q+1$.

Working in analogy with Section 2.3, we define a polynomial $\pi_{n,k}$ of degree k or less, by

$$\pi_{n,k}(t_{n-1}) = Y_{n-1}, \quad \dot{\pi}_{n,k}(t_{n-i}) = \dot{Y}_{n-i}, \; i = 0, 1, \ldots, k-1. \tag{3.13}$$

[8] The accuracy of these vectors within $O(H^{q+1})$ for data of the same accuracy follows from arguments similar to those in Section 2.1, except that the interpolation formulas are applied to $\dot{Y}$, not Y.

Then $\pi_{n,k}(t_n)$ is the estimate of Y_n given by the bracketed terms in (3.12), or $E_n(k) = Y_n - \pi_{n,k}(t_n)$.

Defining also a polynomial $\pi_{n,k+1}$ of degree $k + 1$ or less by

$$\pi_{n,k+1}(t_n) = Y_n\,,\quad \pi_{n,k+1}(t_{n-1}) = Y_{n-1}\,,\quad \dot{\pi}_{n,k+1}(t_{n-i}) = \dot{Y}_{n-i}\,,\ i = 0, 1, \ldots, k-1, \tag{3.14}$$

and the difference polynomial $\Delta = \pi_{n,k+1} - \pi_{n,k}$, we have

$$E_n(k) = \Delta(t_n). \tag{3.15}$$

From the conditions $\Delta(t_{n-1}) = 0$, $\dot{\Delta}(t_{n-i}) = 0$, $i = 0, 1, \ldots, k-1$, we have $\Delta(t) = c \int_{t_{n-1}}^{t} \prod_{i=0}^{k-1} (s - t_{n-i})\, ds$ for some constant vector c. But $c/(k + 1)$ is the leading coefficient of the interpolatory polynomial in (3.14), and so $c/(k + 1) = \pi_{n,k+1}{}^{(k+1)}(t_n)/(k + 1)! = Y_n{}^{(k+1)}/(k + 1)! + O(H)$. Substituting into (3.15), and making the change of variables $t = t_n + xh$, we obtain

$$E_n(k) = \left[\int_{-1}^{0} \prod_{0}^{k-1} (x + \xi_i)\, dx/k!\right] h^{k+1} Y_n{}^{(k+1)} + O(H^{k+2}). \tag{3.16}$$

We estimate the principal (asymptotic) term in (3.16) by use of Milne-type formulas. This requires first the asymptotic formula for $Y_n - y_n$, where y_n is calculated from (3.2) (at order q) with exact past values $y_{n-1} = Y_{n-1}$, $\dot{y}_{n-i} = \dot{Y}_{n-i}$, $i = 1, 2, \ldots, q-1$. From the relations (using (3.3) and $k = q$ in (3.12))

$$\beta_{n0} h F(y_n\,, t_n) + h \sum_{i=1}^{q-1} \beta_{ni} \dot{Y}_{n-i} + Y_{n-1} - y_n = 0,$$

$$\beta_{n0} h F(Y_n\,, t_n) + h \sum_{i=1}^{q-1} \beta_{ni} \dot{Y}_{n-i} + Y_{n-1} - Y_n = -E_n(q),$$

we can solve for $Y_n - y_n$ to get

$$Y_n - y_n = E_n(q) + O(H^{q+2}) = \left[\int_{-1}^{0} \prod_{0}^{q-1} (x + \xi_i)\, dx/q!\right] h^{q+1} Y_n{}^{(q+1)} + O(H^{q+2}). \tag{3.17}$$

To get a similar result for $y_{n(0)}$, we define a polynomial $\pi_{n-1,q}$ of degree q or less by

$$\pi_{n-1,q}(t_{n-1}) = Y_{n-1}\,,\qquad \dot{\pi}_{n-1,q}(t_{n-i}) = \dot{Y}_{n-i}\,,\ i = 1, 2, \ldots, q, \tag{3.18}$$

so that $y_{n(0)} = \pi_{n-1,q}(t_n)$. Defining also a polynomial $\pi_{n-1,q+1}(t)$ of degree $q + 1$ or less by

$$\pi_{n-1,q+1}(t_n) = Y_n\,,\quad \pi_{n-1,q+1}(t_{n-1}) = Y_{n-1}\,,\quad \dot{\pi}_{n-1,q+1}(t_{n-i}) = \dot{Y}_{n-i}\,,\ i = 1, 2, \ldots, q, \tag{3.19}$$

and the difference polynomial $\Delta = \pi_{n-1,q+1} - \pi_{n-1,q}$, we have $Y_n - y_{n(0)} = \Delta(t_n)$. But from (3.18) and (3.19), $\Delta(t_{n-1}) = 0$ and $\dot{\Delta}(t_{n-i}) = 0$, $i = 1, 2, \ldots, q$, and therefore $\Delta(t) = c \int_{t_{n-1}}^{t} \prod_{i=1}^{q} (s - t_{n-i})\, ds$ for a constant vector c. As before, we can write

$$c/(q + 1) = \pi_{n-1,q+1}{}^{(q+1)}(t_n)/L! = Y_n{}^{(q+1)}/L! + O(H) \quad (L = q + 1).$$

Substitution and a change of variables yield

$$Y_n - y_{n(0)} = \left[\int_{-1}^{0} \prod_{1}^{q} (x + \xi_i)\, dx/q!\right] h^{q+1} Y_n^{(q+1)} + O(H^{q+2}).$$

As pointed out in [16, lemma 1], errors in the past values do not require a correction here to the coefficient in the asymptotic value of $e_n = y_n - y_{n(0)}$, as long as (2.36) holds. This is a fortuitous property of the Adams methods not shared by those of Section 2. Thus we can simply subtract (3.17) from the above to get

$$e_n = y_n - y_{n(0)} = \frac{\xi_q}{q!}\left[\int_{-1}^{0} \prod_{1}^{q-1} (x + \xi_i)\, dx\right] h^{q+1} Y_n^{(q+1)} + O(H^{q+2})$$

$$= (\xi_q / q l_q q!) h^{q+1} Y_n^{(q+1)} + O(H^{q+2}). \tag{3.20}$$

Here the relation $q l_q = 1/\int_{-1}^{0} \prod_{1}^{q-1} (x + \xi_i)\, dx$, from (3.9) and (3.10) has been used. From (3.16) we now get

$$E_n(q) = \left[q l_q \int_{-1}^{0} \prod_{0}^{q-1} (x + \xi_i)\, dx/\xi_q\right] e_n + O(H^{q+2}). \tag{3.21}$$

This formula, together with the formula

$$E_n(q-1) = \left[q \int_{-1}^{0} \prod_{0}^{q-2} (x + \xi_i)\, dx\right] (h^q y_n^{(q)}/q!) + O(H^{q+1}), \tag{3.22}$$

based on the last column of z_n, allow for the estimation of local errors at orders q and $q - 1$, if the current order is q.

To estimate $E_n(q + 1)$, we must again make the assumption (without justification) that the remainder term in (3.20), multiplied by $l_q/h^{q+1}\xi_q$, is smoothly varying from step to step. Then we can write (3.20) at step $n - 1$ as well as n, and combine to get an estimate of $h^{q+2} Y_n^{(q+2)}$. When inserted into (3.16) ($k = q + 1$), the result is

$$E_n(q+1) = \left[q l_q \int_{-1}^{0} \prod_{0}^{q} (x + \xi_i)\, dx/L\xi_q\right] (e_n - Q_n e_{n-1}) + O(H^{q+3}) \tag{3.23}$$

where $Q_n = [\xi_q(n) l_q(n-1)/\xi_q(n-1) l_q(n)](h_n/h_{n-1})^{q+1}$.

The comments in Section 2.3 about the heuristic nature of these error estimates apply equally well here. However, in contrast to the situation for the backward differentiation methods, we have not seen any other such estimation formulas with which to compare. Moreover, no other implementations of variable-step Adams methods with the Nordsieck history array seem to exist.

The descriptions and justifications in Section 2.4 of the algorithm for error control and step/order selection apply here largely without change. The error norms D_k ($k = q - 1, q, q + 1$) are computed with the help of (3.21), (3.22), and (3.23), and the control of h and q on the basis of these proceeds as given in (2.45)–(2.48). If the order is to be reduced to $q' = q - 1$, an adjustment to the array z_n is necessary, as in Section 2. The adjusted array z_n^* is to correspond to the polynomial π_n^* of degree $q - 1$ or less, given by

$$\pi_n^*(t_n) = y_n, \quad \dot{\pi}_n^*(t_{n-i}) = \dot{y}_{n-i}, \quad i = 0, 1, \ldots, q - 2. \tag{3.24}$$

By comparison with (3.2), we obtain

$$\pi_n(t) - \pi_n^*(t) = d(x)h^q y_n^{(q)}/q! \ (t = t_n + hx), \quad d(x) = q\int_0^x \prod_0^{q-2} (u + \xi_i)\,du, \tag{3.25}$$

where $h^q y_n^{(q)}/q!$ is the last column of z_n. (The derivation of (3.25) is completely analogous to that of (2.49).) If $q' = q + 1$, the adjusted array z_n^* is just z_n augmented by a zero column, by the same reasoning as in Section 2. (See [43] for a similar procedure for the variable-step Adams methods with a different type of history vector.)

As with the stiff integrator, startup is achieved by using $q = 1$ initially, and the order is kept equal to q for $q + 1$ consecutive steps before considering a change of order. Slight differences in the programmed implementation are the same as for the stiff integrator, as is the interpolation to nonmesh points (2.51).

The stability regions of the constant-step Adams formulas impose no restrictions on the order, unlike the case for the constant-step backward differentiation formulas. Therefore, the EPISODE program allows orders from one to twelve, this being regarded as maximum range likely to be called for in practical applications. For discussions of the stability of the variable-step, variable-order Adams method, see [17, 18, 38, 47].

4. NUMERICAL RESULTS

In this section, we give numerical results for the EPISODE package for two example problems. Of course, much more testing has been, and will be, done with the package than is reported here. These two problems are intended only to indicate the typical characteristics of the methods used.

In order to present results for the various options, we use the following notation, which is that used in the EPISODE program and in [22]. The method used is identified by a "method flag" MF, which has the values 10, 11, 12, 13, 20, 21, 22, and 23. The first decimal digit of MF is denoted $METH$, and indicates the basic method used: 1 for Adams, and 2 for backward differentiation. The second digit, $MITER$, indicates the corrector iteration method: 0 for functional iteration, 1 for the chord method with a user-supplied Jacobian, 2 for the chord method with finite-difference Jacobian, and 3 for the chord method with a diagonal approximation to the Jacobian. (These methods were described in Section 2.2.)

For the computational results which follow, *error per step* was controlled, and the calculations were performed on a CDC-7600 computer.

Example 1. A mockup of a photochemistry problem [10, 19], with a diurnally varying source term was given in Section 1, (1.5). The source term has a period of 1 day (86400 seconds) and has a sharp rise and a sharp dropoff in each period, representing the effect of the sun. The problem is quite stiff, because of the short time constant $B^{-1} = 10^{-8}$ seconds, while the problem is to be run for 5 days.

This problem was run with a local error tolerance ϵ of 10^{-3}, 10^{-6}, and 10^{-9}. The weight w_i of Section 2.4 was taken as $w_{1n} = \max_{m<n} |y_m|$. The initial step size used was $h = \epsilon/100$. The options MF = 21, 22, or 23 all performed about equally well, while the others were quite unsatisfactory, because of the stiffness. Because of the

updating of the P matrix, $MF = 23$ costs about 20 percent less here. Only the results for $MF = 21$ are given here. Table I gives, for each ϵ, the following four quantities:

NS = number of steps taken
NF = number of f evaluations
NJ = number of J evaluations (= number of LU decompositions if $MITER$ is 1 or 2)
EO = maximum error overrun, defined as the largest value of $| y_n - y(t_n) | / \epsilon w_{1n}$ seen.

For comparison purposes, the same table is given for the GEAR package [22]. In cases where the problem was not completed, the largest value of t to which integration was successfully completed is given in parentheses under NS. In these cases, integration was interrupted when repeated error test failures caused the program to halt.

It should be noted that the GEAR package failed to complete any of the cases except that with the largest value of ϵ (and there only for $MF = 22$). However, for that case, it was competitive with EPISODE in efficiency.

This problem (aside from the methods used on it here) illustrates an interesting point. Because of the periodic near-discontinuity, the step size h can conceivably reach levels as low as ut, where u is the unit roundoff of the machine. This would not seem acceptable under most circumstances, and, in fact, causes an error halt in some programs. In this problem it can be a normal occurrence. EPISODE allows this occurrence with no ill effects, except that t is temporarily constant in the machine.

The problem in this example is a simple prototype of a large class of diurnal kinetic modeling problems [19]. Such problems, with up to 396 dependent variables, have been run successfully with EPISODE, in connection with air pollution studies. The GEAR package, on the other hand, has had very little success with these problems.

Example 2. This example is based on the following diffusion-convection problem, posed by Sincovec [45]. A function $u(x, t)$ on $0 \le x \le 1$, $t \ge 0$, is to satisfy

$$\partial u/\partial t = \partial^2 u/\partial x^2 - c \cdot \partial u/\partial x, \quad u(0, t) = 1 \text{ for } t > 0,$$
$$(\partial u/\partial x)(1, t) = 0 \text{ for } t > 0, \quad u(x, 0) = 0 \text{ for } 0 < x < 1. \tag{4.1}$$

The solution of this problem involves a moving wave front whose steepness and

Table I. Numerical Results for Diurnal Chemistry Problem

Program	ϵ	NS	NF	NJ	EO
EPISODE	10^{-3}	894	1446	440	.05
	10^{-6}	2133	3864	621	.98
	10^{-9}	5281	9625	915	.31
GEAR	10^{-3}	940	1543	265	.24
	10^{-6}	$(2.1 \cdot 10^5)$			
	10^{-9}	(1.2)			

speed is measured by the constant c. The problem serves as a model for problems in atmospheric simulation and in oil recovery.

The partial differential equation here is easily treated by the method of lines [31]. We discretize the space variable x and replace spatial derivatives by central difference approximations. If we use an equally spaced mesh of N intervals in x and approximate $u(k/N, t)$ by $u_k(t)$, we arrive at the system of ordinary differential equations

$$\dot{u}_k = (u_{k-1} - 2u_k + u_{k+1})/(1/N)^2 - c(u_{k+1} - u_{k-1})/(2/N), \quad k = 1, 2, \ldots, N.$$

We then simulate initial and boundary conditions by setting

$$u_k(0) = 0 \ (k = 1, 2, \ldots, N), \quad u_0 = 1 \ (\text{all } t), \quad u_{N+1} = u_{N-1} \ (\text{all } t).$$

The values used for this example are $N = 100$ and $c = 200$, and the time interval is $0 \leq t \leq 1/2c$ (at the end of which the front is roughly centered at $x = \frac{1}{2}$). The problem was run under both EPISODE and GEAR for all eight values of MF. The error control was on absolute error. The results are given in Table II, where a block containing the three numbers NS, NF, and NJ is given in each table entry. The values of ϵ and the initial h were chosen as in Example 1. As judged from the results for $\epsilon = 10^{-9}$, the errors in the results for the larger ϵ appeared to be about ϵ or less, uniformly.

As can be seen from Table II, there is not a dramatic variation in efficiency among the various choices of MF here, except that, understandably, $MITER = 3$ is a poor choice. This indicates that this problem is at most only mildly stiff. (In contrast, a larger value of N or a smaller value of c is known to make it stiff [31].)

The table also demonstrates the need to choose $METH$ and $MITER$ *independently*. For example, when $\epsilon = 10^{-9}$, the GEAR package performs best with $MF = 11$ here. This is not possible with Gear's program [15], where, in the notation used here, only $MF = 10$, 21, and 22 are available.

Another inference here is that EPISODE requires more frequent updating of the matrix $P = I - hl_1^{-1}J$ than does GEAR. This is to be expected, as the coefficient hl_1^{-1} is changing much more frequently (possibly every step) in EPISODE. The inaccuracy of P also tends to cause EPISODE to require more corrector iterations per step, although this is counterbalanced somewhat by the larger size of the steps. It is hoped that a more effective strategy for updating P will be developed in the future for this situation.

The problem in Example 2 has the simple form $\dot{y} = Ay$ for a constant (tridiagonal) matrix A. For this class of problems, more efficient solution methods (e.g. [4, 7]) exist. Here, the example serves only to represent a larger class of problems, for which those special methods do not apply.

5. SUMMARY

The algorithms described here together constitute a polyalgorithm in the sense of Rice [39], that is, a collection of algorithms directed at the same mathematical problem. In this case there are two basic integrators, stiff and nonstiff, each of which can be used with any of the four corrector iteration methods described in Section 2.2. The iteration method can be chosen independently of the basic integrator. The

Table II. Numerical Results for Diffusion-Convection Problem

	MF							
ϵ	10	11	12	13	20	21	22	23
				EPISODE				
	118	63	63	199	120	60	60	207
10^{-3}	219	90	90	456	230	83	83	477
	0	10	10	144	0	12	12	136
	229	137	137	360	189	173	173	347
10^{-6}	343	192	192	801	309	200	200	788
	0	21	21	169	0	18	18	219
	461	282	282	701	521	524	524	777
10^{-9}	897	395	395	1488	573	552	552	1414
	0	45	45	241	0	39	39	291
				GEAR				
	149	56	55	202	152	74	75	263
10^{-3}	250	91	86	452	259	110	111	597
	0	9	9	110	0	9	9	147
	278	150	140	402	212	193	201	432
10^{-6}	434	205	198	880	335	231	240	1035
	0	10	11	143	0	11	14	171
	706	288	315	747	580	558	580	897
10^{-9}	797	359	396	1729	621	596	621	2125
	0	13	21	202	0	13	21	251

need for this independence can be shown by numerical examples. As a result, the polyalgorithm implemented in EPISODE consists of eight different algorithms. Of course, there is a great deal in common in the eight algorithms, and much of the program is shared among them.

The features that characterize this polyalgorithm include the following: dynamically variable step size and order, implicit Adams methods of orders one through twelve, backward differentiation formulas of orders one through five, a variety of corrector iteration methods, the use of the Nordsieck history array, and estimates of the principal part of the local error. None of these features is new, but the combination of them in one program is.

The advantages generally attributed to the use of a Nordsieck array are the following. For a given order, the array is the same no matter what the linear multistep or multivalue method and so can be used for more than one such method at once. If h is changed, the adjustment of z_n is trivial; one rescales it by powers of h'/h. If output is desired at nonmesh points, it is easily obtained by means of a finite Taylor series. None of these advantages relies on the use of fixed-step formulas or interpolation for change of step size. Moreover, the added overhead associated

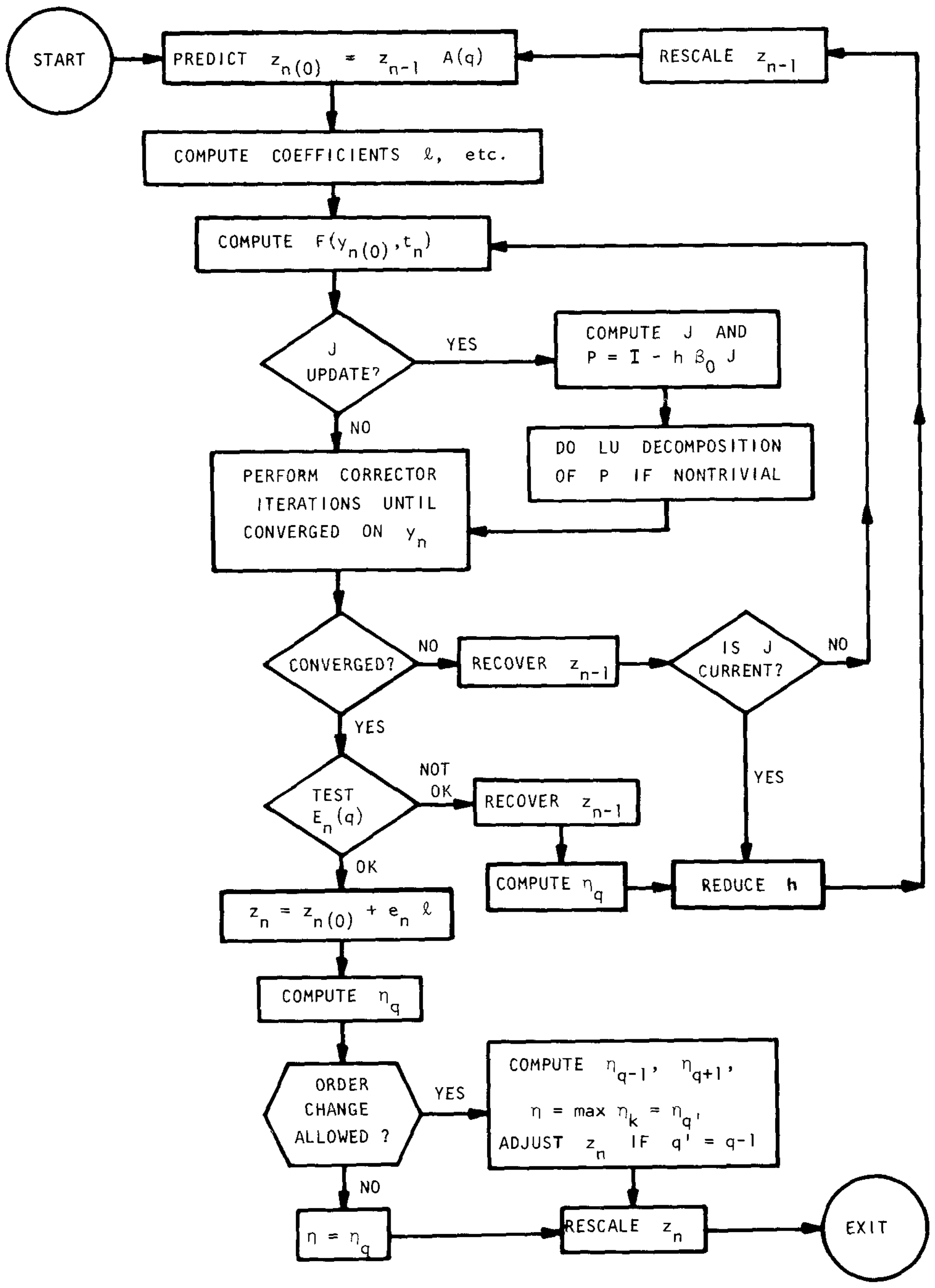
START
PREDICT $z_{n(0)} = z_{n-1} A(q)$
RESCALE z_{n-1}
COMPUTE COEFFICIENTS ℓ, etc.
COMPUTE $F(y_{n(0)}, t_n)$
J UPDATE?
YES
COMPUTE J AND $P = I - h\beta_0 J$
DO LU DECOMPOSITION OF P IF NONTRIVIAL
NO
PERFORM CORRECTOR ITERATIONS UNTIL CONVERGED ON y_n
CONVERGED?
NO
RECOVER z_{n-1}
IS J CURRENT?
NO
YES
YES
TEST $E_n(q)$
NOT OK
RECOVER z_{n-1}
COMPUTE η_q
REDUCE h
OK
$z_n = z_{n(0)} + e_n \ell$
COMPUTE η_q
ORDER CHANGE ALLOWED ?
YES
COMPUTE η_{q-1}, η_{q+1}, $\eta = \max \eta_k = \eta_{q'}$ ADJUST z_n IF $q' = q-1$
NO
$\eta = \eta_q$
RESCALE z_n
EXIT

Fig. 1. Block diagram of core integrator

with the variable-step, variable-order feature does not seem to be appreciably greater than for any other type of history array.

The macroscopic flowchart in Figure 1 shows the basic algorithm for taking a single integration step. This algorithm is implemented in a core subroutine. There are auxiliary subroutines for the computing of coefficients needed, the LU decomposition and backsubstitution for the solution of linear systems, the adjustments to z_n on reduction of order, and the interpolation to off-mesh points. To complete the EPISODE package, a driver subroutine has been written to make repeated calls to the core integrator and to do the appropriate checking and looping. This structure has been purposefully made modular in that, as far as is practical, specific tasks are assigned to separate subroutines. Modular structure allows for ease of modification when necessary, and hopefully also for a better understanding of the total package.

The EPISODE package, as the name implies, is at present still in an experimental stage. We expect that it will undergo considerable "fine tuning" and testing before it is widely released. However, even at this time, we believe that EPISODE adequately fills a major gap in the array of existing software for ordinary differential equations.

ACKNOWLEDGMENTS

This paper, and more directly the EPISODE program, has benefited greatly from discussions with two of its most enlightened users, Niel Madsen and Robert Dickinson.

REFERENCES

1. Brown, R. L. Recursive calculations of corrector coefficients. ACM SIGNUM Newsletter, *8* (Oct. 1973), 12–13.
2. Bjurel, G., Dahlquist, G., Lindberg, B., Linde, S., and Oden, L.. Survey of stiff ordinary differential equations. Rep. NA 70.11, Dep. of Information Processing, Royal Inst. Technology, S-100 44, Stockholm, Nov. 1972.
3. Brayton, R. K., Gustavson, F. G., and Hachtel, G. D. A new efficient algorithm for solving differential-algebraic systems using implicit backward differentiation formulas. *Proc. IEEE 60* (1972), 98–108.
4. Cavendish, J. C., Culham, W. E., and Varga, R. S. A comparison of Crank-Nicolson and Chebyshev-rational methods for numerically solving linear parabolic equations. *J. Computational Phys. 10* (1972), 354–368.
5. Chang, J. S., Hindmarsh, A. C., and Madsen, N. K. Simulation of chemical kinetics transport in the stratosphere. Rep. UCRL-74823, Lawrence Livermore Lab., U. of California, Livermore, Calif., Oct. 1973. (Also in *Stiff Differential Systems*, R. A. Willoughby, Ed., Plenum Press, New York, 1974, pp. 51–65.)
6. Coddington, E. A., and Levinson, N. *Theory of Ordinary Differential Equations.* McGraw-Hill, New York, 1955.
7. Cody, W. J., Meinardus, G., and Varga, R. S. Chebyshev rational approximations to e^{-x} in $[0, +\infty)$ and applications to heat conduction problems. *J. Approximation Theory 2* (1969), 50–65.
8. Crane, P. C., and Fox, P. A. A comparative study of computer programs for integrating differential equations. In Numerical Mathematics Computer Programs, Library One—Basic Routines for General Use, Vol. 2, Issue 2, Numerical Mathematics Program Library Project, Computer Sci. Res. Cent., Bell Telephone Lab., Inc., Murray Hill, N.J., Feb. 1969.
9. Curtiss, C. F., and Hirschfelder, J. O. Integration of stiff equations. *Proc. Nat. Acad. Sci. U.S.A. 38* (1952), 235–243.

10. DICKINSON, R. P., JR. Private communication.
11. EHLE, B. L. A comparison of some methods for solving certain stiff ordinary differential equations. Rep. 70, Dep. of Math., U. of Victoria, Victoria, B.C., Canada, Nov. 1972.
12. ENRIGHT, W. H. Studies in the numerical solution of stiff differential equations. Tech. Rep. 46, Dep. of Computer Sci., U. of Toronto, Toronto, Ont., Canada, Oct. 1972. (Also Ph.D. Th., Dep. of Computer Science, U. of Toronto, 1972.)
13. GEAR, C. W. *Numerical Initial Value Problems in Ordinary Differential Equations.* Prentice-Hall, Englewood Cliffs, N.J., 1971.
14. GEAR, C. W. The automatic integration of ordinary differential equations. *Comm. ACM 14,* 3 (March 1971), 176–179.
15. GEAR, C. W. Algorithm 407: DIFSUB for solution of ordinary differential equations. *Comm. ACM 14,* 3 (March 1971), 185–190.
16. GEAR, C. W. Asymptotic estimation of errors and derivatives for the numerical solution of ordinary differential equations. UIUCDCS-R-73-598, U. of Illinois, Urbana, Ill., Oct. 1973; also, in Information Processing 74 (IFIP 74), Vol. 3, North-Holland, Amsterdam, 1974, pp. 447–451.
17. GEAR, C. W., AND TU, K.-W. The effect of variable mesh size on the stability of multistep methods. UIUCDCS-R-73-570, U. of Illinois, Urbana, Ill., April 1973. (Also in *SIAM J. Numer. Anal. 11* (1974), 1025–1043.)
18. GEAR, C. W., AND WATANABE, D. S. Stability and convergence of variable order multistep methods. *SIAM J. Numer. Anal. 11* (1974), 1044–1058.
19. GELINAS, R. J. Diurnal kinetic modelling. UCRL-75373, Lawrence Livermore Lab., U. of California, Livermore, Calif., Jan. 1974. (To appear in Proc. of the IAMAP/IAPSO Conf., Melbourne, Australia, Jan. 14–25, 1974)
20. HALE, J. K. *Ordinary Differential Equations.* Wiley-Interscience, New York, 1969.
21. HENRICI, P. *Discrete Variable Methods in Ordinary Differential Equations.* Wiley, New York, 1962.
22. HINDMARSH, A. C. GEAR: ordinary differential equation system solver. UCID-30001, Rev. 2, Lawrence Livermore Lab., U. of California, Livermore, Calif., Aug. 1972.
23. HINDMARSH, A. C. The construction of mathematical software, part III: the control of error in the GEAR package for ordinary differential equations. UCID-30050, Pt. 3, Lawrence Livermore Lab , U. of California, Livermore, Calif., Aug. 1972.
24. HINDMARSH, A. C. Linear multistep methods for ordinary differential equations: method formulations, stability, and the methods of Nordsieck and Gear. UCRL-51186, Rev. 1, Lawrence Livermore Lab., U. of California, Livermore, Calif., March 1972.
25. HINDMARSH, A. C. GEARB: solution of ordinary differential equations having banded Jacobian. UCID-30059, Lawrence Livermore Lab., U. of California, Livermore, Calif., May 1973.
26. HULL, T. E., ENRIGHT, W. H., FELLEN, B. M., AND SEDGWICK, A. E. Comparing numerical methods for ordinary differential equations. *SIAM J. Numer. Anal. 9* (1972), 603–637.
27. KROGH, F. T. A variable step variable order multistep method for the numerical solution of ordinary differential equations. In Information Processing 68 (Proc. IFIP 68), Vol. I, A. J. H. Morrell, Ed., North-Holland, Amsterdam, 1969, pp. 194–199.
28. KROGH, F. T. Changing stepsize in the integration of differential equations using modified divided differences. Proc. of the Conf. on the Numerical Solution of Ordinary Differential Equations, U. of Texas at Austin, Oct. 19–20, 1972, D. G. Bettis, Ed., Lecture Notes in Mathematics, Vol. 362, Springer-Verlag, New York, 1974, pp. 22–71.
29. LAPIDUS, L., AND SEINFELD, J. H. *Numerical Solution of Ordinary Differential Equations.* Academic Press, New York, 1971.
30. LINIGER, W., AND WILLOUGHBY, R. Efficient integration methods for stiff systems of ordinary differential equations. *SIAM J. Numer. Anal. 7* (1970), 47–66.
31. MADSEN, N. K , AND SINCOVEC, R. F. The numerical method of lines for the solution of nonlinear partial differential equations. UCRL-75142, Lawrence Livermore Lab., U. of California, Livermore, Calif., Sept. 1973.
32. MÄKELA, M. On a generalized interpolation approach to the numerical integration of ordinary differential equations. *Ann. Acad. Sci. Fennicae,* Ser. A, I, 503 (1973), 1–43.

33. MILNE, W. E. A note on the numerical integration of differential equations. *J. Res. Nat. Bur. Stand. 43* (1949), 537–542.
34. MILNE, W. E. *Numerical Solution of Differential Equations*, Wiley, New York, 1953.
35. MILNE-THOMSON, L. M. *The Calculus of Finite Differences.* Macmillan, London, 1933.
36. NORDSIECK, A. On the numerical integration of ordinary differential equations. *Math. Computation 16* (1962), 22–49.
37. ODEH, F., AND LINIGER, W. A note on the unconditional fixed-h stability of linear multistep formulae *Computing 7* (1971), 240–253.
38. PIOTROWSKY, P. Stability, consistency and convergence of variable K-step methods. In Conf. on the Numerical Solution of Differential Equations, J. L. Morris, Ed., Lecture Notes in Mathematics, Vol. 109, Springer-Verlag, New York, 1969, pp. 221–227.
39. RICE, J. R. On the construction of polyalgorithms for automatic numerical analysis. In *Interactive Systems for Experimental Applied Mathematics*, M. Klerer and J. Reinfelds, Eds., Academic Press, New York, 1968, pp. 301–313.
40 RUBNER-PETERSON, T. An efficient algorithm using backward time-scaled differences for solving stiff differential-algebraic systems. Tech. U. of Denmark, 2800 Lyngby, Sept. 1973. Presented at the Seminar in Numerical Analysis, Royal Inst. of Technology, Stockholm, Oct. 10–12, 1973.
41. SCHECTER, R. S. *The Variational Method in Engineering.* McGraw-Hill, New York, 1967.
42. SEDGWICK, A. An efficient variable order variable step Adams method. Tech. Rep. 53, Dep. of Computer Sci, U. of Toronto, Toronto, Ont., Canada, May 1973. (Also Ph.D. Th., Dep. of Computer Sci., U. of Toronto, 1973.)
43. SHAMPINE, L. F., AND GORDON, M. K. *Computer Solution of Ordinary Differential Equations: Initial Value Problems.* In press, Freeman, San Francisco, Calif.
44. SHAMPINE, L. F., AND GORDON, M. K. Local error and variable order Adams codes. *Appl. Math. Computation*, to appear.
45. SINCOVEC, R. F. Private communication.
46. STETTER, H. J. Asymptotic expansions for the error of discretization algorithms for nonlinear functional equations. *Numer. Math. 7* (1965), 18–31.
47. TU, K.-W. Stability and convergence of general multistep and multivalue methods with variable step size. UIUCDCS-R-72-526, U. of Illinois, Urbana, Ill., July 1972. (Also Ph.D. Th., Dep. of Math., U. of Illinois, 1972.)

Received April 1974

ADDED IN PROOF. The norm now used in the error control algorithm in EPISODE is the root-mean-square norm, $\|v\| = \left[N^{-1} \sum_{1}^{N} (v^i)^2\right]^{1/2}$ for the N-vector v, rather than the maximum norm.