# An Alternative Implementation of Variable Step-Size Multistep Formulas for Stiff ODEs

K. R. JACKSON
Yale University
and
R. SACKS-DAVIS
Monash University, Australia

An alternative technique for the implementation of variable step-size multistep formulas is developed for the numerical solution of ordinary differential equations. Formulas based on this technique have the property that their leading coefficients are constant; this is important for methods which solve stiff systems. In addition, both theoretical and empirical results indicate that methods based on this technique have stability properties similar to those of the corresponding variable coefficient implementations. As a particular example, we have implemented the backward differentiation formulas in this form, and the numerical results look very promising.

Key Words and Phrases: stiff ODEs, variable step-size methods, backward differential formulas
CR Categories: 5.17

## 1. INTRODUCTION

Practical codes for solving the initial value problem

$$\dot{Y}(t) = f(t, Y(t)), \qquad Y(t_0) = Y_0 \tag{1.1}$$

vary the step size and possibly the order to take as large a step as possible consistent with a local control of the estimated error. Two techniques are commonly used to implement variable step-size multistep methods. One technique is based on fixed coefficient formulas and uses interpolation to generate approximations to the solution at evenly spaced past points; DIFSUB [10] and GEAR [14] are two well-known codes based on fixed coefficient formulas. The other technique uses variable coefficient formulas which do not require past values to be evenly spaced; EPISODE [3] and BDF [2] are examples of codes based on variable coefficient formulas. Each of the methods mentioned above


This work was supported by the National Research Council of Canada while the authors were at the University of Toronto.
Authors' addresses K. R. Jackson, Department of Computer Science, Yale University, New Haven, CT 06520; R. Sacks-Davis, Department of Computer Science, Monash University, Clayton, Victoria, Australia 3168.

contains an implementation of the backward differentiation formulas; these formulas are appropriate for solving stiff systems.

In deciding which of the two techniques should be used to implement a code, several factors must be considered. First, the amount of work that each implementation requires at each step differs. For a variable coefficient $k$-step multistep formula, the coefficients must be recalculated if the step size has changed during the past $k$ steps. Although the amount of work required to calculate the coefficients depends on $k$, it is independent of the number of equations in the system to be solved. On the other hand, for fixed coefficient formulas, the coefficients can be precomputed, but interpolation must be performed when the step size is changed. Although the interpolation can be performed very efficiently if a Nordsieck history array is used, the amount of work required to perform the interpolation is proportional to the number of equations in the system.

Second, both the theoretical and empirical results for the two techniques indicate that the variable coefficient implementations have better stability properties. For example, Gear, Tu, and Watanabe [13] point out that it is easy to construct examples for which a fixed coefficient implementation of an Adams method diverges even though the step-size and order selection strategies that it employs satisfy the easily accommodated restrictions which are sufficient to ensure that a variable coefficient Adams method converges. Thus a method based on variable coefficient formulas is better suited to take advantage of order and step-size changing opportunities to reduce the number of steps required to solve a problem.

For multistep codes designed to solve nonstiff problems, the trend during the past decade has been to use variable coefficient Adams formulas (cf. Krogh [17], Sedgwick [19], and Shampine and Gordon [20]). The numerical test results of Shampine, Watts, and Davenport [22] and Enright and Hull [8] indicate that these codes are the most efficient ones currently available for solving large classes of nonstiff problems.

However, when one considers methods designed to solve stiff problems, a third factor must be considered. At each step of the integration, an implicit equation of the form

$$\alpha_{n,0} y_n + h_n f(t_n, y_n) + c_n = 0 \tag{1.2}$$

must be solved for $y_n$, where $c_n$ contains all the remaining terms in the formula not involving $y_n$. Because the system is stiff, it is usual to use a modified Newton iteration scheme of the form

$$\hat{W}_n^s (y_n^s - y_n^{s+1}) = \alpha_{n,0} y_n^s + h_n f(t_n, y_n^s) + c_n \tag{1.3}$$

to solve eq. (1.2). The matrix $\hat{W}_n^s$ of eq. (1.3) is an approximation to the Newton iteration matrix

$$W_n^s = \alpha_{n,0} I + h_n f_y(t_n, y_n^s).$$

The iteration matrix, $\hat{W}_n^s$, is usually stored in its LU factored form and each update of $\hat{W}_n^s$ requires a Jacobian evaluation and an LU decomposition. Each iteration of the Newton scheme (1.3) requires a function evaluation as well as both a forward and a back substitution.

In a fixed coefficient implementation, $\alpha_{n,0}$ changes only when the order (i.e.,

the formula itself) is changed. Thus, for example, if the Jacobian $f_y(t, y)$ is constant and $\hat{W}_n^s$ is updated each time the step size or order is changed, then $\hat{W}_n^s = W_n^s$ throughout the integration.

On the other hand, for a $k$-step variable coefficient formula, $\alpha_{n,0}$ changes for $k$ steps following a change in step size, even if the step size is not changed again during those steps. Thus, although $f_y(t, y)$ may not change significantly (or at all) from step to step, $W_n^s$ may. Hence one would expect a method based on variable coefficient formulas to require more matrix factorizations and/or function evaluations than would be the case if the leading coefficients were constant.

Extensive test results for GEAR and EPISODE indicate that for moderately easy stiff problems GEAR requires significantly fewer matrix factorizations and function evaluations than EPISODE. However, EPISODE, being the more stable, is able to solve efficiently certain types of problems on which GEAR failed or performed poorly [4, 5, 7].

An alternative technique for implementing variable step-size backward differentiation formulas is developed in Section 2. It is easily seen that this technique applies to any class of formulas based on interpolation. (See [16] and [18] for two other examples.) Like the fixed coefficient formulas, these formulas possess the very important property that their leading coefficient $\alpha_{n,0}$ is constant; for this reason we refer to them as *fixed leading coefficient formulas.* However, like the variable coefficient formulas, the other coefficients depend on the ratios of step sizes used during the past steps.

An expression for the truncation error coefficient of a fixed leading coefficient backward differentiation formula is developed in Section 3, and we use this expression to develop an error estimate for the formula.

The stability of fixed leading coefficient formulas is discussed in Section 4. The empirical evidence and our limited theoretical results indicate that a method based on fixed leading coefficient backward differentiation formulas is more stable than a method based on fixed coefficient formulas, although not as stable as one based on variable coefficient backward differentiation formulas. On the other hand, for Adams formulas or the stiff second derivative formulas of Enright [6], theoretical results show that a variable step-size method based on fixed leading coefficient formulas is as stable as one based on variable coefficient formulas, and, consequently, more stable than a fixed coefficient implementation.

The modifications necessary to convert EPISODE, which uses a Nordsieck history array, to a code based on fixed leading coefficient formulas are discussed in Section 5. In making these modifications, we have attempted to change EPISODE as little as possible so that the comparison of the performance of these two codes will reflect only the difference in the implementation of the formulas.

Finally, our numerical test results are presented in Section 6. These results support our belief that this technique is very promising for the implementation of variable step-size multistep methods for stiff ODEs.

## 2. POLYNOMIAL BASIS

In this section we contrast the fixed leading coefficient implementation of the backward differentiation formulas with both the variable coefficient and the fixed coefficient implementations.

Suppose that $n-1$ steps of the integration have been completed and that it is required to advance a step from $t_{n-1}$ to $t_n$ with step size $h_n = t_n - t_{n-1}$. In the code EPISODE, which incorporates variable coefficient formulas, the predictor is based on the interpolatory polynomial $\pi_n^p(t)$ of degree $k$ or less which satisfies the $k+1$ conditions:

$$\begin{aligned} \pi_n^p(t_{n-i}) &= y_{n-i}, \qquad i = 1, \ldots, k; \\ \dot{\pi}_n^p(t_{n-1}) &= f_{n-1}. \end{aligned} \tag{2.1}$$

Here, $y_{n-i}$ is the previously computed approximation to the exact solution $Y(t_{n-i})$ at the point $t_{n-i}$, $i = 1, \ldots, k$, and $f_{n-1} \equiv f(t_{n-1}, y_{n-1})$.

It is not necessary, at this stage, to specify how $\pi_n^p(t)$ is represented; several alternatives, which include the use of divided differences, ordinate values, or a Nordsieck history array, are possible.

The strategy for advancing a step is to compute $y_n$ so that the corrector polynomial $\pi_n^c(t)$ of degree $k$ or less satisfies the following $k+2$ conditions:

$$\begin{aligned} \pi_n^c(t_{n-i}) &= y_{n-i}, \qquad i = 0, \ldots, k; \\ \dot{\pi}_n^c(t_n) &= f_n. \end{aligned} \tag{2.2}$$

Our definition of $\pi_n^c(t)$ is identical to the definition of $\pi_n(t)$ given in [3]. Note that if the order is not changed from step $n-1$ to step $n$, then $\pi_n^p(t) = \pi_{n-1}^c(t)$. Therefore, in this case, $\pi_n^p(t)$ actually satisfies $k+2$ conditions, and it could equally well be defined in terms of $y$-values only. However, when the order is decreased, $\pi_n^p(t)$ is not, in general, equal to $\pi_{n-1}^c(t)$, as is explained in more detail in [3, Section 2.4]. For this reason, we have chosen to define $\pi_n^p(t)$ independently of $\pi_{n-1}(t)$.

In the prediction stage of a variable coefficient implementation, the values

$$y_{n,0} \equiv \pi_n^p(t_n) \qquad \text{and} \qquad f_{n,0} \equiv \dot{\pi}_n^p(t_n) \tag{2.3}$$

are calculated. Then from eqs. (2.1), (2.2), and (2.3), we have

$$\pi_n^c(t) - \pi_n^p(t) = a(t) \cdot (y_n - y_{n,0}), \tag{2.4}$$

where $a(t)$ is the polynomial of degree $k$ that satisfies

$$\begin{aligned} a(t_{n-i}) &= 0, \qquad i = 1, \ldots, k; \\ a(t_n) &= 1. \end{aligned}$$

If we differentiate eq. (2.4) and evaluate the derivative at $t_n$, then we obtain the implicit equation

$$\hat{\alpha}_{n,0}(y_n - y_{n,0}) + h_n(f_n - f_{n,0}) = 0 \tag{2.5}$$

for $y_n$. Here

$$\hat{\alpha}_{n,0} = -h_n \dot{a}(t_n) = -\sum_{j=1}^{k} \frac{h_n}{t_n - t_{n-j}}.$$

Equation (2.5) is solved using a modified Newton scheme similar to eq. (1.3). We

see that the coefficient $\hat{\alpha}_{n,0}$ depends on each of the step sizes taken in the past $k$ steps.

An alternative way of writing eq. (2.5) is

$$\sum_{j=0}^{k} \hat{\alpha}_{n,j} y_{n-j} + h_n f_n = 0, \tag{2.6}$$

which follows from the interpolatory conditions (2.2) and is the usual way of writing a multistep formula.

In the fixed leading coefficient implementation of the backward differentiation formulas, the prediction stage is based on the polynomial $\omega_n^p(t)$ of degree $k$ or less that satisfies

$$\begin{aligned} \omega_n^p(t_{n-i}) &= y_{n-i}, \qquad i = 1, \ldots, k; \\ \dot{\omega}_n^p(t_{n-1}) &= f_{n-1}. \end{aligned} \tag{2.7}$$

The polynomial $\omega_n^p(t)$ satisfies the same interpolatory conditions as $\pi_n^p(t)$. Of course, the points that $\omega_n^p(t)$ interpolates are those generated by the fixed leading coefficient implementation.

On the other hand, the interpolatory conditions satisfied by the corrector polynomial associated with a fixed leading coefficient implementation differ from those satisfied by the corrector polynomial associated with a variable coefficient implementation. Specifically, the corrector polynomial $\omega_n^c(t)$ of degree $k$ or less satisfies the following $k + 2$ conditions:

$$\begin{aligned} \omega_n^c(t_n) &= y_n, \\ \dot{\omega}_n^c(t_n) &= f_n, \\ \omega_n^c(t_n - ih_n) &= \omega_n^p(t_n - ih_n), \qquad i = 1, \ldots, k. \end{aligned} \tag{2.8}$$

Therefore, $\omega_n^c(t)$ interpolates $\omega_n^p(t)$ at evenly spaced past points rather than interpolating the computed values, $\{y_{n-i}\}$, at the grid points, $\{t_{n-i}\}$, as $\pi_n^c(t)$ does.

In the prediction stage of a fixed leading coefficient implementation, the values

$$y_{n,0} \equiv \omega_n^p(t_n) \qquad \text{and} \qquad f_{n,0} \equiv \dot{\omega}_n^p(t_n) \tag{2.9}$$

are calculated. Then, from eqs. (2.7), (2.8), and (2.9), we have

$$\omega_n^c(t) - \omega_n^p(t) = b(t) \cdot (y_n - y_{n,0}), \tag{2.10}$$

where

$$\begin{aligned} b(t_n - ih_n) &= 0, \qquad i = 1, \ldots, k; \\ b(t_n) &= 1. \end{aligned}$$

If we differentiate eq. (2.10) and evaluate the derivative at $t_n$, we obtain the corrector equation for the fixed leading coefficient implementation

$$\alpha_0(y_n - y_{n,0}) + h_n(f_n - f_{n,0}) = 0, \tag{2.11}$$

where

$$\alpha_0 = -h_n \cdot \dot{b}(t_n) = -\sum_{j=1}^{k} \frac{1}{j}.$$

Unlike the variable coefficient implementation, the leading coefficient $\alpha_0$ used in the iteration matrix of the Newton scheme does not depend on the step-size sequence used in the integration, although it does depend, of course, on $k$ (i.e., the formula used).

Using eq. (2.8), we may write the corrector equation in the following form:

$$\alpha_0 y_n + \sum_{j=1}^{k} \alpha_j \omega_n^{\mathrm{p}}(t_n - jh_n) + h_n f_n = 0. \tag{2.12}$$

Written this way, the coefficients $\{\alpha_j\}$ are simply the constant coefficients of the $k$th-order fixed coefficient formula. Using the generalized Lagrange form of $\omega_n^{\mathrm{p}}(t)$, however, we may in turn write eq. (2.12) as

$$\alpha_0 y_n + \sum_{j=1}^{k} \alpha_{n,j} y_{n-j} + h_n f_n + h_n \beta_{n,1} f_{n-1} = 0. \tag{2.13}$$

We see that the leading coefficients do not depend on the previous step sizes, while the others do. The extra term, $h_n \beta_{n,1} f_{n-1}$, is due to our choice of $\omega_n^{\mathrm{p}}(t)$. If we had chosen to base our predictor on past $y$-values only, so that

$$\omega_n^{\mathrm{p}}(t_{n-i}) = y_{n-i}, \qquad i = 1, \ldots, k+1, \tag{2.14}$$

then eq. (2.13) would be replaced by

$$\alpha_0 y_n + \sum_{j=1}^{k+1} \alpha_{n,j}^{*} y_{n-j} + h_n f_n = 0. \tag{2.15}$$

We consider this possibility later.

The fixed leading coefficient implementation differs from the fixed coefficient implementation in one very important respect: The predictor polynomial for the fixed coefficient implementation interpolates interpolated past values, rather than the computed values $\{y_{n-j}\}$. That is, in a fixed coefficient implementation, the predictor polynomial $\theta_n^{\mathrm{p}}(t)$ of degree $k$ or less satisfies the $k+1$ conditions

$$\begin{aligned} \theta_n^{\mathrm{p}}(t_{n-1}) &= y_{n-1}, \\ \dot{\theta}_n^{\mathrm{p}}(t_{n-1}) &= f_{n-1}, \\ \theta_n^{\mathrm{p}}(t_{n-1} - ih_{n-1}) &= \theta_{n-1}^{\mathrm{p}}(t_{n-1} - ih_{n-1}), \qquad i = 1, \ldots, k-1. \end{aligned} \tag{2.16}$$

On the other hand, the corrector polynomial, $\theta_n^{\mathrm{c}}(t)$, for the fixed coefficient implementation is similar to the corrector polynomial, $\omega_n^{\mathrm{c}}(t)$, for the fixed leading coefficient implementation in that $\theta_n^{\mathrm{c}}(t)$ is the polynomial of degree $k$ or less that satisfies

$$\begin{aligned} \theta_n^{\mathrm{c}}(t_n) &= y_n, \\ \dot{\theta}_n^{\mathrm{c}}(t_n) &= f_n, \\ \theta_n^{\mathrm{c}}(t_n - ih_n) &= \theta_n^{\mathrm{p}}(t_n - ih_n), \qquad i = 1, \ldots, k. \end{aligned} \tag{2.17}$$

(It should be noted that in some implementations eq. (2.16) may not be satisfied

when the order is changed. This point is discussed for Adams formulas by Shampine and Gordon [21].)

Like the variable coefficient implementation, the predictor polynomial is equal to the corrector polynomial from the previous step (i.e., $\theta_n^p(t) = \theta_{n-1}^c(t)$), provided that the order is not changed at step $n$. The fixed leading coefficient implementation differs from both of the other implementations in this respect.

If, in a fixed coefficient implementation, the step size is changed at every step, then the corrector equation,

$$\alpha_0 y_n + \sum_{j=1}^{k} \alpha_j \theta_n^p(t_n - jh_n) + h_n f_n = 0,$$

may be expressed in the form

$$\alpha_0 y_n + \sum_{j=1}^{n} \tilde{\alpha}_{n,j} y_{n-j} + h_n f_n + h_n \sum_{j=1}^{n-2} \tilde{\beta}_{n,j} f_{n-j} = 0, \tag{2.18}$$

provided, as well, that the order $k$ is always kept greater than two. Thus the solution at $t_n$ may depend on all of the previously computed values. Consequently, eq. (2.18) is not a "local" formula as are eqs. (2.6), (2.13), and (2.15). This gives one an intuitive understanding for the cause of the numerical instability of fixed coefficient formulas with respect to step-size changes.

## 3. ESTIMATION OF THE TRUNCATION ERROR

In this section we derive estimates of the truncation error for the fixed leading coefficient formulas. Consistent with the notation of Byrne and Hindmarsh [3], we define the truncation error of the $k$th order formula (2.13) to be

$$E_n(k) = \frac{[\alpha_0 Y(t_n) + \sum_{j=1}^{k} \alpha_{n,j} Y(t_{n-j}) + h_n \dot{Y}(t_n) + h_n \beta_{n,1} \dot{Y}(t_{n-1})]}{\alpha_0},$$

where $Y(t)$ is the exact solution of eq. (1.1). Throughout this section, we assume that $Y(t)$ is in $C^{k+2}$ and the ratios of the successive step sizes, $h_n/h_{n-1}$, are uniformly bounded above.

We define $\Omega_n^p(t)$ to be the polynomial of degree $k$ or less that satisfies

$$\Omega_n^p(t_{n-i}) = Y(t_{n-i}), \qquad i = 1, \ldots, k;$$
$$\dot{\Omega}_n^p(t_{n-1}) = \dot{Y}(t_{n-1}).$$

Similarly, we define $\Omega_n^c(t)$ to be the polynomial of degree $k$ or less that satisfies

$$\Omega_n^c(t_n) = Y(t_n),$$
$$\Omega_n^c(t_n - ih_n) = \Omega_n^p(t_n - ih_n), \qquad i = 1, \ldots, k.$$

Of course, the polynomials $\Omega_n^p(t)$ and $\Omega_n^c(t)$ are analogous to the polynomials $\omega_n^p(t)$ and $\omega_n^c(t)$, respectively, considered in the previous section, the difference being that $\Omega_n^p(t)$ and $\Omega_n^c(t)$ are based on the exact values $\{Y(t_{n-i})\}$ rather than the computed values $\{y_{n-i}\}$.

From the interpolatory conditions for $\Omega_n^p(t)$, it follows that $E_n(k)$ may be rewritten as

$$E_n(k) = \frac{[\alpha_0 Y(t_n) + \sum_{j=1}^{k} \alpha_{n,j} \Omega_n^p(t_{n-j}) + h_n \dot{Y}(t_n) + h_n \beta_{n,1} \dot{\Omega}_n^p(t_{n-1})]}{\alpha_0}.$$

But remembering that the coefficients $\{\alpha_{n,j}\}$ come from rewriting eq. (2.12) as eq. (2.13) using the generalized Lagrange form of $\omega_n^p(t)$, we have that

$$E_n(k) = \frac{[\alpha_0 Y(t_n) + \sum_{j=1}^{k} \alpha_j \Omega_n^p(t_n - jh_n) + h_n \dot{Y}(t_n)]}{\alpha_0},$$

where the $\{\alpha_j\}$ are the coefficients of the fixed step-size backward differentiation formula. Now, from the interpolation conditions for $\Omega_n^c(t)$, it follows that this last equation may be rewritten as

$$E_n(k) = \frac{[\sum_{j=0}^{k} \alpha_j \Omega_n^c(t_n - jh_n) + h_n \dot{Y}(t_n)]}{\alpha_0}.$$

But from the derivation of eq. (2.12) from eq. (2.8), it follows that this equation may, in turn, be rewritten more concisely as

$$E_n(k) = \frac{[h_n \dot{Y}(t_n) - h_n \dot{\Omega}_n^c(t_n)]}{\alpha_0}.$$

In the analysis that follows, it is useful to express this last equation as the sum of two differences:

$$E_n(k) = \frac{[h_n \dot{Y}(t_n) - h_n \dot{\Omega}_n^p(t_n)]}{\alpha_0} + \frac{[h_n \dot{\Omega}_n^p(t_n) - h_n \dot{\Omega}_n^c(t_n)]}{\alpha_0}.$$

From the interpolatory conditions for $\Omega_n^p(t)$, it follows that

$$Y(t) - \Omega_n^p(t) = p(t) \cdot Y[t, t_{n-1}, t_{n-1}, t_{n-2}, \ldots, t_{n-k}],$$

where

$$p(t) = (t - t_{n-1})^2 (t - t_{n-2}) \cdots (t - t_{n-k}).$$

Therefore,

$$\dot{Y}(t) - \dot{\Omega}_n^p(t) = \dot{p}(t) \cdot Y[t, t_{n-1}, t_{n-1}, t_{n-2}, \ldots, t_{n-k}] + p(t) \cdot \dot{Y}[t, t_{n-1}, t_{n-1}, t_{n-2}, \ldots, t_{n-k}].$$

But

$$\dot{Y}[t, t_{n-1}, t_{n-1}, t_{n-2}, \ldots, t_{n-k}] = Y[t, t, t_{n-1}, t_{n-1}, t_{n-2}, \ldots, t_{n-k}].$$

Consequently,

$$h_n \dot{Y}(t_n) - h_n \dot{\Omega}_n^p(t_n) = h_n \cdot \dot{p}(t_n) \cdot Y[t_n, t_{n-1}, t_{n-1}, t_{n-2}, \ldots, t_{n-k}] + h_n \cdot p(t_n) \cdot Y[t_n, t_n, t_{n-1}, t_{n-1}, t_{n-2}, \ldots, t_{n-k}].$$

Similarly, from the interpolatory conditions for $\Omega_n^c(t)$, it follows that

$$\Omega_n^p(t) - \Omega_n^c(t) = \frac{t - t_n + h_n}{h_n} \cdots \frac{t - t_n + kh_n}{kh_n} \cdot [\Omega_n^p(t_n) - \Omega_n^c(t_n)].$$

Hence

$$h_n\dot{\Omega}_n^{\mathrm{p}}(t_n) - h_n\dot{\Omega}_n^{\mathrm{c}}(t_n) = \left[1 + \frac{1}{2} + \cdots + \frac{1}{k}\right][\Omega_n^{\mathrm{p}}(t_n) - \Omega_n^{\mathrm{c}}(t_n)]$$
$$= -\alpha_0[\Omega_n^{\mathrm{p}}(t_n) - \Omega_n^{\mathrm{c}}(t_n)].$$

But $\Omega_n^{\mathrm{c}}(t_n) = Y(t_n)$. Consequently,

$$\Omega_n^{\mathrm{p}}(t_n) - \Omega_n^{\mathrm{c}}(t_n) = \Omega_n^{\mathrm{p}}(t_n) - Y(t_n)$$
$$= -p(t_n) \cdot Y[t_n, t_{n-1}, t_{n-1}, t_{n-2}, \ldots, t_{n-k}].$$

Thus

$$h_n\dot{\Omega}_n^{\mathrm{p}}(t_n) - h_n\dot{\Omega}_n^{\mathrm{c}}(t_n) = \alpha_0 \cdot p(t_n) \cdot Y[t_n, t_{n-1}, t_{n-1}, t_{n-2}, \ldots, t_{n-k}].$$

Combining these results, we have

$$E_n(k) = \frac{h_n\dot{p}(t_n) + \alpha_0 p(t_n)}{\alpha_0} \cdot Y[t_n, t_{n-1}, t_{n-1}, t_{n-2}, \ldots, t_{n-k}]$$
$$+ \frac{h_n p(t_n)}{\alpha_0} \cdot Y[t_n, t_n, t_{n-1}, t_{n-1}, t_{n-2}, \ldots, t_{n-k}]. \tag{3.1}$$

However, since we have assumed that $Y(t)$ is in $C^{k+2}$, this can be rewritten as

$$E_n(k) = C_n(k)h_n^{k+1}Y^{(k+1)}(t_n) + O(h_{n-k}^{k+2}), \tag{3.2}$$

where

$$C_n(k) = \frac{h_n\dot{p}(t_n) + \alpha_0 p(t_n)}{\alpha_0 h_n^{k+1}(k+1)!}.$$

Letting $\xi_i = (t_n - t_{n-i})/h_n$, it follows that

$$C_n(k) = \frac{\xi_1 \cdots \xi_k}{(k+1)!} \frac{(1 - \hat{\alpha}_{n,0} + \alpha_0)}{\alpha_0}, \tag{3.3}$$

where we have used the fact that

$$\hat{\alpha}_{n,0} = -\left(1 + \frac{1}{\xi_2} + \cdots + \frac{1}{\xi_k}\right)$$

and

$$\alpha_0 = -\left(1 + \frac{1}{2} + \cdots + \frac{1}{k}\right).$$

It should be noted that $C_n(k)$ may be zero for $k \geq 4$. On the other hand, the corresponding coefficient never vanishes for the variable coefficient implementation (see [3, eq. (2.34)]). This disadvantage, however, does not seem to be very serious since $C_n(k)$ can vanish only after a drastic decrease in step size (in which case it might be prudent to use a lower order formula). Moreover, if $C_n(k)$ does vanish, then the coefficient of the first divided difference in eq. (3.1) becomes zero also and the truncation error is given simply by the second term in that

expression, the coefficient of which never vanishes. Consequently, one could use an approximation to this higher order term in an error estimate. However, to conform with EPISODE, we have not taken this approach, but have instead based our error estimate on eqs. (3.2) and (3.3) alone.

Of course, to use eq. (3.2), an approximation to $h_n^{k+1}Y^{(k+1)}(t_n)$ must be developed. Again, to conform with EPISODE, we have used a modified Milne error estimate, although we present an alternative later.

The modified Milne estimate is based on the difference between predicted and corrected values. From eqs. (2.7) and (2.9), the predicted value, $y_{n,0}$, satisfies an explicit equation of the form

$$\bar{\alpha}_{n,0}y_{n,0} + \sum_{j=1}^{k} \bar{\alpha}_{n,j}\, y_{n-j} + h_n f_{n-1} = 0.$$

Here,

$$\bar{\alpha}_{n,0} = \frac{-\xi_1(\xi_2 - 1) \cdots (\xi_k - 1)}{\xi_2 \cdots \xi_k}. \tag{3.4}$$

The truncation error of the formula may be written as

$$\bar{E}_n(k) = \bar{C}_n(k)h_n^{k+1}Y^{(k+1)}(t_n) + O(h_{n-k}^{k+2}),$$

where

$$\bar{C}_n(k) = \frac{\xi_1 \cdots \xi_k}{(k+1)!}. \tag{3.5}$$

Since both the fixed leading coefficient and variable coefficient implementations use predictor polynomials that satisfy the same interpolatory conditions, the derivation of eqs. (3.4) and (3.5) may be found in [3].

Using this notation, we have the following asymptotic relation due to Gear [11] for $y_n - y_{n,0}$ which is valid for constant step size (and even slightly more general conditions):

$$y_n - y_{n,0} = [\bar{C}_n(k) - d_n(k)C_n(k)]h_n^{k+1}Y^{(k+1)}(t_n) + O(h_{n-k}^{k+2}), \tag{3.6}$$

where, in our case,

$$d_n(k) = \frac{\alpha_0}{\bar{\alpha}_{n,0}(1 + \beta_{n,1})}. \tag{3.7}$$

When the step size is constant, $\beta_{n,1} = 0$ and eq. (3.7) reduces to

$$d_n(k) = -k\alpha_0. \tag{3.8}$$

Combining eqs. (3.2), (3.6), and (3.8) gives us our estimate of the truncation error,

$$\mathrm{Est}_n(k) = \frac{C_n(k)}{\bar{C}_n(k) + k\alpha_0 C_n(k)}\,(y_n - y_{n,0}).$$

which is asymptotically correct under the assumption of constant step size.

The reason we chose to use eq. (3.7) evaluated for constant step size in our

estimate is that it may happen that $\beta_{n,1} = -1$, and the error estimate based on eq. (3.7) becomes zero.[1]

Our estimates of the truncation error at orders $k - 1$ and $k + 1$ require approximations to $h_n^k Y^{(k)}(t_n)$ and $h_n^{k+2} Y^{(k+2)}(t_n)$, respectively. We have used the same strategies as are used in EPISODE to obtain these approximations.

It should be pointed out that other estimates that are also asymptotically correct under the same conditions as the modified Milne error estimate can be used if slightly different strategies are adopted. For example, Gear [11] proved that estimates of $h_n^{k+1} Y^{(k+1)}(t_n)$ based upon the divided difference

$$[y_n, y_{n-1}, y_{n-2}, \ldots, y_{n-k-1}] \tag{3.9}$$

are asymptotically correct as well. If the predictor is based only on past $y$-values, then this difference is readily available.

Let $\phi_n(t)$ be the polynomial of degree $k + 1$ satisfying

$$\phi_n(t_{n-i}) = y_{n-i}, \qquad i = 0, \ldots, k + 1.$$

Then, if $\omega_n^p(t)$ is given by eq. (2.14),

$$\phi_n(t) - \omega_n^p(t) = (t - t_{n-1}) \cdots (t - t_{n-k-1})[y_n, y_{n-1}, \ldots, y_{n-k-1}].$$

Evaluating this equation at $t_n$ we obtain

$$y_n - y_{n,0} = \xi_1 \cdots \xi_{k+1} h_n^{k+1} [y_n, y_{n-1}, \ldots, y_{n-k-1}]. \tag{3.10}$$

In the previous section, we considered the possibility of using a predictor based only on past $y$-values in a fixed leading coefficient implementation. The corrector formula would be of the form of eq. (2.15) and an error estimate based on eq. (3.10) would be available.

A similar error estimate is also available for variable coefficient implementations. Recall from eq. (2.2) that the corrector is based on a polynomial of degree $k$ satisfying $k + 2$ conditions. If the order is not changed, then this polynomial is used in the next step to predict $y_{n,0}$. In this case, we can interpret the polynomial $\pi_n^p(t)$ as being based only on past $y$-values. Then eq. (3.10) holds, and we have a simple alternative to the modified Milne error estimate. The only modifications to EPISODE necessary to incorporate this estimate are in the strategies for changing order and for starting.

Interestingly, if either the variable coefficient or the fixed leading coefficient formulas are implemented in Nordsieck or divided difference form, then an

[1] One of the referees suggested that we use $\alpha_0 E_n(k)/(1 + \beta_{n,1})$ as our estimate of the error, rather than $E_n(k)$. (We used the latter because it is similar to the estimate used in EPISODE.) The former, however, is a "normalized" error estimate corresponding to an equivalent formula for which the sum of the beta coefficients is one (See, for example, Gear [10, Section 8.1] for a discussion of normalization.) The suggested estimate has the advantage that it does not vanish for $\beta_{n,1} = -1$. In fact, as the referee points out, this error estimate reduces to

$$\frac{\alpha_0 E_n(k)}{1 + \beta_{n,1}} = \frac{-\bar{\alpha}_{n,0}(1 + \alpha_0 - \hat{\alpha}_{n,0})}{1 - \bar{\alpha}_{n,0}} (y_n - y_{n,0}),$$

after noting that $\hat{\alpha}_{n,0} + \beta_{n,1}\bar{\alpha}_{n,0} = \alpha_0$.

estimate based on the divided difference (3.9) is available even when the predictor is defined by eq. (2.1) or eq. (2.7), respectively. In this case, however, the implementation is more complicated.

We have that

$$\begin{aligned}[y_n, y_{n-1}, y_{n-2}, \ldots, y_{n-k-1}](t_n - t_{n-k-1}) \\ = [y_n, y_{n-1}, y_{n-1}, y_{n-2}, \ldots, y_{n-k}](t_n - t_{n-1}) \\ + [y_{n-1}, y_{n-1}, y_{n-2}, \ldots, y_{n-k}] \\ - [y_{n-1}, y_{n-2}, \ldots, y_{n-k-1}],\end{aligned}$$

which may be verified by writing $[y_n, y_{n-1}, y_{n-1}, y_{n-2}, \ldots, y_{n-k}]$ as $[y_n, y_{n-1}, y_{n-2}, \ldots, y_{n-k}, y_{n-1}]$. If the previous step was taken at order $k$, then $[y_{n-1}, y_{n-2}, \ldots, y_{n-k-1}]$ is available from the error estimate of lower order, $[y_{n-1}, y_{n-1}, y_{n-2}, \ldots, y_{n-k}]$ is the last scaled derivative in the Nordsieck array, and $[y_n, y_{n-1}, y_{n-1}, y_{n-2}, \ldots, y_{n-k}]$ is equal to $(y_n - y_{n,0})/(h_n^{k+1}\xi_1 \cdots \xi_k)$.

## 4. STABILITY

It is extremely difficult to give theoretical results about the stability of the backward differentiation formulas in the general setting of arbitrary step sizes (cf. Brayton and Conley [1]). The numerical results of Tu [23] and Brayton, Gustavson, and Hachtel [2] indicate that the variable coefficient formulas perform much better than the fixed coefficient formulas with respect to numerical stability when the step size is changing frequently; it was primarily for this reason that Byrne and Hindmarsh developed EPISODE.

Our test results in Section 6 seem to indicate that the fixed leading coefficient implementation does not suffer from the numerical instability that is characteristic of the fixed coefficient implementation. Moreover, we performed some elementary tests similar to those reported in [23] to gain a crude comparison of the numerical stability of the variable step-size implementations discussed in Section 2.

For this crude comparison, we considered the test equation $\dot{Y} = \lambda Y$ and the step-size sequence $h_{2n} = h$, $h_{2n+1} = rh$ for $n = 0, 1, \ldots$, where $h$ is the fundamental step size used in the integration and $r$ and $1/r$ are the step-size ratios. Since this is a linear homogeneous problem, one step of a variable step-size backward differentiation formula can be represented in matrix form as

$$\mathbf{z}_{2n+1} = \mathbf{S}(h\lambda, r)\mathbf{z}_{2n}$$

or

$$\mathbf{z}_{2n+2} = \mathbf{S}(h\lambda, 1/r)\mathbf{z}_{2n+1},$$

where $\mathbf{z}$ contains some representation of the past data (such as ordinate values, scaled derivatives, or divided differences). Consequently, a double step can be represented as

$$\mathbf{z}_{2n+2} = \mathbf{SD}(h\lambda, r)\mathbf{z}_{2n},$$

where

$$\mathbf{SD}(h\lambda, r) = \mathbf{S}(h\lambda, 1/r)\mathbf{S}(h\lambda, r).$$

Thus we can consider $\mathbf{SD}(h\lambda, r)$ as the underlying matrix for a fixed step-size "double-step" formula. Consequently, for this test equation and step-size sequence, a method is absolutely stable if and only if all of the eigenvalues of $\mathbf{SD}(h\lambda, r)$ have magnitude less than one, and it is strongly $h \to 0$ stable if and only if all of the eigenvalues of $\mathbf{SD}(0, r)$ have magnitude less than one except for the principal eigenvalue (which is equal to one). As we are only interested in the eigenvalues of $\mathbf{SD}(h\lambda, r)$, the representation that we choose for the past data is immaterial, since, if two representations are related by a linear transformation (as are all commonly used representations—see Gear [10, Section 9.2]), then their associated matrices $\mathbf{SD}(h\lambda, r)$ are related by a similarity transformation. Consequently, the eigenvalues of these related matrices are identical.

For several values of $h\lambda$, orders 2 through 6, and the following four variable step-size implementations of backward differentiation formulas:

(1) VC—the variable coefficient implementation,
(2) FL$y$—the fixed leading coefficient implementation based on the predictor eq. (2.14) that uses $y$-values only,
(3) FL$f$—the fixed leading coefficient implementation based on the predictor eq. (2.7) that uses an $f$-value, and
(4) FC—the fixed coefficient implementation,

Table I lists the smallest computed value of $r$ greater than one for which $\mathbf{SD}(h\lambda, r)$ has an eigenvalue that is greater than or equal to one in magnitude. (In the case of $h\lambda = 0$, $\mathbf{SD}(0, r)$ was first deflated to remove the principal eigenvalue that is equal to one and its associated eigenvector.) For values of $-h\lambda$ between 0 and $10^{-3}$, the values of $r$ do not appear to vary significantly, as one would expect. Similarly, for values of $-h\lambda$ greater than $10^2$, the values of $r$ are approximately equal to those values listed for $-h\lambda = 10^2$.

This limited numerical evidence supports our belief that the fixed leading coefficient implementation based on the predictor eq. (2.14) tends to be more stable than the associated fixed coefficient implementation, particularly for orders 3 and 4, and especially when $-h\lambda$ is large. For stiff methods, of course, it is very important that a method remain absolutely stable for a wide range of values of $-h\lambda$.

For small values of $-h\lambda$, neither fixed leading coefficient implementation seems to be as stable as the variable coefficient implementation. However, for large values of $-h\lambda$, the fixed leading coefficient implementation based on the predictor eq. (2.14) that uses $y$-values only appears to be almost as stable as the variable coefficient implementation for this test problem and step-size sequence. These empirical results also tend to give some support for a preference for the predictor eq. (2.14) over the predictor eq. (2.7) as the basis for the fixed leading coefficient implementation.

For Adams formulas, however, we do have conclusive theoretical results which indicate that fixed leading coefficient implementations of these formulas are as stable as variable coefficient implementations and, consequently, more stable than fixed coefficient implementations. (Similar results can be developed for the second derivative formulas of Enright [6].) In particular, Jackson [16] shows that, for both the fixed leading coefficient and the variable coefficient implementations

Table I

| | | $-h\lambda$ | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Order | Method | 0 | $10^{-3}$ | $10^{-2}$ | $10^{-1}$ | $10^{0}$ | $10^{1}$ | $10^{2}$ |
| 2 | VC | — | — | — | — | — | — | — |
| 2 | FL $y$ | — | — | — | — | — | — | — |
| 2 | FL $f$ | — | — | — | 73.67 | 11.93 | 6.337 | 5.877 |
| 2 | FC | — | — | — | 73.67 | 11.93 | 6.337 | 5.877 |
| 3 | VC | — | — | — | 52.16 | — | — | — |
| 3 | FL $y$ | — | — | — | — | — | — | — |
| 3 | FL $f$ | — | — | — | 16.09 | 3.644 | 2.472 | 2.452 |
| 3 | FC | 6.172 | 6.163 | 6.090 | 5.408 | 3.076 | 2.290 | 2.272 |
| 4 | VC | — | — | 39.74 | 12.67 | 6.178 | — | — |
| 4 | FL $y$ | 3.617 | 3.613 | 3.587 | 3.366 | 2.787 | — | — |
| 4 | FL $f$ | 2 640 | 2.638 | 2.624 | 2.488 | 1 815 | 1.602 | 1 654 |
| 4 | FC | 2.366 | 2.365 | 2.359 | 2.287 | 1.809 | 1.611 | 1.661 |
| 5 | VC | 6.289 | 6.267 | 6.083 | 4.898 | 2.766 | — | — |
| 5 | FL $y$ | 1.720 | 1.719 | 1.714 | 1.674 | 1.456 | 24.77 | — |
| 5 | FL $f$ | 1.585 | 1.585 | 1.581 | 1.549 | 1.324 | 1.348 | 1.414 |
| 5 | FC | 1.595 | 1.595 | 1.592 | 1.564 | 1.347 | 1.374 | 1.440 |
| 6 | VC | 2.501 | 2.499 | 2.478 | 2.287 | 1.411 | — | — |
| 6 | FL $y$ | 1.225 | 1.225 | 1.224 | 1.208 | 1.087 | 4.231 | — |
| 6 | FL $f$ | 1.227 | 1.227 | 1.225 | 1.210 | 1.081 | 1.244 | 1.287 |
| 6 | FC | 1.263 | 1.263 | 1.261 | 1.244 | 1.095 | 1.294 | 1.343 |

*Note.* A dash in place of a number in this table indicates that no value of $r$ between 1 and 100 was found for which $\mathbf{SD}(h\lambda, r)$ has an eigenvalue that is greater than or equal to one in magnitude, where again the special case $h\lambda = 0$ was treated by first deflating $\mathbf{SD}(0, r)$, as mentioned above.

of the Adams formulas, the only restrictions on the order and step-size selection strategies necessary to ensure convergence are that both the order of the formulas and the step-size ratios, $h_n/h_{n-1}$, used throughout the integration be bounded above. Implementations of the Adams formulas based on the fixed coefficient technique are, in general, unstable under these conditions (see Tu [23] or Gear and Tu [12].)

We conclude this section by giving some theoretical results about the stability of the fixed leading coefficient implementation of the backward differentiation formulas when applied to the test equation $\dot{Y} = \lambda Y$ with large values of $|h_n\lambda|$; these results concur with our earlier numerical observations.

If the step size has been constant for $k$ steps, then all of the variable step-size implementations that we have considered reduce to the $k$-step constant coefficient backward differentiation formula. The regions of absolute stability of these formulas are well known (see [10]). In particular, the formulas of orders 1 through 6 are stiffly stable.

When the step size is changing, however, the coefficient $\beta_{n,1}$ of eq. (2.13) is not necessarily zero, and it is even possible that $|\beta_{n,1}| > 1$. Hence, on a particular step, one of the roots of the characteristic equation

$$\alpha_0\zeta^k + \sum_{j=1}^{k} \alpha_{n,j}\zeta^{k-j} + h_n\lambda\zeta^k + h_n\lambda\beta_{n,1}\zeta^{k-1} = 0$$

may have modulus greater than 1. Thus it seems unlikely that the associated formula can be shown to be stiffly stable (in a generalized sense) unless the step-size changes are severely restricted.

On the other hand, if the predictor is based on $y$-values only, then the corrector equation is given by eq. (2.15), and we can guarantee its stability for large values of $|h_n\lambda|$. That is, if this fixed leading coefficient formula is applied to the test problem $\dot{Y} = \lambda Y$ with initial values satisfying $|y_i| \le M$, $i = 0, 1, \ldots, k$, and with step sizes $h_n$ such that, for some constant $\alpha$, $|\alpha_0| \le \alpha$, $|\alpha^*_{n,i}| \le \alpha$ for all $n$ and $i$ and $|h_n\lambda| \ge D \ge (k+2)\alpha$ for all $n$, then it may be shown that the values $y_n$ satisfy

$$|y_n| \le \left[\frac{\alpha(k+1)}{D-\alpha}\right]^{\lfloor n/(k+1)\rfloor} M, \qquad n = k+1, k+2, \ldots,$$

where $\lfloor a \rfloor$ is the largest integer less than or equal to $a$. Moreover, it is possible to guarantee that $|\alpha^*_{n,i}| \le \alpha$ for some $\alpha$, provided that the step-size ratios are bounded above. Thus the sequence $\{y_n\}$ tends to zero for $|h_n\lambda|$ sufficiently large. This is similar to what Byrne and Hindmarsh prove for the variable coefficient formulas (although there appears to be a slight error in the exponent of their equation [3, eq. (2.50)]).

## 5. IMPLEMENTATION OF NORDSIECK FORM

In this section we show how to implement the fixed leading coefficient backward differentiation formulas in a method that uses a Nordsieck history array to store past data. As much as possible, we use the notation of Byrne and Hindmarsh [3] and contrast our method to their implementation of the variable coefficient formulas in EPISODE.

Advancing a step from $t_{n-1}$ to $t_n$ can be viewed as updating the interpolatory polynomial $\omega^p_n(t)$ of eq. (2.7) to the polynomial $\omega^p_{n+1}(t)$ satisfying a similar set of $k+1$ conditions at $t_n$. In Nordsieck form, the polynomial $\omega^p_n(t)$ is represented by an array of its scaled derivatives

$$\mathbf{z}_{n-1} = \left[y_{n-1}, h_n y^{(1)}_{n-1}, \frac{h_n^2 y^{(2)}_{n-1}}{2!}, \ldots, \frac{h_n^k y^{(k)}_{n-1}}{k!}\right],$$

where $y^{(i)}_{n-1}$ is the $i$th derivative of $\omega^p_n(t)$ evaluated at $t_{n-1}$.

The first step of the algorithm is the calculation of the $i$th derivative of $\omega^p_n(t)$ evaluated at $t_n$ for $i = 0, \ldots, k$. In particular, the values

$$y_{n,0} \equiv \omega^p_n(t_n) \qquad \text{and} \qquad f_{n,0} \equiv \dot{\omega}^p_n(t_n) \tag{5.1}$$

must be calculated. The computation of the predicted values is achieved by forming

$$\mathbf{z}_{n,0} = \mathbf{z}_{n-1}\mathbf{A}(k),$$

where $\mathbf{A}(k)$ is the $(k+1) \times (k+1)$ Pascal matrix. Let

$$\Delta_n(t) = \omega^p_{n+1}(t) - \omega^p_n(t).$$

In order to determine $\mathbf{z}_n$ from the vector of predicted values, $\mathbf{z}_{n,0}$, we have to calculate the scaled derivatives of $\Delta_n(t)$ evaluated at $t_n$. From eqs. (2.7) and (5.1),

it follows that

$$\Delta_n(t) = c(t) \cdot (y_n - y_{n,0}) + d(t) \cdot (f_n - f_{n,0}),$$

where $c(t)$ and $d(t)$ are polynomials of degree $k$ satisfying

$$c(t_n) = 1,$$
$$\dot{c}(t_n) = 0,$$
$$c(t_{n-i}) = 0, \qquad i = 1, \ldots, k-1;$$

and

$$d(t_n) = 0,$$
$$\dot{d}(t_n) = 1,$$
$$d(t_{n-i}) = 0, \qquad i = 1, \ldots, k-1.$$

Hence from eq. (2.11),

$$\Delta_n(t) = e(t) \cdot (y_n - y_{n,0}),$$

where

$$e(t) = \frac{c(t) - \alpha_0 d(t)}{h_n}.$$

Let

$$q_i(t) = \prod_{j=1}^{i} (t - t_{n-j}), \qquad i \geq 1;$$
$$q_0(t) = 1.$$

From the interpolatory properties of $c(t)$ and $d(t)$, it is easily shown that

$$e(t) = \frac{q_{k-1}(t)}{q_{k-1}(t_n)} \cdot \{1 + \mu_n(t - t_n)\},$$

where

$$\mu_n = \frac{-\alpha_0}{h_n} - \frac{\dot{q}_{k-1}(t_n)}{q_{k-1}(t_n)}.$$

(Contrast this with the variable coefficient case where $e(t) = q_k(t)/q_k(t_n)$.)

As previously mentioned, we let

$$\xi_i = \frac{t_n - t_{n-i}}{h_n}, \qquad i = 1, \ldots, k,$$

but we also define

$$\xi_k^* = \frac{1}{\mu_n h_n}.$$

Now, consider the change in variable

$$x = \frac{t - t_n}{h_n}$$

and let

$$\Lambda_n(x) = \left(1 + \frac{x}{\xi_k^*}\right) \prod_{j=1}^{k-1} \left(1 + \frac{x}{\xi_j}\right) = \sum_{j=0}^{k} \ell_j x^j. \tag{5.2}$$

The coefficients $\{\ell_j\}$ in eq. (5.2) are computed and stored in the vector

$$\ell = [\ell_0, \ell_1, \ldots, \ell_k].$$

We have then

$$h_n^i \Delta_n^{(i)}(t_n) = \frac{\Lambda_n^{(i)}(0)(y_n - y_{n,0})}{i!} = \ell_i \cdot (y_n - y_{n,0}),$$

so that

$$\mathbf{z}_n = \mathbf{z}_{n,0} + \ell \cdot (y_n - y_{n,0}). \tag{5.3}$$

The only difference between this scheme and the one used in EPISODE to calculate $\mathbf{z}_n$ from $\mathbf{z}_{n,0}$ is in the calculation of the normalized coefficients $\{\ell_j\}$, and here the only difference is that $\xi_k^*$ is used in place of $\xi_k$. (Of course, if the step size is constant for $k$ steps, then $\xi_k^* = \xi_k$.) As in EPISODE, the solution, $y_n$, is calculated from the relation given by the second component of eq. (5.3). Namely,

$$h_n f_n = h_n f_{n,0} + \ell_1 \cdot (y_n - y_{n,0}).$$

It should be noted that in the fixed leading coefficient implementation (but not in the variable coefficient implementation) $\ell_1 = -\alpha_0$. Thus $\ell_1$ changes only when the order is changed.

It remains only to describe the adjustments to the array $\mathbf{z}_n$ that are necessary when changing order or step size. The adjustments when decreasing order are the same as in the variable coefficient case. When increasing order, however, the array $\mathbf{z}_n$ must be modified to become an array $\mathbf{z}_n^*$ based on the polynomial $\omega_{n+1}^*(t)$ of degree $k + 1$ defined by

$$\omega_{n+1}^*(t_{n-i}) = y_{n-i}, \qquad i = 0, \ldots, k;$$
$$\dot{\omega}_{n+1}^*(t_n) = f_n.$$

Then

$$\omega_{n+1}^*(t) - \omega_{n+1}^{\mathrm{p}}(t) = c \cdot (t - t_n)^2 (t - t_{n-1}) \cdots (t - t_{n-k+1}),$$

where

$$c = [y_n, y_n, y_{n-1}, \ldots, y_{n-k}].$$

In the variable coefficient case, $c = 0$, since the corresponding polynomials $\pi_{n+1}^*(t)$ and $\pi_{n+1}^{\mathrm{p}}(t)$ satisfy

$$\pi_{n+1}^*(t) = \pi_n^{\mathrm{c}}(t) = \pi_{n+1}^{\mathrm{p}}(t).$$

This is not, in general, true for the fixed leading coefficient implementation, and it can be shown that

$$c = \frac{1}{\xi_1 \cdots \xi_k}\left\{\frac{1}{\xi_k^*} - \frac{1}{\xi_k}\right\}(y_n - y_{n,0}).$$

Increasing order is then achieved in a manner similar to decreasing order. Finally, we remark that, if the step size is to be changed for the next step, then the Nordsieck history array must be rescaled accordingly.

## 6. NUMERICAL RESULTS

The numerical results presented in this section compare the performance of EPISODE to a modified version of EPISODE based on fixed leading coefficient formulas, which we call MODIFIED EPISODE.[2] We used the fixed leading coefficient formulas based on the predictor eq. (2.7) rather than eq. (2.14), since this required fewer changes to the strategies of EPISODE; we felt it was important to make as few changes as possible to EPISODE so that the performance of the two codes would reflect only the difference in the implementation of the underlying variable step-size formulas.

The EPISODE package incorporates a number of methods; all of our results are computed with MF = 21, which uses backward differentiation formulas and a modified Newton iteration scheme with an analytic Jacobian to solve the corrector equation at each step.

The tables in this section give the number of steps (STEPS), function evaluations (FNS), and Jacobian evaluations (JACS) required by each of the methods EPISODE and MODIFIED EPISODE to solve a particular problem at a tolerance TOL. For these methods, the number of matrix factorizations is equal to the number of Jacobian evaluations. Also given is a measure of the error (ERROR) incurred by each method.

In each example, the values of all the input parameters are specified. IERROR is the error control indicator; IERROR = 1 denotes absolute error control, while IERROR = 3 denotes semirelative error control (see Hindmarsh and Byrne [15, p. 14]).

The results for the first three examples were obtained on an IBM 370 in DOUBLE PRECISION, while the results for Example 4 were obtained on a DEC 20, also in DOUBLE PRECISION.

*Example* 1. We begin by testing the effect of simply changing the corrector formulas and the corresponding error estimates of EPISODE to those of MODIFIED EPISODE. No effort has been made to tune MODIFIED EPISODE: That is, the heuristics used are exactly those of EPISODE. Both methods were run on the 25 problems of STIFF DETEST [9] at the tolerances $10^{-2}$, $10^{-4}$, and $10^{-6}$. The summaries of the results for each tolerance (as well as an overall summary) are presented in Table II. The problems were run with IERROR = 1 and two values for ERROR are given with each entry. The first is the average, over the 25

[2] A more detailed description of the changes to EPISODE may be obtained from the authors.

Table II. A Comparison of Results, MODIFIED EPISODE/EPISODE, for the 25 Problems of STIFF DETEST, with Both Methods Using the Same Heuristics

| TOL | STEPS | FNS | JACS | ERROR |
|---|---|---|---|---|
| $10^{-2}$ [a] | $\frac{3697}{3408} = 1.08$ | $\frac{4216}{4634} = 0.91$ | $\frac{569}{579} = 0.98$ | $\frac{2.76\ (22.58)}{1.40\ (11.54)}$ |
| $10^{-4}$ | $\frac{5072}{5331} = 0.95$ | $\frac{6299}{7468} = 0.84$ | $\frac{868}{846} = 1.02$ | $\frac{4.53\ (23.91)}{4.10\ (30.81)}$ |
| $10^{-6}$ | $\frac{7376}{7393} = 1.00$ | $\frac{9132}{11261} = 0.81$ | $\frac{1185}{1093} = 1.08$ | $\frac{9.14\ (75.75)}{7.65\ (88.68)}$ |
| Summary | $\frac{16145}{16132} = 1.00$ | $\frac{19647}{23363} = 0.84$ | $\frac{2622}{2518} = 1.04$ | $\frac{5.48\ (75.75)}{4.38\ (88.68)}$ |

[a] EPISODE did not solve problems D3 and E5 at TOL = $10^{-2}$, while MODIFIED EPISODE solved D3 but not E5 at this tolerance. Therefore, the statistics for these two problems are not included in Table II.

problems, of the global errors at the endpoints of the integrations. The second (parenthesized) figure is the maximum of the global errors at the endpoints. Both errors are given in units of the tolerance.

Because both methods used the same heuristics, they required approximately the same number of steps to solve the problems of STIFF DETEST. Also, they changed step size at roughly the same points. Hence each required a similar number of Jacobian evaluations and matrix factorizations. In addition, both the average and the maximum of the global errors are comparable for the two methods. However, MODIFIED EPISODE achieved a significant saving in the number of function evaluations required due to its fixed leading coefficient formulas and the resulting faster convergence of the Newton iteration scheme.

In order to gauge the potential of the fixed leading coefficient implementation, we made some changes to the heuristics of MODIFIED EPISODE. Just as the strategies used in GEAR, a fixed coefficient implementation, are not suitable for the variable coefficient method EPISODE, the strategies in a fixed leading coefficient implementation should differ from those used in GEAR and EPISODE.

To take advantage of the fixed leading coefficient formulas in MODIFIED EPISODE, the heuristics were changed to keep the step size constant over larger segments of the integration. This was accomplished by changing the parameter THRESH from 1.3 to 1.5 and ETAMX3 from 1.5 to 10.0. (The step size is increased from $h$ to $h'$ only if the ratio $\eta = h'/h$ satisfies $\eta \geq$ THRESH and (after the first few steps) $\eta \leq$ ETAMX3.) Because we allow larger step-size ratios, the bias factors BIAS1 and BIAS2 were changed from 25 to 36 so that the method would aim at slightly smaller tolerances (see Hindmarsh and Byrne [15, p. 41]). The results with these heuristics for MODIFIED EPISODE appear in Table III.

It can be seen that MODIFIED EPISODE requires significantly fewer function and Jacobian evaluations to achieve roughly the same accuracy as EPISODE. However, MODIFIED EPISODE does use more steps.

The program GEAR performs very well on the problems of STIFF DETEST [9], but it has difficulty on some of the problems in [4, 5]. The following examples are chosen to test the fixed leading coefficient implementation on three of these

Table III. A Comparison of Results, MODIFIED EPISODE/EPISODE, for the 25 Problems of STIFF DETEST, with MODIFIED EPISODE using the Heuristics Described in the Second Part of Example 1

| TOL | STEPS | FNS | JACS | ERROR |
|---|---|---|---|---|
| $10^{-2}$ [a] | $\frac{3726}{3408} = 1.09$ | $\frac{4191}{4634} = 0.90$ | $\frac{486}{579} = 0.84$ | $\frac{2.58}{1.40}$ $\frac{(26.41)}{(11.54)}$ |
| $10^{-4}$ | $\frac{5300}{5331} = 0.99$ | $\frac{6140}{7468} = 0\,82$ | $\frac{699}{846} = 0.83$ | $\frac{2.77}{4.10}$ $\frac{(16.48)}{(30.81)}$ |
| $10^{-6}$ | $\frac{8091}{7393} = 1.09$ | $\frac{9349}{11261} = 0.83$ | $\frac{982}{1093} = 0.90$ | $\frac{8.17}{7.65}$ $\frac{(70.13)}{(88\,68)}$ |
| Summary | $\frac{17117}{16132} = 1.06$ | $\frac{19680}{23363} = 0.84$ | $\frac{2167}{2518} = 0.86$ | $\frac{4.51}{4.38}$ $\frac{(70.13)}{(88.68)}$ |

[a] EPISODE did not solve problems D3 and E5 at TOL = $10^{-2}$, while MODIFIED EPISODE solved D3 but not E5 at this tolerance. Therefore, the statistics for these two problems are not included in Table III.

problems. In each case, the heuristics used in MODIFIED EPISODE are the modified heuristics described above in the second part of Example 1.

*Example* 2. This test problem is the mockup of a diurnal chemical process described in [3] as well as in [4, 5]:

$$\dot{Y}(t) = \dot{H}(t) - B[Y(t) - H(t)],$$

$$Y(0) = H(0),$$

$$H(t) = \frac{D + A \cdot E(t)}{B},$$

$$E(t) = \begin{cases} \exp(-cw/\sin(wt)), & \sin(wt) > 0, \\ 0, & \sin(wt) \le 0, \end{cases}$$

$$A = 10^{-18}, \quad B = 10^{8}, \quad c = 4, \quad D = 10^{-19}, \quad w = \frac{\pi}{43200}.$$

The solution resembles a square wave with a period of 24 hours. This is an extremely difficult problem to solve numerically as the solution rises and falls very sharply. The code GEAR, based on fixed coefficient formulas, fails to solve it altogether.

The problem was solved with $h_0 = 10^{-8}$, IERROR = 3, and $t_{OUT} = 432000$. Output was generated at intervals of 12 hours beginning at 6 hours and running $4\frac{1}{2}$ days and, in addition, at $t_{OUT}$ (5 days). ERROR is the largest observed value (to three decimal places) of

$$\frac{|y_n - Y(t_n)|}{\text{TOL} \cdot Y(t_n)}$$

at the output points. Because MODIFIED EPISODE can increase its step size very quickly, we imposed a maximum step size of HMAX = 12 hours to prevent it from missing a half period. The results for this problem appear in Table IV.

As can be seen, MODIFIED EPISODE performs very well on this problem

Table IV. A Comparison of Results, MODIFIED EPISODE/ EPISODE, for Example 2[a]

| TOL | STEPS | FNS | JACS | ERROR |
|---|---|---|---|---|
| $10^{-2}$ | $\frac{455}{582} = 0.78$ | $\frac{778}{920} = 0.85$ | $\frac{286}{440} = 0.65$ | $\frac{0.000}{0.000}$ |
| $10^{-4}$ | $\frac{964}{1063} = 0.91$ | $\frac{1541}{1927} = 0.80$ | $\frac{398}{526} = 0.76$ | $\frac{0.040}{0\,017}$ |
| $10^{-6}$ | $\frac{1940}{2216} = 0.88$ | $\frac{2983}{4205} = 0.71$ | $\frac{563}{690} = 0.82$ | $\frac{0.095}{0.021}$ |

[a] HMAX = 12 hours.

and shows no signs of numerical instability. (Even better results were obtained with HMAX = 6 hours.)

*Example* 3. The Van der Pol oscillator is the basis for this problem [5]:

$$\dot{Y}_1(t) = Y_2(t),$$

$$\dot{Y}_2(t) = 100 \cdot [1 - Y_1^2(t)] \cdot Y_2(t) - Y_1(t),$$

$$Y_1(0) = 2, \qquad Y_2(0) = 0.$$

The object of the problem is to find the zeros of $Y_1(t)$. This is difficult since $Y_1(t)$ changes very rapidly in a small neighborhood of each zero. The tactic used by Byrne et al. [5] for finding the roots was to let each method choose its own step size for an initial time interval and, thereafter, to require the integrator to return to the main program after each step. A check was then made to see whether $Y_1(t)$ had changed sign. If it had, then Newton's method,

$$t_z = t_{\text{OUT}} - \frac{Y_1(t_{\text{OUT}})}{Y_2(t_{\text{OUT}})},$$

was used to improve the output value, $t_{\text{OUT}}$, and the improved value, $t_z$, was taken as an approximation to the root, $t^*$, of $Y_1(t)$ in the neighborhood of $t^*$. The process was repeated until four roots had been found.

The problem was solved with $h_0 = 10^{-8}$ and IERROR = 3. The results appear in Table V. ERROR measures the maximum value of

$$\frac{t_z - t^*}{\text{TOL} \cdot t^*}$$

at each tolerance.

Again, MODIFIED EPISODE is able to solve the problem more efficiently than EPISODE. On the other hand, the report of Byrne et al. [5] shows that GEAR performed much worse than EPISODE on this problem with respect to both speed and accuracy.

*Example* 4. We again borrow a test problem from Byrne et al. [4, 5]. This example is based on Burgers' equation

$$U_t + UU_x = aU_{xx}, \qquad 0 \le x \le 1, \qquad t \ge 0, \tag{6.1}$$

Table V. A Comparison of Results, MODIFIED EPISODE/EPISODE, for Example 3

| TOL | STEPS | FNS | JACS | ERROR |
|---|---|---|---|---|
| $10^{-2}$ | $\frac{255}{572} = 0.38$ | $\frac{452}{769} = 0.59$ | $\frac{135}{145} = 0.93$ | $\frac{5.13}{1.28}$ |
| $10^{-4}$ | $\frac{441}{645} = 0.68$ | $\frac{761}{1090} = 0.69$ | $\frac{123}{188} = 0.65$ | $\frac{27.33}{122.80}$ |
| $10^{-6}$ | $\frac{1083}{957} = 1.13$ | $\frac{1530}{1689} = 0.91$ | $\frac{176}{205} = 0.86$ | $\frac{99.89}{66.72}$ |

which has

$$U(x, t) = [1 + \exp(x/(2a) - t/(4a))]^{-1} \tag{6.2}$$

as a particular solution. One way of solving such time-dependent problems is to apply the "method of lines" to the PDE to derive a system of ODEs whose solution is an approximation to the solution of the PDE.

In particular, if the spacial derivatives in eq. (6.1) are replaced by finite difference approximations, the following system of ODEs is obtained:

$$\dot{U}_i = \frac{-U_i(U_{i+1} - U_{i-1})}{2H} + \frac{a(U_{i+1} - 2U_i + U_{i-1})}{H^2} \tag{6.3}$$

for $i = 1, 2, \ldots, N$, where

$$H = \frac{1}{N+1},$$

$$U_0(t) = U(0, t),$$

and

$$U_{N+1}(t) = U(1, t).$$

Of course, $U_i(t)$ is meant to be an approximation to $U(iH, t)$.

Because of the discretization error in the spacial derivatives, even if

$$U_i(0) = U(iH, 0), \qquad i = 1, 2, \ldots, N,$$

$U_i(t)$ is not equal to $U(iH, t)$ for $t > 0$. However, if we let $f_i(\mathbf{U})$ be the right-hand side of eq. (6.3) and let

$$g_i(t) = [1 + \exp(iH/(2a) - t/(4a))]^{-1},$$

then the vector-valued initial value problem

$$\dot{\mathbf{Y}}(t) = \mathbf{f}(\mathbf{Y}) + \{\mathbf{g}(t) - \mathbf{f}(\mathbf{g}(t))\}, \qquad \mathbf{Y}(0) = \mathbf{g}(0), \tag{6.4}$$

obviously has the solution $\mathbf{Y}(t) = \mathbf{g}(t)$, provided that we use $Y_0(t) = g_0(t)$ and $Y_{N+1}(t) = g_{N+1}(t)$ as the boundary conditions in the equations defining $f_1(\mathbf{Y})$ and $f_N(\mathbf{Y})$, respectively.

As Byrne et al. [4, 5] suggest, we actually solve eq. (6.4) rather than eq. (6.3) in this example, since we have the advantage of knowing the solution of eq. (6.4).

Table VI. A Comparison of Results, MODIFIED EPISODE/ EPISODE, for Example 4 (Burgers' Equation: $N = 20$, $a = 0.05$)

| TOL | STEPS | FNS | JACS | ERROR |
|---|---|---|---|---|
| $10^{-2}$ | $\frac{48}{40} = 1.20$ | $\frac{66}{63} = 1.05$ | $\frac{6}{9} = 0\,67$ | $\frac{1.18}{1.00}$ |
| $10^{-4}$ | $\frac{89}{71} = 1\,25$ | $\frac{100}{111} = 0.90$ | $\frac{14}{10} = 1\,40$ | $\frac{3.79}{3.18}$ |
| $10^{-6}$ | $\frac{203}{145} = 1.40$ | $\frac{219}{219} = 1.00$ | $\frac{22}{21} = 1.05$ | $\frac{31.54}{9.78}$ |

The integration was performed on the interval [0, 4] with $a = 0.05$, $N = 20$, $h_0 = 0.1 \cdot \mathrm{TOL}$, and IERROR = 3. The results appear in Table VI, where ERROR is

$$\max_{t_j} \left\{ \sum_{i=1}^{N} \frac{[(y_{i,j} - Y_i(t_j))/\mathrm{YMAX}_{i,j}]^2}{N} \right\}^{1/2} \Big/ \mathrm{TOL}$$

at the output points $t_j = j/2, j = 1, 2, \ldots, 8$. In the above formula, $y_{i,j}$ is the value of the $i$th component of the computed solution at $t_j$ and $\mathrm{YMAX}_{i,j}$ is the maximum observed value of the $i$th component of the computed solution on the interval [0, $t_j$]. (YMAX is computed by both EPISODE and MODIFIED EPISODE; see Hindmarsh and Byrne [15, pp. 22, 23].)

This is a mildly stiff problem whose solution is a traveling wave (see [5, Figure 2.5.1]). Because the problem is only mildly stiff, it would seem that a "good" approximation to the Jacobian is not really required for the convergence of the Newton iteration. In addition, because the traveling wave gradually damps out, frequent increases in the step size are required to solve this problem efficiently. As a result, EPISODE solves this problem more efficiently than does GEAR, while the results in Table VI show that the performance of EPISODE and MODIFIED EPISODE is comparable in terms of function and Jacobian evaluations, although MODIFIED EPISODE consistently takes more steps.

To conclude, our results indicate that a significant saving in both function and Jacobian evaluations is possible if a fixed leading coefficient rather than a variable coefficient implementation is used in methods designed to solve stiff systems. This saving is due to the fact that the iteration matrix used in the modified Newton scheme is a more accurate approximation to the actual Newton iteration matrix over many steps of the integration. Moreover, it appears that the method does not suffer from the stability problems of a fixed coefficient implementation such as GEAR.

## REFERENCES

1. Brayton, R K., and Conley, C.C. Some results on the stability and instability of the backward differentiation methods with non-uniform time steps. *Proc. Int. Symp Numerical Analysis*, Dublin, 1972, pp. 13–33
2. 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.

3. BYRNE, G.D., AND HINDMARSH, A.C. A polyalgorithm for the numerical solution of ordinary differential equations. *ACM Trans. Math. Software 1*, 1 (March 1975), 71–96.
4. BYRNE, G.D., HINDMARSH, A.C., JACKSON, K.R , AND BROWN, H.G. A comparison of two ODE codes· GEAR and EPISODE *Comput. Chem. Eng. 1* (1977), 133–147.
5. BYRNE, G.D., HINDMARSH, A.C., JACKSON, K.R., AND BROWN, H.G. Comparative test results for two ODE solvers—EPISODE and GEAR. Tech. Rep. ANL-77-19, Argonne National Lab., Argonne, Ill., 1977.
6. ENRIGHT, W.H. Second derivative multistep methods for stiff ordinary differential equations. *SIAM J Numer. Anal. 11* (1974), 321–331.
7. ENRIGHT, W.H , AND HULL, T.E. Comparing numerical methods for the solution of stiff systems of ODE's arising in chemistry. In *Numerical Methods for Differential Systems*, L. Lapidus and W.E. Schieser, Eds Academic Press, New York, 1976, pp. 45–66.
8. ENRIGHT, W.H , AND HULL, T.E. Test results on initial value methods for non-stiff ordinary differential equations. *SIAM J. Numer. Anal 13* (1976), 944–961
9. ENRIGHT, W.H., HULL, T.E., AND LINDBERG, B. Comparing numerical methods for stiff systems of ODE's. *BIT 15* (1975), 10–48
10. GEAR, C.W. *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice-Hall, Englewood Cliffs, N.J , 1971.
11. GEAR, C.W Asymptotic estimation of errors and derivatives for the numerical solution of ordinary differential equations. Information Processing 74, North-Holland, Amsterdam, 1974, pp. 447–451.
12. GEAR, C.W., AND TU, K.W. The effect of variable mesh size on the stability of multistep methods. *SIAM J Numer. Anal. 11* (1974), 1025–1043.
13. GEAR, C.W., TU, K.W , AND WATANABE, D.S. The stability of automatic programs for numerical problems. In *Stiff Differential Systems*, R.A. Willoughby, Ed. Plenum Press, New York, 1974, pp. 111–121.
14. HINDMARSH, A.C. GEAR: Ordinary differential equation system solver. Tech. Rep. UCID-30001, Rev 3, Lawrence Livermore Lab., Univ. California, Livermore, Calif., 1974.
15. HINDMARSH, A.C., AND BYRNE, G.D. EPISODE: An effective package for the integration of systems of ordinary differential equations. Tech. Rep. UCID-30112, Rev. 1, Lawrence Livermore Lab., Univ. California, Livermore, Calif., 1977
16. JACKSON, K.R. Variable stepsize, variable order integrand approximation methods for the numerical solution of ordinary differential equations Ph.D. Dissertation, Tech. Rep. 129, Dep. Computer Science, Univ. Toronto, Toronto, Canada, 1978.
17. KROGH, F.T. VODQ/SVDQ/DVDQ—Variable order integrators for the numerical solution of ordinary differential equations. TU Doc. CP-2308, NPO-11643, Jet Propulsion Lab , California Inst. Technol., Pasadena, Calif., 1969.
18 SACKS-DAVIS, R. Software for the solution of stiff ODE's based upon second derivative formulas. Tech. Rep. 145, Dep Computer Science, Technion, Haifa, Israel, 1979.
19 SEDGWICK, A.E. An effective variable order, variable stepsize Adams method Ph.D. Dissertation, Tech Rep. 53, Dep. Computer Science, Univ. Toronto, Toronto, Canada, 1973.
20. SHAMPINE, L.F., AND GORDON, M.K. *Computer Solution of Ordinary Differential Equations*. W.H. Freeman, San Francisco, Calif., 1975
21. SHAMPINE, L.F., AND GORDON, M.K. Local error and variable order Adams codes. *Appl. Math. Comput. 1* (1975), 47–66.
22. SHAMPINE, L.F., WATTS, H.A , AND DAVENPORT, S.M Solving non-stiff ordinary differential equations—The state of the art. *SIAM Rev. 18* (1975), 376–411.
23. TU, K.W. Stability and convergence of general multistep and multivalue methods with variable step size. Ph D. Dissertation, Tech. Rep UIUCDCS-R-72-526, Dep. Computer Science, Univ. Illinois at Urbana-Champaign, Urbana, Ill., 1972.

Received July 1978; revised August 1979; accepted October 1979