MATHEMATICS OF COMPUTATION, VOLUME 33, NUMBER 145
JANUARY 1979, PAGES 125–138

A New Step-Size Changing Technique for Multistep Methods

By G. K. Gupta and C. S. Wallace

Abstract. The step-size changing technique is an important component of a Variable Step Variable Order algorithm for solving ordinary differential equations using multistep methods. This paper presents a new technique for changing the step-size and compares its performance to that of the Variable-Step and Fixed-Step Interpolation techniques.

1. Introduction. It is generally assumed that in solving ordinary differential equations numerically, the best linear multistep methods (formulae) available are those based on interpolation; that is, the Adams-Moulton formulae (AMF) for solving the nonstiff equations and the backward differentiation formulae (BDF) for solving the stiff equations. Recent work by the authors reported in Wallace and Gupta (1973), Gupta and Wallace (1975) and Gupta (1975, 1976) has shown that the multistep formulae based on interpolation are not necessarily the best. This is particularly true for formulae for solving stiff equations and several high order formulae of orders up to 12 suitable for solving stiff equations have been presented in the papers referred to above. Existence of multistep methods of arbitrarily high orders for solving stiff equations has been proved by Jeltsch (1976).

Several excellent algorithms for solving ordinary differential equations using multistep methods BDF and AMF have been published recently, e.g. Krogh (1969), Brayton et al. (1972), Gear (1971b), Shampine and Gordon (1975) and Byrne and Hindmarsh (1975). All these algorithms, except that of Gear, use the Variable-Step (VS) technique for varying the step-size. In the VS technique, the multistep formulae used are based on unequally spaced points (x_{n-i}, y_{n-i}), $i = 0, 1, \ldots, k$. This technique provides almost complete freedom in selection of step-sizes because the VS Adams-Moulton formulae are always stable and convergent. This was proved by Piotrowski (1969). Also Brayton et al. (1972) found that BDF could sometimes become unstable for the VS technique but the formulae based on the VS technique were more stable than those based on the Fixed-Step Interpolation (FSI) technique used by Gear (1971b).

The FSI technique used by Gear (1971b) in his subroutine DIFSUB essentially consists of the following two steps:

Received June 20, 1977.

AMS (MOS) subject classifications (1970). Primary 65L05; Secondary 65D30.

Key words and phrases. Multistep methods, stiff equations, ordinary differential equations, initial value problems.

(a) Suppose the solution at unequally spaced points x_{n-i}, $i = 0, 1, 2, \ldots, k$, has been computed and it is required to compute the solution at $x_{n+1} = x_n + h_n$. Then interpolate using the current approximating polynomial to obtain the values of the solution ($\tilde{y}_{n-i}$) at $\tilde{x}_{n-i} = x_n - ih_n$, $i = 1, 2, \ldots, k$. Note that, in general, the values $\tilde{y}_{n-i}$ are not computed by interpolating through the last $k + 1$ computed values of the solution.

(b) Use the fixed step-size multistep formula to compute y_{n+1}.

The FSI technique is easily implemented using the representation suggested by Nordsieck (1962), but the technique is not suitable when high order formulae are used. Krogh (1973) has shown that for AMF, the FSI technique can cause serious trouble if the step-size is reduced for methods of orders greater than 7. It is expected that these results are applicable to all multistep formulae.

Though the FSI technique can become unstable, it is a technique which is flexible in the sense that any set of formulae can be used in subroutine DIFSUB simply by replacing the coefficients of the formulae being used presently (and the truncation error coefficients). This is in contrast to an algorithm using the VS technique. To use a new set of formulae in an algorithm using the VS technique must necessarily involve replacing all the code which computes the VS coefficients. Since there is no consensus on what are the best formulae for solving stiff equations, flexibility to use several formulae in an algorithm may be desirable.

In this paper we present a technique called the Average-Step Interpolation technique, which we believe does provide the flexibility to use any set of multistep formulae in an algorithm and is shown to be stable by numerical testing.

We have assumed that the equation being solved is $y' = f(x, y)$, $y(x_0) = y_0$. We note that the predictor-corrector algorithm may be represented by the following polynomial representation.

$$P_{n+1}(x) = P_n(x) + \delta_{n+1} C((x - x_{n+1})/h_{n+1}) \tag{1.1}$$

$P_{n+1}(x)$ and $P_n(x)$ are the polynomials approximating y at x_{n+1} and x_n respectively. C is the modifier polynomial representing the formulae used, h_{n+1} is the present step-size and δ_{n+1} is the correction applied so that $P_{n+1}(x)$ satisfies the differential equation at x_{n+1}. (Note that in this definition of δ_{n+1} there is no need to normalize C.) The representation (1.1) is similar to that presented in Wallace and Gupta (1973). Also (1.1) is equivalent to the following representation used by Gear (1971a, p. 224)

$$a_{n+1} = Aa_n + lw. \tag{1.2}$$

In Eq. (1.2), a_{n+1} and a_n are vectors of the scaled derivatives of the polynomials $P_{n+1}(x)$ (at x_{n+1}) and $P_n(x)$ (at x_n) respectively. l is a vector of the scaled derivatives of the modifier polynomial C at x_{n+1}, A is the Pascal triangle and $w = \delta_{n+1} C(0)/l_0$; l_0 being the first element of l. This representation of a polynomial by a vector of its scaled derivatives was originally suggested by Nordsieck (1962) and has been recently used by Byrne and Hindmarsh (1975). In this paper we are going to assume that this is the representation being used because it seems to be a very convenient way to represent all multistep methods.

2. Average-Step Interpolation (ASI) Technique. Let us first consider an example which demonstrates why the FSI technique is unsatisfactory. The modifier polynomial corresponding to the BDF of order $k + 1$ is

$$C(t) = (t - t_n)(t - t_{n-1}) \cdots (t - t_{n-k}), \tag{2.1}$$

where

$$t_{n-i} = (x_{n-i} - x_{n+1})/(x_{n+1} - x_n), \tag{2.2}$$

$$t = (x - x_{n+1})/(x_{n+1} - x_n). \tag{2.3}$$

Now consider that the solution at the following points has been computed

$$x_{n-3} = 0.6, \quad x_{n-2} = 1.0, \quad x_{n-1} = 2.0, \quad x_n = 3.0.$$

It is now required to compute solution at $x_{n+1} = 3.5$. We therefore have $t_{n-3} = -5.8$, $t_{n-2} = -5.0$, $t_{n-1} = -3.0$, $t_n = -1.0$, $t_{n+1} = 0.0$. If BDF of order 4 is being used, the modifier polynomial using VS technique is given by

$$C(t) = (t + 1)(t + 3)(t + 5)(t + 5.8).$$

Using the FSI technique, the following modifier polynomial is obtained

$$C(t) = (t + 1)(t + 2)(t + 3)(t + 4).$$

The vectors of the scaled derivatives of $C(t)$, that is the vectors l in the representation (1.2), are (to three digits) $[1.0, 1.706, 0.864, 0.170, 0.0115]^T$ for VS and $[1.0, 2.083, 1.458, 0.417, 0.0417]^T$ for FSI.

Since the correction term w in Eq. (1.2) is computed using only the first two elements of l, it can easily be shown that the difference between w using FSI and w using VS will be of the order of the difference between the second elements of the vectors l. Therefore using the FSI technique, the correction applied to the 3rd and 4th derivatives of the approximating polynomial will be much greater than that applied using the VS technique. In our opinion this is the main cause of the trouble using FSI. We therefore want that the interpolation points should be well spread over the interval containing the last k points (i.e. $x_n - x_{n-k}$) rather than crowding together towards x_n whenever the step-size is being reduced as in FSI. The best solution, of course, is to compute the VS formulae but a good compromise seems to be to choose the interpolation points equally spaced over the interval $x_n - x_{n-k}$, the spacing being the average step-size in the last k-steps. That is why the name ASI.

The last element of l in the above vectors was $1/C(0)$, $C(0)$ being the product of the roots of the modifier polynomial. For BDF these roots are the interpolation points. When the step-size is being reduced, the sum of the roots of the modifier polynomial is obviously going to be much greater for the VS formulae than for the FSI formulae.

In the above example, if we used the ASI technique, then the modifier polynomial will be

$$C(t) = (t + 1)(t + 2.6)(t + 4.2)(t + 5.8).$$

Therefore the vector l using ASI is $[10, 1.795, 0.994, 0.215, 0.0158]^T$.

Graphically the idea behind FSI and ASI is illustrated in Figure 1 and Figure 2 respectively. In these figures P_n is the polynomial approximating y at x_n, C is the modifier polynomial (for BDF) used in computing y_{n+1} and the vertical lines show the points used in computing C.

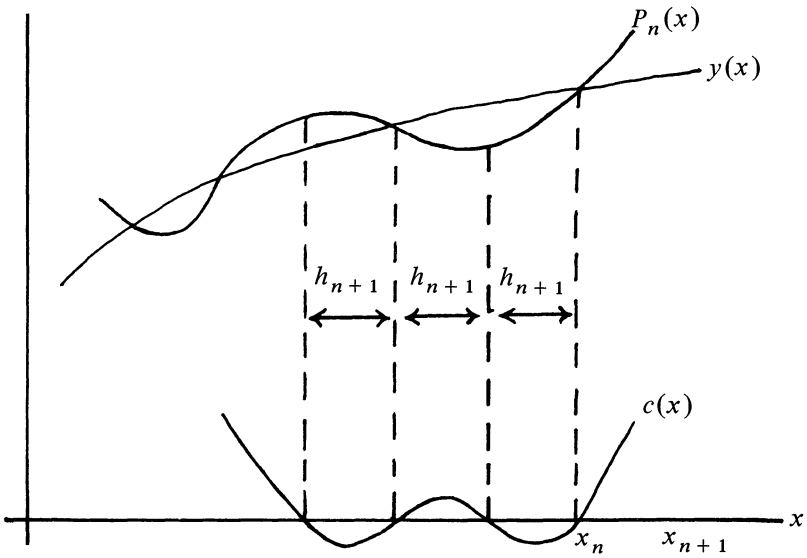

FIGURE 1. *Fixed-Step Interpolation*

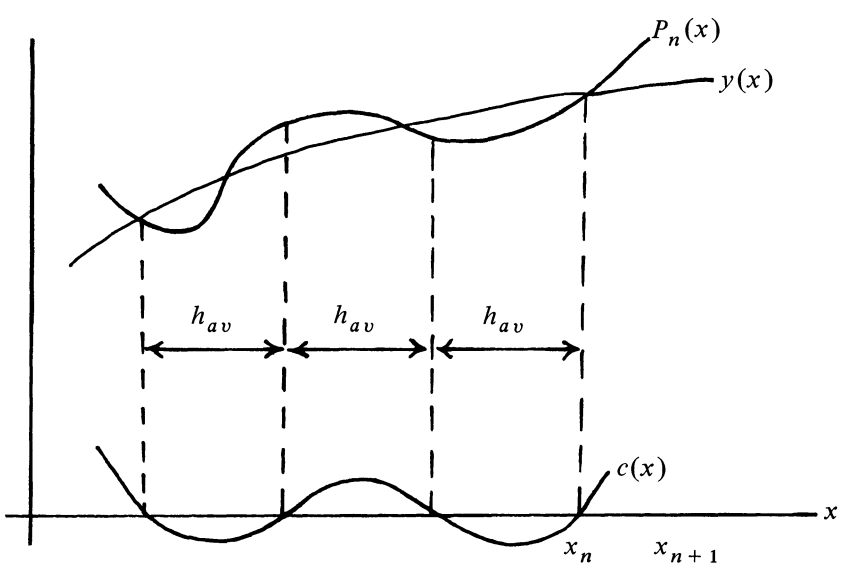

FIGURE 2. *Average-Step Interpolation*

3. An Algorithm for ASI. Let the order of the method being used be $k + 1$ and let the last $k + 1$ points at which the solution has been computed be x_{n-i}, $i = 0, 1, \ldots, k$. The average step-size in last $k + 1$ steps is $h_{av} = (x_n - x_{n-k})/k$. Since we are going to compute $C(t)$ using the scaled variable t defined by (2.3), we define h_t as the average step-size in the scaled variable t. Therefore

$$h_t = h_{av}/(x_{n+1} - x_n).$$

Now the collocation points are (again, using the scaled variable t)

$$-1 - ih_t, \qquad i = 0, 1, \ldots, k.$$

Since these points are equally spaced, we can represent the modifier polynomial based on these points as a vector of scaled derivatives at $t = -1$. This representation is independent of h_t and therefore may be stored in the subroutine.

Now let $C(t)$ be given by

$$C(t) = c_0 + c_1 t + c_2 t^2 + \cdots + c_{k+1} t^{k+1}. \tag{3.1}$$

The vector of the coefficients $\{c_i\}$ is equivalent to the vector l (except a scalar multiplication). Also this vector of coefficients may be thought of as the vector of the scaled derivatives of the polynomial $C(t)$ at $t = 0$. (Note that the present step-size in variable t is always 1.) Therefore to compute l we need to compute the vector of the scaled derivatives of $C(t)$ at $t = 0$. We already have a vector of scaled derivatives (scaled so that the step-size is h_t) of $C(t)$ at $t = -1$. Therefore two steps are involved in computing l:

(a) Rescale the vector of the scaled derivatives of $C(t)$ at $t = -1$ by multiplying the ith element of the vector h_t^{k+1-i} $(i = 0, 1, \ldots, k)$.

(b) Extrapolate (by multiplying by the Pascal triangle) to obtain l.

Consider an example to illustrate the above algorithm. For BDF of order 4, the modifier polynomial using ASI is

$$C(t) = (t + 1)(t + 1 + h_t)(t + 1 + 2h_t)(t + 1 + 3h_t).$$

This may be represented as the following vector of the scaled derivatives of $C(t)$

$$[0, 6h_t^3, 11h_t^2, 6h_t, 1]^T.$$

The vector is now multiplied by the Pascal triangle giving us the vector l.

4. Stability of FSI and ASI Techniques. In this section we rigorously analyze a simple example where FSI gets into trouble while ASI does not. The stability of the FSI technique has been analyzed by Gear and Tu (1974), who have shown that the FSI technique is stable if at least k steps of constant step-size are taken between step changes for k-step Adams method. Also it was shown that if the step-size varies so that

$$h_{2i} = \alpha h_{2i+1} = h_{2i+2} = \alpha h_{2i+3} = \cdots \tag{4.1}$$

then the Adams method becomes unstable. We now analyze the stability of the ASI technique if the step-size varies according to (4.1).

We follow an approach similar to that of Gear and Tu (1974) and define a predictor-corrector method, when the step-size may not be constant, as

$$a_{n+1} = A_n a_n + w_n l_n \tag{4.2}$$

a_{n+1}, a_n are vectors of the scaled derivatives of the approximating polynomials at x_{n+1} and x_n respectively, l_n is a vector of the scaled derivatives of the modifier polynomial at x_{n+1}, w_n is the correction required (a scalar) so that a_{n+1} satisfied the differential equation at x_{n+1}, and $A_n a_n$ is the predicted vector of derivatives. To simplify the analysis, we consider that the differential equation being solved in $y' = f(x)$ and that a third order Adams method is being used to solve it. Therefore at x_{n+1}, the second element of vector a_{n+1} must be equal to $h_n f(x_{n+1})$, where $h_n = x_{n+1} - x_n$. (Further assume that l_n in (4.2) always has its second element equal to 1.) Since a_{n+1} must satisfy the differential equation at x_{n+1}, the second element of $A_n a_n + w_n l_n$ must be equal to $h_n f(x_{n+1})$. This implies that

$$w_n = h_n f(x_{n+1}) - e_2^T A_n a_n$$

and therefore,

$$a_{n+1} = A_n a_n + h_n f(x_{n+1}) l_n - e_2^T A_n a_n l_n \tag{4.3}$$

where

$$e_2^T = [0, 1, 0, 0, \ldots, 0].$$

To investigate the stability of the above predictor-corrector scheme, we need to find an expression for the global error propagation. The global error vector at x_{n+1} is ϵ_{n+1} which is defined as

$$\epsilon_{n+1} = a_{n+1} - b_{n+1} \tag{4.4}$$

where b_{n+1} is a vector of scaled true derivatives of the solution. Also let us define

$$\begin{aligned} \tilde{a}_{n+1} &= A_n b_n + w l_n \\ &= A_n b_n + h_n f(x_{n+1}) l_n - e_2^T A_n b_n l_n. \end{aligned} \tag{4.5}$$

$\tilde{a}_{n+1} - b_{n+1}$ is the truncation error, d_n, and therefore

$$\begin{aligned} \epsilon_{n+1} &= a_{n+1} - \tilde{a}_{n+1} + \tilde{a}_{n+1} - b_{n+1} \\ &= a_{n+1} - \tilde{a}_{n+1} + d_n \\ &= (I - l_n e_2^T) A_n \epsilon_n + d_n \quad \text{using (4.3) and (4.5).} \end{aligned}$$

Therefore

$$\epsilon_{n+1} = S_n \epsilon_n + d_n \tag{4.6}$$

where

$$S_n = (I - l_n e_2^T)A_n.$$

For the FSI technique $l_n = l$ and it has been shown by Tu (1972) that one of the eigenvalues of $\Pi_n S_n$ becomes large if step-size changes as in (4.1). For the ASI technique, $l_n = ADA^{-1}l$ where A is the pascal triangle, D is a scaling matrix, and l is the vector l_n as used in FSI. For a third order Adams method, $A^{-1}l = [0, 0, 1/4, 1/6]^T$ and D, a diagonal matrix, has its diagonal as $(1, r, r^2, r^3)$ where $r = 2\alpha/(\alpha + 1)$ when step-size is reduced and $r = 2/(\alpha + 1)$ when it is increased using the scheme (4.1). Using the above vectors l_n, we can find expressions for S_i when the step-size goes down by a factor of α and also S_{i+1} when step-size goes up by a factor of α. It can be shown that the product of $S_i S_{i+1}$ simplifies to

$$\begin{bmatrix} 1 & ? & ? & ? \\ 0 & 0 & 0 & 0 \\ 0 & ? & (2u-1)(2v-1) + 6u(2v-1)/3\alpha & (2u-1)(3v\alpha) + 6uv \\ 0 & ? & 2(2u-1)(2v-1)/3 + 4u(2v-1)/3\alpha & 2(2u-1)v\alpha + 4uv \end{bmatrix}$$

where $u = (\alpha + 1)/(6 + 2\alpha)$ and $v = (\alpha + 1)/(6\alpha + 2)$. The above matrix has eigenvalues of $1, 0, 0, -(\alpha - 1)^2/\{4\alpha + 3(\alpha + 1)^2\}$. If $\alpha > 1$ then the last eigenvalue is certainly less than unity and therefore the ASI technique is stable.

5. Numerical Testing. In this section the performance of the new technique ASI is compared with that of VS and FSI techniques. Numerical testing similar to that suggested by Krogh (1973) was done and detailed results are presented in Gupta (1978). Some of the more interesting results are presented in this paper.

To test the techniques we used AMF of orders 5 to 14 to solve the differential equations $y' = f(x)$; $f(x) = e^{-x}$, $\cos x$, $\cos x e^{\sin x}$, $4e^{2x}/(e^{2x} + 1)^2$. We selected quadrature type differential equations and AMF only for simplicity and, as commented by Krogh (1973), we expect other methods and other differential equations to produce similar results. The following algorithm was used in testing:

(a) Compute the coefficients of a polynomial, $P(x)$ of degree m, approximating the solution of the differential equation being used in the test. $P(x)$ satisfies the following conditions

$$P'(x_0 - ih) = f(x_0 - ih), \qquad i = 0, 1, \ldots, m - 1,$$

and

$$P(x_0) = y(x_0)$$

where h is the initial step-size and $x_0 = 0.0$.

(b) Now to test the performance of the three techniques when the step-size is changed from h to αh, we compute the vector of the scaled derivatives of $P(x)$ such that the ith element of the vector is

$$(\alpha h)^i P^{(i)}(x_0)/i!, \qquad i = 0, 1, \ldots, m.$$

$P^{(i)}(x_0)$ being the ith derivative of $P(x)$ at $x = x_0$.

(c) Integrate the differential equation for m steps using AMF for each of the three techniques of changing the step-size.

At each step during the numerical testing, the local error in the numerical solution is computed. That is, we compute $(y_{n+1} - y(x_{n+1}))/y(x_{n+1})$ where y_{n+1} and $y(x_{n+1})$ are the computed and true solutions respectively of the following differential equation:

$$y' = f(x, y), \quad y(x_n) = y_n.$$

The results may now be summarized as follows

(1) *Test Series* 1. The aim of these tests was to test the step-size changing technique when the order of the method was held constant.

Krogh (1973) presented results of numerical testing and concluded that FSI works quite satisfactorily when the step-size is increased. Our testing confirms this conclusion. Therefore in this paper our attention is focused on the performance of the three techniques whenever the step-size is reduced.

Methods of orders 5 to 14 were used in the testing. For methods of orders 5 to 7, we tested the three step-size changing techniques when the step-size was reduced by a factor of 2 and by a factor of 4. For higher order methods, the step-size was reduced by a factor of 2. Similar results were obtained for methods of different orders except that usually the performance of FSI got worse at higher orders. In this paper we only present results for methods of orders 8 to 14.

In Tables 1 and 2, we present the local errors computed at each step using the three step-size changing techniques for differential equations $y' = e^x$ and $y' = \cos x$. In Tables 3 and 4, similar results are presented for differential equations $y' = \cos x e^{\sin x}$ and $y' = 4e^{2x}/(e^{2x} + 1)^2$. In Table 5, the 11th derivatives of the approximating polynomials are presented after a step-size change from 0.75 to 0.375. The differential equation being solved was $y' = e^x$ and AMF of order 11 was being used. Results in Table 5 show what exactly happens to the approximating polynomial after the step-size is reduced using the various step-size changing techniques.

(2) *Test Series* 2. To test the step-size changing techniques when the order is changed by one.

In a variable order variable step-size algorithm a situation may arise when the order is lowered and the step-size is reduced. Also it is possible that in some instances when the order is being lowered the step-size may be increasing, the latter usually happening when the derivatives of the approximating polynomials are somewhat corrupted. On the other hand when the order of the method being used is increased, the step-size must be increasing.

We therefore tested the three step-size techniques when the step-size was reduced and the order was lowered by one and also when the step-size was being increased and the order was either lowered by one or increased by one. The results obtained are similar to those obtained when the order was held constant and are therefore of little interest.

Studying the various results of numerical testing when the order of the method and the step-size both were changed, we did find some other minor items of interest. Firstly we found that when the step-size was reduced substantially it was usually better (that is, the local errors were lower) to reduce the order of the method as well. Secondly, the order reducing technique used by Gear (1971b) whereby the mth derivative of the approximating polynomial is dropped, when the order is reduced to $m - 1$, was found to work quite satisfactorily. Also in increasing the order from m to $m + 1$ it was found to be better to just put the $(m + 1)$st derivative of the approximating polynomial to zero rather than using the technique of Gear (1971b). This was suggested by Shampine and Gordon (1975).

(3) *Test Series* 3. To test the stability of the ASI technique.

A large number of tests were conducted to investigate the stability of the ASI technique. In many tests the FSI technique was found to be unstable, but in all the tests, using various sequences of step-sizes, the ASI and VS techniques were always found to be stable. For example, for the method of order 7, the ODE $y' = e^x$ was solved with alternating step-sizes of 0.5 and 0.1. After ten steps, the 7th derivative of the approximating polynomial was $1.1\ E + 11$ when the FSI technique was used. Using the ASI technique, the derivative was only 6.96. In fact the FSI technique was unstable even if the step-sizes of 0.5 and 0.2 were used alternately.

In Table 6, we present results for the differential equations $y' = e^x$ and $y' = \cos x$ when the step-sizes of 0.75 and 0.375 were used alternately using a method of order 11. Similar results were obtained for other differential equations and methods of different orders.

In the above stability tests, the step-sizes used were alternating between two very different values. This alternating behavior cannot be expected to occur in practice and this artificial example was used only to highlight the stable behavior of the ASI technique. Some other sequences of step-sizes have been used and again ASI technique was found to be stable.

6. Discussion. The new step-size changing technique (ASI) seems to be performed quite well in the numerical testing; though not as well as the VS technique. ASI is quite stable and in our opinion is attractive because it provides the flexibility to use several sets of multistep formulae in an algorithm in contrast to the VS technique which requires that separate algorithms must be included for computing the VS coefficients for each set of formulae.

Though generally ASI has performed better than FSI, there are instances where FSI seems to be better. For example, in Table 2 we can see that after step number 10 FSI performs much better than ASI. Similar results were obtained for other differ-

ential equations (Table 4) when method of order 14 was used and the step-size was reduced by a factor of 2. Also for the differential equation $y' = 4e^{2x}/(4e^{2x} + 1)^2$, the performance of ASI is not better than that of FSI (Table 3).

The mild unstability displayed by the ASI technique in some of the tests is most likely due to the following reason. If a step-size change takes place in going from x_n to x_{n+1} and the ASI technique is used for computing the solution at $x_{n+1}, x_{n+2}, \ldots, x_{n+m}$ then the approximating polynomial $P_{n+i}(x)$ at x_{n+i} $(m \geqslant i > 1)$ is computed based on interpolation points which are, except for the last point, all different from the points at which the solution was computed. This is, it seems, not a problem for most cases but in severe cases, like reducing the step-size by a factor of 2 at order 14, it can cause mild unstability. Further testing is being carried out and results will be available in Gupta (1978).

The ASI technique can be implemented efficiently. Step (a) of the algorithm presented in Section 3 takes $2m - 1$ multiplications for a method of order m. Step (b) requires $(m + 1)(m + 2)/2$ additions for extrapolation. Work done in step (b) can be reduced because in the Nordsieck representation the prediction step also involves extrapolation and therefore Step (b) of ASI technique and the prediction step may be combined as follows:

(a) We have the approximating polynomial at x_n represented in Eq. (1.2) by the vector a_n. Rather than computing the predicted vector Aa_n, compute only the first two elements (that is, predicted values of y_{n+1} and hy'_{n+1}). This takes $2m - 1$ additions and $m - 1$ multiplications.

(b) Rather than computing the predicted vector in step (b) of ASI technique compute only the first two elements. Again this takes $2m - 1$ additions and $m - 1$ multiplications.

(c) Now compute the correction to be applied (that is, w). We now have the first two elements of vector a_{n+1} in Eq. (1.2).

(d) Except for the first two elements, add w times the vector obtained after step (a) in Section 3, to a_n.

(e) Extrapolate the corrected vector (except the first two elements). We now have a_{n+1}.

The work done in steps (c), (d),and (e) above is part of the predictor-corrector algorithm and is not additional due to ASI. Therefore the ASI algorithm cost is a linear function of the order m.

A further saving can be made by using the ASI technique for only (say) $m/2$ steps after a step-size reduction (rather than m steps) and then switching to the FSI technique. Also when the step-size is increasing we need not use ASI since FSI works quite satisfactorily.

In Gupta (1975), we have presented an algorithm using the ASI technique as suggested here. The results of preliminary testing presented in Gupta (1975) suggest that the subroutine has performed satisfactorily. Further extensive testing is being carried out and the results will be presented in a later paper.

		$y' = e^x$			$y' = \cos x$		
Step No	x	FSI	ASI	VS	FSI	ASI	VS
1	0.375	0.314E-4	-0.740E-5	-0.740E-5	-0.695E-3	0.132E-4	0.132E-4
2	0.750	-0.248E-4	-0.857E-5	-0.274E-5	0.420E-3	-0.612E-5	-0.954E-5
3	1.125	0.124E-4	-0.619E-5	-0.138E-5	-0.249E-3	-0.201E-4	-0.558E-5
4	1.500	-0.479E-5	-0.315E-5	-0.848E-6	0.110E-3	-0.193E-4	-0.306E-5
5	1.875	0.459E-6	-0.154E-5	-0.603E-6	-0.350E-4	-0.544E-5	-0.176E-5
6	2.250	-0.506E-6	-0.860E-6	-0.486E-6	0.518E-5	0.623E-6	-0.101E-5
7	2.625	-0.418E-6	-0.204E-6	-0.437E-6	-0.574E-6	-0.526E-5	-0.344E-6
8	3.000	-0.420E-6	-0.511E-6	-0.437E-6	0.227E-5	0.156E-4	0.228E-5

TABLE 1. *Local errors for methods of order* 8 *when the step-size is reduced from* 0.75 *to* 0.375

		$y' = e^x$			$y' = \cos x$		
Step No.	x	FSI	ASI	VS	FSI	ASI	VS
1	0.375	0.681E-6	-0.641E-7	-0.641E-7	0.220E-4	-0.507E-5	-0.507E-5
2	0.750	-0.120E-5	-0.963E-7	-0.182E-7	-0.297E-4	-0.549E-5	-0.786E-6
3	1.125	0.165E-5	-0.866E-7	-0.696E-8	0.438E-4	-0.477E-5	-0.148E-6
4	1.500	-0.171E-5	-0.467E-7	-0.322E-8	-0.586E-4	-0.142E-5	-0.200E-7
5	1.875	0.134E-5	-0.169E-7	-0.169E-8	0.692E-4	0.159E-5	0.761E-8
6	2.250	-0.801E-6	-0.152E-7	-0.101E-8	-0.733E-4	0.528E-6	0.132E-7
7	2.625	0.364E-6	-0.176E-7	-0.624E-9	0.761E-4	-0.493E-5	0.155E-7
8	3.000	-0.124E-6	-0.483E-8	-0.452E-9	-0.132E-3	0.547E-5	0.325E-7
9	3.375	0.306E-7	0.214E-8	-0.323E-9	-0.289E-4	-0.157E-4	-0.949E-8
10	3.750	-0.544E-8	-0.602E-8	-0.274E-9	0.290E-5	0.880E-5	-0.110E-8
11	4.125	0.365E-9	0.941E-11	-0.220E-9	-0.303E-6	-0.218E-5	0.476E-9
12	4.500	-0.217E-9	0.242E-8	-0.197E-9	0.192E-7	-0.521E-5	0.101E-8
13	4.875	-0.192E-9	-0.626E-8	-0.192E-9	0.141E-8	0.187E-4	0.138E-8
14	5.250	-0.186E-9	0.749E-8	-0.186E-9	0.180E-8	-0.412E-4	0.182E-8

TABLE 2. *Local errors for methods of order* 14 *when the step-size is reduced from* 0.75 *to* 0.375

		$y' = \cos x\, e^{\sin x}$			$y' = 4e^{2x}/(e^{2x}+1)^2$		
Step No.	x	FSI	ASI	S	FSI	ASI	VS
1	0.250	-0.260E-4	-0.420E-4	-0.420E-4	-0.292E-2	0.485E-4	0.485E-4
2	0.500	0.112E-4	-0.879E-4	-0.415E-4	0.433E-2	-0.976E-3	-0.750E-3
3	0.750	-0.112E-4	-0.923E-4	-0.163E-4	-0.451E-2	-0.173E-2	0.196E-3
4	1.000	0.174E-4	-0.170E-4	0.132E-4	0.383E-2	0.112E-2	0.767E-3
5	1.250	0.173E-4	0.702E-4	0.255E-4	-0.226E-2	0.321E-2	-0.443E-3
6	1.500	0.754E-5	0.314E-4	0.109E-4	-0.143E-3	-0.419E-2	-0.707E-3
7	1.750	-0.196E-4	-0.442E-4	-0.194E-4	0.242E-3	0.424E-2	0.225E-3
8	2.000	-0.377E-4	-0.152E-4	-0.390E-4	0.594E-3	-0.116E-2	0.629E-3

TABLE 3. *Local errors for methods of order 8 when the step-size is reduced from* 0.5 *to* 0.25

		$y' = \cos x\, e^{\sin x}$			$y' = 4e^{2x}/(e^{2x}+1)^2$		
Step No.	x	FSI	ASI	VS	FSI	ASI	VS
1	0.250	0.450E-1	-0.391E-3	-0.391E-3	-0.311E-4	-0.880E-3	-0.880E-3
2	0.500	-0.964E-1	-0.300E-3	0.135E-3	0.897E-3	-0.135E-2	0.482E-3
3	0.750	0.147E+0	0.142E-3	0.767E-5	-0.490E-2	0.221E-2	0.894E-3
4	1.000	-0.218E+0	0.461E-3	-0.267E-5	0.133E-1	0.861E-2	-0.223E-2
5	1.250	0.186E+0	0.152E-3	-0.118E-4	-0.237E-1	-0.418E-2	0.133E-2
6	1.500	-0.189E+0	-0.480E-3	-0.240E-5	0.309E-1	-0.404E-1	0.289E-2
7	1.750	0.107E+0	-0.266E-3	0.166E-4	-0.398E-1	0.178E-1	-0.709E-2
8	2.000	-0.643E-1	0.802E-3	0.703E05	0.402E-1	0.156E+0	0.477E-2
9	2.250	0.248E-1	0.114E-4	-0.281E-4	-0.263E-1	-0.302E+0	0.416E-2
10	2.500	-0.757E-2	-0.158E-2	-0.162E-4	0.145E-2	-0.271E+0	-0.938E-2
11	2.750	0.147E-2	0.287E-2	0.487E-4	0.378E-2	0.112E+1	0.318E-2
12	3.000	-0.959E-4	-0.376E-2	0.373E-4	0.411E-2	0.529E+1	0.427E-2
13	3.250	-0.684E-4	0.533E-2	-0.673E-4	-0.221E-2	0.243E+1	-0.196E-2
14	3.500	-0.812E-4	0.694E-3	-0.794E-4	-0.211E-2	0.168E+1	-0.175E-2

TABLE 4. *Local errors for methods of order* 14 *when the step-size is reduced from* 0.5 *to* 0.25

Step No.	True Value	Using FSI	Using ASI	Using VS
0	1.0	0.030	0.030	0.030
1	1.455	5.506	0.060	0.060
2	2.117	-20.761	0.085	0.116
3	3.080	44.044	0.156	0.218
4	4.482	-55.750	0.291	0.392
5	6.521	49.990	0.478	0.683
6	9.488	-26.965	0.756	1.153
7	13.805	13.032	1.420	1.886
8	20.086	0.861	2.647	2.995
9	29.224	4.993	3.659	4.617
10	42.521	6.914	8.102	6.914

TABLE 5. *11th derivatives of the approximating polynomials when the step-size is reduced for the equation* $y' = e^x$

	$y' = \cos x$			$y' = e^x$		
Step No.	FSI	ASI	VS	FSI	ASI	VS
1	0.112E-3	0.983E-5	0.983E-5	0.474E-5	-0.647E-6	-0.647E-6
2	0.246E-2	0.122E-3	0.445E-4	0.132E-3	-0.851E-5	-0.286E-5
3	-0.207E-2	-0.643E-6	0.392E-5	-0.844E-4	-0.55 E-7	-0.326E-6
4	-0.156E+0	0.134E-3	0.185E-4	-0.236E-2	-0.565E-5	-0.175E-5
5	0.184E+0	-0.181E-4	0.340E-6	0.151E-2	-0.797E-8	-0.222E-6
6	-0.300E+2	0.129E-3	0.374E-4	0.423E-1	-0.412E-5	-0.132E-5
7	0.860E+0	-0.140E-4	0.287E-5	-0.279E-1	0.169E-6	-0.180E-6
8	0.260E+3	0.978E-4	0.130E-4	-0.758E+0	-0.323E-5	-0.114E-5
9	0.936E+0	-0.208E-4	0.111E-5	0.319E+0	0.309E-6	-0.164E-6
10	0.849E+3	0.105E-3	0.386E-5	0.134E+2	-0.280E-5	-0.109E-5

TABLE 6. *Local errors for method of order* 11 *using alternating step-sizes of* 0.75 *and* 0.375

Department of Computer Science
Monash University
Clayton, Victoria, Australia

1. R. K. BRAYTON, F. G. GUSTAVSON & G. D. HACHTEL, (1972), "A new efficient algorithm for solving differential-algebraic systems using implicit backward differentiation formulae," *Proc. IEEE,* v. 60, pp. 98–108.

2. G. D. BYRNE & A. C. HINDMARSH, (1975), "A polyalgorithm for the numerical solution of ordinary differential equations," *ACM Trans. Math. Software,* v. 1, pp. 71–96.

3. W. H. ENRIGHT, T. E. HULL & B. LINDBERG, (1975), "Comparing numerical methods for stiff systems of O.D.E.s," *BIT,* v. 15, pp. 10–48.

4. C. W. GEAR, (1971a), *Numerical Initial Value Problems in Ordinary Differential Equations,* Prentice-Hall, Englewood Cliffs, N. J.

5. C. W. GEAR, (1971b), "Algorithm 407–DIFSUB for solution of ordinary differential equations," *Comm. ACM,* v. 14, pp. 185–190.

6. C. W. GEAR & K. W. TU, (1974), "The effect of variable mesh size on the stability of multistep methods," *SINUM,* v. 11, pp. 1025–1043.

7. G. K. GUPTA, (1975), *New Multistep Methods for the Solution of Ordinary Differential Equations,* Ph. D. Thesis, Monash Univ., Australia. (Unpublished.)

8. G. K. GUPTA, (1976), "Some new high-order multistep formulae for solving stiff equations," *Math. Comp.,* v. 30, pp. 417–432.

9. G. K. GUPTA (1978), *Numerical Testing of the ASI Technique of Step-Size Changing,* Tech. Rep., Dept. of Computer Science, Monash Univ., Victoria, Australia.

10. G. K. GUPTA & C. S. WALLACE, (1975), "Some new multistep methods for solving ordinary differential equations," *Math. Comp.,* v. 29, pp. 489–500.

11. P. HENRICI, (1962), *Discrete Variable Methods in Ordinary Differential Equations,* Wiley, New York.

12. T. E. HULL, W. H. ENRIGHT, B. M. FELLER & A. E. SEDGWICK, (1971), "Comparing numerical methods for ordinary differential equations," *SINUM,* v. 9, pp. 603–637.

13. R. JELTSCH, (1976), "Stiff stability and its relation to A_0 and $A(0)$-stability," *SINUM,* v. 13, pp. 8–17.

14. F. T. KROGH, (1973), "Algorithms for changing the step-size," *SINUM,* v. 10, pp. 949–965.

15. F. T. KROGH, (1969), *A Variable Step Variable Order Multistep Method for the Numerical Solution of Ordinary Differential Equations* (Proc. IFIP Congr., 1968), North-Holland, Amsterdam, pp. 194–199.

16. A. NORDSIECK, (1962), "On the numerical integration of ordinary differential equations," *Math. Comp.,* v. 16, pp. 22–49.

17. P. PIOTROWSKI, (1969), *Stability, Consistency and Convergence of Variable K-Step Methods,* Conference on Numerical Solution of Differential Equations, Lecture Notes in Math., vol. 109, Springer-Verlag, New York and Berlin.

18. A. SEDGWICK, (1973), *An Effective Variable Order Variable Step Adams Method,* Tech. Rep. No. 53, Dept. of Computer Science, Univ. of Toronto, Canada.

19. L. F. SHAMPINE & M. K. GORDON, (1975), "Local error and variable order Adams codes," *Appl. Math. Comput.,* v. 1, pp. 47–66.

20. L. F. SHAMPINE & M. K. GORDON, (1975), *Computer Solution of Ordinary Differential Equations: The Initial Value Problem,* Freeman, San Francisco, Calif.

21. K. W. TU, (1972), *Stability and Convergence of General Multistep and Multivalue Methods with Variable Step Size,* Report No. UIUCDCS-R-72-526, Dept. of Computer Science, Univ. of Illinois, Unbana.

22. C. S. WALLACE & G. K. GUPTA, (1973), "General multistep methods to solve ordinary differential equations," *Austral. Comput. J.,* v. 5, pp. 62–69.