MATHEMATICS OF COMPUTATION
VOLUME 40, NUMBER 161
JANUARY 1983, PAGES 193–206

Block Runge-Kutta Methods for the Numerical Integration of Initial Value Problems in Ordinary Differential Equations Part II. The Stiff Case

By J. R. Cash

Abstract. The approach described in the first part of this paper is extended to include diagonally implicit Runge-Kutta (DIRK) formulae. The algorithms developed are suitable for the numerical integration of stiff differential systems, and their efficiency is illustrated by means of some numerical examples.

1. Introduction. In a recent paper [2], Cash and Bond have derived a class of block cyclic integration formulae suitable for the numerical integration of the stiff differential system

$$\frac{dy}{dx} = f(x, y), \qquad y(x_0) = y_0, \qquad y \in \mathbf{R}^s. \tag{1.1}$$

These formulae were derived and analyzed as block implicit linear multistep methods. However, a much more convenient way of investigating them is to consider them as block diagonally implicit Runge-Kutta formulae. In the second part of this paper we extend the analysis presented in Part I to derive a new class of block DIRK formulae. These formulae have the implicitness necessary for the efficient solution of stiff differential systems but do not call for the large computational effort normally required by fully implicit Runge-Kutta formulae. This new, much more general, approach allows us to derive formulae with many advantages over the formulae derived in [2] and among these advantages are:

(1) Better accuracy.

(2) Better stability.

(3) A better algorithm for estimating the local truncation error and hence for adjusting the step length of integration and varying the order of the formula being used.

The better accuracy and stability comes from the fact that, by writing our formulae as block DIRK methods, we have numerous free parameters at our disposal and these can be used to advantage. The better order and step changing algorithms result from the fact that, although the error estimation procedures given in [2] generally perform well in practice, they are not asymptotically correct in the sense that they do not necessarily yield the correct error estimate in the limit $h = 0$.

Received August 10, 1981; revised February 24, 1982 and June 21, 1982.
1980 *Mathematics Subject Classification.* Primary 65L05.

0025-5718/82/0000-0712/$03.75

Furthermore, there are classes of problems for which the error estimates given in [2] perform very poorly [11]. A direct extension of the error estimation algorithm given in Part I of this paper is inappropriate since this in effect gives an error estimate per unit step, and our numerical experience with block methods indicates that this is not the best way to control the error when integrating stiff differential systems. Instead we will present a different error estimation algorithm which is asymptotically correct as $h \to 0$ and which has been found to perform well in practice. A different way of building up the block will also be described, and this has been found to be much more efficient than that described in [2].

2. The General Approach. In this section we describe our general approach to the problem of deriving efficient block diagonally implicit Runge-Kutta formulae. DIRK formulae were first proposed by Nørsett [14] and have more recently been investigated and extended by Alexander [1], Crouzieux [7] and Cash [5]. The idea of using Runge-Kutta formulae in a block form is an old one—going back at least as far as Milne [13] and more recently developed in the implicit case by Shampine and Watts [16], Williams and de Hoog [17] and Gear [8]. However, investigations of implicit block integration formulae have been based on fully implicit Runge-Kutta formulae, and an investigation of the potentially more efficient class of block DIRK formulae has not been considered. In part this is due to the fact that block formulae are sometimes only proposed as starting procedures for high order linear multistep methods [8]. As we shall see later, DIRK formulae lend themselves naturally to efficient implementation in block form and have several computational advantages over other Runge-Kutta formulae for the integration of stiff differential systems.

As the starting point of our analysis we consider the second order implicit integration formula given in [2]

$$(2.1)\quad \begin{aligned} y_{n+i}^{(1)} &= y_{n+i-1}^{(1)} + hf\left(x_{n+i}, y_{n+i}^{(1)}\right), \qquad i = 1,2, y_n^{(1)} \equiv y_n, \\ y_{n+i}^{(2)} &= y_{n+i-1}^{(2)} + hf\left(x_{n+i}, y_{n+i}^{(2)}\right) - \frac{1}{2}\Delta_h^2 y_n^{(1)}, \qquad i = 1,2,3, y_n^{(2)} \equiv y_n, \end{aligned}$$

where $\Delta_h y_n^{(1)} \equiv y_{n+1}^{(1)} - y_n^{(1)}$ and where we assume that the solution has already been computed up to and including the point (x_n, y_n). Our first step is to express this formula as a Runge-Kutta method of the general form

$$(2.2)\quad y_{n+1} - y_n = h\sum_{i=1}^{q} b_i k_i, \qquad k_i = f\left(x_n + c_i h, y_n + h\sum_{j=1}^{q} a_{ij}k_j\right), \qquad 1 \leqslant i \leqslant q.$$

A particularly convenient way of expressing our Runge-Kutta formula is to use the well-known "Butcher Matrix" notation whereby (2.2) is represented in the form

$$(2.3)\quad \begin{array}{c|c} c & A \\ \hline & b^T \end{array}.$$

Here A is the $q \times q$ matrix a_{ij} and b, c are vectors of dimension q. Using this notation we can write formula (2.1) for the computation of $y_{n+3}^{(2)}$ as

$$(2.4)\qquad \begin{array}{c|ccccc} 1 & 1 & & & & \\ 2 & 1 & 1 & & & \\ 1 & 1/2 & -1/2 & 1 & & \\ 2 & 1 & -1 & 1 & 1 & \\ 3 & 3/2 & -3/2 & 1 & 1 & 1 \\ \hline & 3/2 & -3/2 & 1 & 1 & 1 \end{array}$$

see also [6, p.115]. This formula has the important property that as well as providing a second order solution at x_{n+3} it also provides both first and second order solutions at x_{n+1} and x_{n+2}. In addition, if (2.4) is expressed in the more general form

$$(2.5)\qquad \begin{array}{c|ccccc} 1 & a_{11} & & & & \\ 2 & a_{21} & a_{22} & & & \\ 1 & a_{31} & a_{32} & a_{33} & & \\ 2 & a_{41} & a_{42} & a_{43} & a_{44} & \\ 3 & a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \\ \hline & a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{array}$$

the formula used to compute $y_{n+i}^{(j)}, j \in [1,2], i \in [1,2]$ and ; $j = 2, i = 3$ is

$$(2.6)\qquad y_{n+i}^{(j)} = y_n + h \sum_{m=1}^{2(j-1)+i} a_{2(j-1)+i,m} k_m .$$

Thus, for example, the formulae used to compute $y_{n+1}^{(1)}, y_{n+2}^{(1)}, y_{n+1}^{(2)}, y_{n+2}^{(2)}, y_{n+3}^{(2)}$ are respectively

$$\begin{array}{c|c} 1 & 1 \\ \hline & 1 \end{array} \qquad \begin{array}{c|cc} 1 & 1 & \\ 2 & 1 & 1 \\ \hline & 1 & 1 \end{array} \qquad \begin{array}{c|ccc} 1 & 1 & & \\ 2 & 1 & 1 & \\ 1 & 1/2 & -1/2 & 1 \\ \hline & 1/2 & -1/2 & 1 \end{array} \qquad \begin{array}{c|cccc} 1 & 1 & & & \\ 2 & 1 & 1 & & \\ 1 & 1/2 & -1/2 & 1 & \\ 2 & 1 & -1 & 1 & 1 \\ \hline & 1 & -1 & 1 & 1 \end{array} \qquad \text{and (2.4).}$$

We see that our five stage block formula (2.4) provides 5 separate solutions and, as will be seen later, this rule generalizes in that our general m-stage formula yields m separate solutions.

Our aim now is to generalize formula (2.4) to (2.5) and to choose the extra free parameters at our disposal so as to improve the computational efficiency of our method. This is not entirely straightforward because formula (2.4) has proved to be remarkably efficient. What we need to do is to isolate the precise reasons for this efficiency and then to make sure that (2.5) also shares these properties. The main reasons for the efficiency of (2.4) are:

(1) At each step we solve for a $y_{n+i}^{(j)}$ value rather than for an "f value". This property, which is also shared by some other implicit Runge-Kutta formulae, makes the process of prediction much simpler, and also we are able to use many of the well tried devices incorporated by Hindmarsh [9] in his version of GEAR. In particular,

when computing $y_{n+1}^{(2)}$, $y_{n+2}^{(2)}$, we already have the (normally very good) approximations $y_{n+1}^{(1)}$, $y_{n+2}^{(1)}$ available and, furthermore, we have a built-in local error estimate $y_{n+i}^{(2)} - y_{n+i}^{(1)}$ available for the error in $y_{n+i}^{(1)}$, $i = 1, 2$. This is highlighted if we consider the first three steps of (2.1) which are

$$y_{n+1}^{(1)} = y_n + hf\left(x_{n+1}, y_{n+1}^{(1)}\right),$$
$$y_{n+2}^{(1)} = y_{n+1}^{(1)} + hf\left(x_{n+2}, y_{n+2}^{(1)}\right),$$
$$y_{n+1}^{(2)} = y_n + hf\left(x_{n+1}, y_{n+1}^{(2)}\right) - \frac{1}{2}\left(y_{n+2}^{(1)} - 2y_{n+1}^{(1)} + y_n\right).$$

We see that when computing $y_{n+1}^{(2)}$ we already have the iterate $y_{n+1}^{(1)}$ available and, furthermore, $f(x_{n+1}, y_{n+1}^{(1)})$ will also be available. Thus the computation of $y_{n+1}^{(2)}$ is normally very cheap. Our numerical experience has shown that the five-stage block formula (2.4) generally requires considerably less computational effort than is required by a conventional five-stage DIRK formula. Also formula (2.4) obtains second order accuracy at three grid points using 5 stages, whereas, for Alexander's second order formula, to obtain comparable accuracy at the same three grid points, 6 stages are required. Furthermore, for reasons just explained, the computation required to use (2.4) is normally much less than it would be for Alexander's or similar schemes (see also the results of Section 6). Finally, we remark that all formulae in this paper require less function evaluations per step than are required by conventional DIRK formulae of the same order to achieve comparable accuracy. This leads us to make the important point that, although in the explicit case we can normally compare the efficiency of two (nonblock) Runge-Kutta formulae of the same order by counting the number of stages, in the case of implicit Runge-Kutta formulae such a comparison may not be valid. Indeed, when comparing block DIRK formulae with conventional DIRK formulae such a criterion is almost invariably of little value since stages in block formulae normally require much less computational effort than stages in conventional DIRK formulae. The other properties we require, and these are shared by some other implicit Runge-Kutta formulae, are:

(2) Our formulae are strongly S-stable [15].

(3) Our formulae simultaneously produce approximations at a sequence of points. In general we require that our pth order formulae should produce pth order solutions at $p + 1$ equally spaced points.

(4) The coefficient matrix of the modified Newton scheme used to solve for $y_{n+i}^{(j)}$ is independent of both i and j. This property is recognized as a very useful one for a numerical integration method to possess.

The reason why we consider the equi-distribution of errors approach has been explained in the first part of this paper and need not be elaborated upon. Thus, in conclusion, we can say that we adopt the particular approach described in this paper because a) the equi-distribution of errors approach offers some important computational advantages which are discussed in Part I, b) our formulae have the advantageous properties (1)–(4) just described, c) our formulae of orders (2)–(5) require less function evaluations per step than are required by conventional S-stable DIRK formulae to achieve comparable accuracy, and furthermore, as will be explained in Section 5, our formulae can be implemented more efficiently than conventional formulae. All of this indicates that our formulae should be significantly more

efficient than standard DIRK formulae, and this expectation is borne out by the numerical results presented in Section 6. We also note that, since each block is self-starting, it is only the last LTE in the block which is propagated forward. If an accurate solution was required only at the *end* of the range of integration, we would not require the equi-distribution of errors property but would only require an accurate solution at the end of each block. However we will not consider this problem in the present paper.

Our aim now is to derive a class of block DIRK formulae which allow variable step and variable order but which maintain the computational advantages (1)–(4) outlined above.

3. The General Class of Formulae. One particular class of formulae which satisfies the requirements set out in Section 2 has the form (2.3), where A is a lower triangular matrix with unit diagonal and $c = (1, 2, 1, 2, 3, 1, 2, 3, 4, \ldots)^T$. There are several important computational and theoretical consequences arising from considering this class of formulae, and these can best be explained by reconsidering our second order formula (2.5). It is straightforward to verify that if (2.5) is to yield first order approximations at $n + 1$, $n + 2$ together with second order approximations at $n + 1$, $n + 2$, $n + 3$, it must have the particular form

$$
\begin{array}{c|cccc}
1 & 1 & & & \\
2 & 1 & 1 & & \\
1 & 1/2 & -1/2 & 1 & \\
2 & 2d_1 & -1 & 2-2d_1 & 1 \\
3 & \frac{5}{2}-3d_2 & 3d_3 & 3d_2 & -\frac{1}{2}-3d_3 \quad 1 \\
\hline
 & \frac{5}{2}-3d_2 & 3d_3 & 3d_2 & -\frac{1}{2}-3d_3 \quad 1
\end{array}
\tag{3.1}
$$

Two important theoretical points which we wish to make are as follows. Since, for example, $y_{n+3}^{(2)}$ is computed over a step length $3h$ rather than h, Butcher's general analysis for Runge-Kutta methods [3] is not immediately applicable. However, because of our special choice of c_i, the order relations for our formulae still take a very simple form. For example, the conditions for the block formula (2.6) to have order 4 are

$$
\begin{aligned}
&\sum_m a_{2(j-1)+i,m} = i, && \sum_m a_{2(j-1)+i,m}c_m = \frac{1}{2}i^2, \\
&\sum_m a_{2(j-1)+i,m}c_m^2 = \frac{1}{3}i^3, && \sum_{m,n} a_{2(j-1)+i,m}a_{m,n}c_n = \frac{1}{6}i^3, \\
&\sum_m a_{2(j-1)+i,m}c_m^3 = \frac{1}{4}i^4, && \sum_{m,n} a_{2(j-1)+i,m}c_m a_{m,n}c_n = \frac{1}{8}i^4, \\
&\sum_{m,n} a_{2(j-1)+i,m}a_{m,n}c_n^2 = \frac{1}{12}i^4, && \sum_{m,n,p} a_{2(j-1)+i,m}a_{m,n}a_{n,p}c_p = \frac{1}{24}i^4,
\end{aligned}
\tag{3.2}
$$

for $j = 1, 2$. This is equivalent to multiplying all coefficients in (3.1) by $\frac{1}{3}$ and using Butcher's analysis. These order relations generalize for higher order in a straightforward way. The second point which we make is that, because of the special form

taken by Eq. (3.1) we have

$$\sum_j a_{ij}c_j = \frac{1}{2}c_i^2, \qquad i = 3,4,5. \tag{3.3}$$

For higher order equations there are several more relationships of this type, and this makes the derivation of high order equations particularly straightforward since many of the order relations are trivially satisfied. It is worth remarking that in order to derive our fifth order formula, given in the next section, we need only to solve *six linear* equations.

We now return to the problem of choosing the free coefficients appearing in (3.1). We will choose two coefficients to get an equi-distribution of errors and the other coefficient to ensure L-stability. The principal terms in the local truncation errors associated with the second order formulae are

$$\text{at } n+1 \qquad \frac{5}{12}h^3f_{yy}f^2 + \frac{2}{3}h^3f_y^2f, \tag{3.4a}$$

$$\text{at } n+2 \qquad \frac{1}{3}h^3f_{yy}f^2 + \frac{8}{6}h^3\left[1 - \frac{3}{4}d_1\right]f_y^2f, \tag{3.4b}$$

$$\text{at } n+3 \qquad -\frac{1}{4}h^3f_{yy}f^2 + \frac{27}{6}h^3\left[-\frac{1}{3} - \frac{2d_3}{3} + \frac{d_2}{3}\right]f_y^2f. \tag{3.4c}$$

To get a reasonably equal distribution of errors we choose $d_1 = 3/2$, $d_2 = 11/9 + 2d_3$. This leads to the final second order block formula

$$\begin{array}{c|ccccc} 1 & 1 & & & & \\ 2 & 1 & 1 & & & \\ 1 & \frac{1}{2} & -\frac{1}{2} & 1 & & \\ 2 & 3 & -1 & -1 & 1 & \\ 3 & -\frac{7}{6} - 6b & 3b & \frac{11}{3} + 6b & -\frac{1}{2} - 3b & 1 \\ \hline & -\frac{7}{6} - 6b & 3b & \frac{11}{3} + 6b & -\frac{1}{2} - 3b & 1 \end{array} \tag{3.5}$$

and it is easy to show that a sufficient condition for this formula to be strongly S-stable is $|b| \leqslant 1$. The easiest way of comparing the local truncation errors of the original formula (2.4) and the modified formula (3.5) is to consider a linear equation since in this case no cancellation of terms in the local truncation errors can occur. A simple calculation shows that for linear equations the LTE's associated with (2.4) are up to 5 times as large as those associated with (3.5), and our numerical experiments have confirmed that (3.5) is generally the more accurate of the two formulae for both linear and nonlinear problems.

4. Some Particular Formulae. Having formulated our general approach in the previous section we are now in a position to give some particular formulae. We shall again list our second order formula, both for the sake of completeness and also because it allows us to describe the precise way in which our formulae are being presented.

Order 1.

$$\begin{array}{c|cc} 1 & 1 & \\ 2 & 1 & 1 \\ \hline & & \end{array}.$$

Strongly S-stable giving first order solutions at $n+1$ and $n+2$.

Order 2.

$$\begin{array}{c|ccc} 1 & \frac{1}{2} & -\frac{1}{2} & 1 \\ 2 & 3 & -1 & -1 \quad 1 \\ 3 & -\frac{7}{6} - 6b & 3b & \frac{11}{3} + 6b \quad -\frac{1}{2} - 3b \quad 1 \\ \hline & & & \end{array} \qquad b = 0.256.$$

Strongly S-stable giving second order solutions at $n+1$, $n+2$, $n+3$. Thus, to get our complete order 2 formula we join the order 2 block onto the bottom of the order 1 block (see Eq. (3.5)). This procedure extends in an obvious way for all orders. Thus, for example, to get the complete third order formula we join the order 3 block given below to the bottom of the order 2 block and join the combined blocks to the bottom of the order 1 block giving a 9 stage formula.

Order 3.

$$\begin{array}{c|ccccccccc} 1 & 2d_4 + 8/3 & -4/3 - d_4 & -21/12 - 2d_4 & d_4 & 5/12 & 1 & & & \\ 2 & 2e_4 + 10/3 & -5/3 - e_4 & e_3 & e_4 & 1/3 & -1 - e_3 - 2e_4 & 1 & & \\ 3 & 0 & 0 & f_3 & f_4 & -1/4 & 9/4 - f_3 & -f_4 & 1 & \\ 4 & -\frac{10}{3} + 2g_4 + 2g_7 & \frac{5}{3} - g_4 - g_7 & 5 - 2g_4 - g_6 - 2g_7 & g_4 & g_5 & g_6 & g_7 & -\frac{1}{3} - g_5 & 1 \\ \hline & & & & & & & & & \end{array}$$

$$d_4 = \frac{36}{25}, \quad e_3 = -\frac{1}{2}, \quad e_4 = -2, \quad f_3 = \frac{5}{84}, \quad f_4 = -\frac{11}{42}, \quad g_7 = 5,$$

$$g_4 = -\frac{10}{3}, \quad g_6 = \frac{10}{11}, \quad g_5 = -\frac{35}{11}.$$

This formula is $S(\alpha)$-stable with $\alpha = 89.98°$. Third order solutions are produced at $n+1, n+2, n+3, n+4$.

Order 4.

$$\begin{array}{c|ccccccccccccc} 1 & 0 & 0 & p_{31} & p_{32} & p_{33} & p_{34} & p_{35} & p_{36} & p_{37} & 1 & & & \\ 2 & 0 & 0 & q_{41} & q_{42} & q_{43} & q_{44} & q_{45} & q_{46} & q_{47} & q_{48} & 1 & & \\ 3 & 0 & 0 & w_{51} & w_{52} & w_{53} & w_{54} & w_{55} & w_{56} & w_{57} & w_{58} & w_{59} & 1 & \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8/3 & -1 & 8/3 & -4/3 & 0 & 1 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{79}{24} & -\frac{91}{24} & \frac{149}{24} & -\frac{41}{24} \quad 1 \\ \hline & & & & & & & & & & & & & \end{array}$$

$$p_{31} = -0.41, \quad p_{32} = -22p_{31}/5, \quad p_{33} = 5p_{31}/3 + 4p_{32}/3, \quad p_{34} = 31/24 - p_{31},$$
$$p_{35} = -59/24 - p_{32}, \quad p_{36} = 37/24 - p_{33}, \quad p_{37} = -3/8.$$
$$q_{41} = -0.23, \quad q_{42} = -22q_{41}/5, \quad q_{43} = 5q_{41}/3 + 4q_{42}/3, \quad q_{44} = 1.19,$$
$$q_{45} = -8/3 - q_{42}, \quad q_{46} = 4/3 - q_{43}, \quad q_{47} = -1/3, \quad q_{48} = 8/3 - q_{41} - q_{44},$$
$$w_{51} = 0.08, \quad w_{52} = -22w_{51}/5, \quad w_{53} = 5w_{51}/3 + 4w_{52}/3, \quad w_{54} = 0.55,$$
$$w_{55} = -0.49, \quad w_{56} = 7/8 - w_{53}, \quad w_{57} = -3/8, \quad w_{58} = 63/24 - w_{51} - w_{54},$$
$$w_{59} = -9/8 - w_{52} - w_{55}.$$

This formula is strongly S-stable and fourth order solutions are produced at $n+1$, $n+2$, $n+3$, $n+4$, $n+5$. The PLTE at $n+4$ is $-(14/45)h^5y_n^{\mathrm{v}}$ while that at $n+5$ is $(49/144)h^5y_n^{\mathrm{v}}$.

Order 5.

1	0	0	0	0	0	0	0	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	1			
2	0	0	0	0	0	0	0	a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}	a_{27}	a_{28}	1		
3	0	0	0	0	0	0	0	a_{31}	a_{32}	a_{33}	a_{34}	a_{35}	a_{36}	a_{37}	a_{38}	a_{39}	1	
4	0	0	0	0	0	0	0	a_{41}	a_{42}	a_{43}	a_{44}	a_{45}	a_{46}	a_{47}	a_{48}	a_{49}	a_{410}	1
5	0	0	0	0	0	0	0	0	0	0	0	0	a_{51}	a_{52}	a_{53}	a_{54}	a_{55}	a_{56}	1

$$a_{11} = 1.21, \quad a_{12} = -\frac{363}{800}, \quad a_{13} = \frac{1181}{720}, \quad a_{14} = -\frac{51319}{13320}, \quad a_{15} = \frac{727}{300},$$

$$a_{16} = -\frac{9473}{7200}, \quad a_{17} = \frac{251}{720}, \quad a_{21} = 0.6, \quad a_{22} = -0.225, \quad a_{23} = 1.88,$$

$$a_{24} = -\frac{178}{45}, \quad a_{25} = 8/3, \quad a_{26} = -\frac{503}{360}, \quad a_{27} = \frac{29}{90}, \quad a_{28} = \frac{499}{450},$$

$$a_{31} = -0.26, \quad a_{32} = 0.0975, \quad a_{33} = 1.41, \quad a_{34} = -3.2, \quad a_{35} = 3.16,$$

$$a_{36} = -\frac{729}{400}, \quad a_{37} = \frac{27}{80}, \quad a_{38} = \frac{621}{400}, \quad a_{39} = \frac{29}{40}, \quad a_{41} = \frac{7}{5},$$

$$a_{42} = -\frac{21}{40}, \quad a_{43} = -1, \quad a_{44} = \frac{1}{2}, \quad a_{45} = 2, \quad a_{46} = -\frac{619}{360},$$

$$a_{47} = \frac{28}{90}, \quad a_{48} = \frac{358}{90}, \quad a_{49} = -\frac{277}{90}, \quad a_{410} = \frac{51}{45}, \quad a_{51} = -\frac{1715}{4608},$$

$$a_{52} = -\frac{49}{144}, \quad a_{53} = \frac{425}{144}, \quad a_{54} = -\frac{175}{72}, \quad a_{55} = \frac{25}{6}, \quad a_{56} = \frac{115}{4608}.$$

This formula is $S(\alpha)$-stable with $\alpha > 89°$ and produces fifth order solutions at $n+1$, $n+2$, $n+3$, $n+4$, $n+5$. The PLTE at $n+5$ is $(475/1440)h^6y_n^{\mathrm{vi}}$.

5. Computational Aspects. The main problems with which we shall concern ourselves in this section are:

(1) The solution of the algebraic equations defining the required solution.

(2) An algorithm for adjusting the step length of integration.

(3) An algorithm for varying order.

Our algorithm for solving the algebraic equations requires little comment since it is based on a modified Newton iteration scheme, and these are now widely used in practice. When obtaining an initial approximation to $y_{n+i}^{(j)}$ we use

(a) $y_{n+i}^{(j-1)}$ if this has been computed.

(b) $y_{n+i-1}^{(j)}$ otherwise.

We choose to use these initial approximations since the function values (or close approximations to them) associated with these quantities have already been computed, and this normally makes the computation of the next iterate particularly cheap. Thus, when computing $y_{n+i}^{(j)}$, the equation which we are seeking an approximate solution of is

$$y_{n+i}^{(j)} = \sum + hf\left(x_{n+i}, y_{n+i}^{(j)}\right), \tag{5.1a}$$

where Σ is a known sum. We seek to find a solution of this using the modified Newton iteration scheme

$$(5.1b)\quad [I - hJ]\left[y^{(j)}_{n+i,p} - y^{(j)}_{n+i,p-1}\right] = -y^{(j)}_{n+i,p-1} + hf\left(x_{n+i}, y^{(j)}_{n+i,p-1}\right) + \Sigma,$$
$$p = 1,2,\ldots,M, y^{(j)}_{n+i} \equiv y^{(j)}_{n+i,M}.$$

If $j > 1$, we can use the initial approximation $y^{(j)}_{n+i,0} = y^{(j-1)}_{n+i}$ and thus an approximation to the derivative $f(x_{n+i}, y^{(j)}_{n+i,0})$ will also be available. In view of this, when applying (5.1b) with $p = 1$, we require no function evaluation and only one backsolve. It is this property in particular which makes our schemes computationally efficient and explains why, for the results of Section 6, a backsolve is not necessarily accompanied by a function evaluation.

For the modified Newton scheme (5.1b) a minimum of four iterations were allowed and a solution of (5.1b) was accepted when two iterates differed in a weighted least-squares norm by less than 2 Tol/5, where Tol is the requested tolerance. Solutions at off-step points were computed in a straightforward way by interpolation. Finally, we note that since our iterates are an approximation to the solution of (5.1a), once that our final iterate $y^{(j)}_{n+i}$ has been computed, we can compute an approximation $\hat{f}(x_{n+i}, y^{(j)}_{n+i})$ to $f(x_{n+i}, y^{(j)}_{n+i})$ using the relation

$$(5.1c)\qquad \hat{f}\left(x_{n+i}, y^{(j)}_{n+i}\right) = \left(y^{(j)}_{n+i} - \Sigma\right)/h.$$

We prefer this to computing $f(x_{n+i}, y^{(j)}_{n+i})$ since the latter amplifies the errors in $y^{(j)}_{n+i}$ which may result in instability and also it saves one function evaluation.

To explain our algorithms for changing stepsize and order it is convenient to consider our first order block scheme which we write symbolically as

$$\begin{array}{|cc|}\hline \text{1st order} & 1 \quad 2 \\ \hline \text{2nd order} & 3 \quad 4 \\ \hline \end{array}.$$

Our order and stepchanging algorithms have been greatly influenced by STRIDE [4]. In particular, the embedded estimate $y^{(2)}_{n+i} - y^{(1)}_{n+i}$ is used to approximate the error in the first order solutions $y^{(1)}_{n+i}$, $i = 1,2$. Having obtained these error estimates, we investigate whether or not it is economical to increase order. To do this we need to estimate the local truncation error in $y^{(2)}_{n+2}$ and this brings us to an interesting point regarding truncation errors. If we consider the backward Euler rule, the usual expression for the local truncation error on the assumption $y_n = y(x_n)$ is

$$(5.2)\qquad \text{T.E.} \equiv y(x_{n+1}) - y(x_n) - hf(x_{n+1}, y(x_{n+1}))$$
$$= \frac{-h^2}{2}y''(x_n) - \frac{h^3}{3}y'''(x_n) + O(h^4).$$

However, if we apply Butcher's analysis, we obtain

$$(5.3)\qquad \text{T.E.} = -\frac{h^2}{2}y''(x_n) - \frac{h^3}{3}f_{yy}f^2 - \frac{5}{6}h^3f_y^2f$$

which is different from (5.2). This is only a normalization difference but it does present us with the problem of deciding exactly what we are going to approximate in practice. We feel that Butcher's definition is more appropriate, and so we use this from now on. Using this analysis, we obtain the following relations:

$$y(x_{n+1}) - y^{(1)}_{n+1} = -\frac{h^2}{2} f_y f - \frac{h^3}{3} f_{yy} f^2 - \frac{5}{6} h^3 f_y^2 f + O(h^4), \tag{5.4}$$

$$y(x_{n+2}) - y^{(1)}_{n+2} = -h^2 f_y f - \frac{7}{6} h^3 f_{yy} f^2 - \frac{8}{3} h^3 f_y^2 f + O(h^4),$$

$$y(x_{n+1}) - y^{(2)}_{n+1} = \frac{5}{12} h^3 f_{yy} f^2 + \frac{2}{3} h^3 f_y^2 f + O(h^4),$$

$$y(x_{n+2}) - y^{(2)}_{n+2} = \frac{1}{3} h^3 f_{yy} f^2 - \frac{1}{3} h^3 f_y^2 f + O(h^4).$$

Also

$$z_n \equiv -\frac{h}{2}\left\{ f\left(x_{n+2}, y^{(2)}_{n+2}\right) - 4f\left(x_{n+1}, y^{(2)}_{n+1}\right) + 3f(x_n, y_n)\right\} \tag{5.5}$$

$$= h^2 f_y f + O(h^4).$$

Eliminating the second order terms in h from (5.4) and solving for $h^3 f_{yy} f^2$ and $h^3 f_y^2 f$, we obtain

$$P_n \equiv 2\left(y^{(2)}_{n+1} - y^{(1)}_{n+1}\right) - \left(y^{(2)}_{n+2} - y^{(1)}_{n+2}\right) = -\frac{1}{2} h^3 f_y^2 f + O(h^4), \tag{5.6a}$$

$$-\frac{4}{3}\left\{ y^{(2)}_{n+1} - y^{(1)}_{n+1} + \frac{1}{2} z_n - 3P_n \right\} = h^3 f_{yy} f^2 + O(h^4). \tag{5.6b}$$

Using these two relations we can estimate the local truncation errors TE1, TE2 in the second order solutions $y^{(2)}_{n+1}, y^{(2)}_{n+2}$.

Defining

$$E_1 = \max_{j=1,2} \left\| y^{(2)}_{n+j} - y^{(1)}_{n+j} \right\|, \qquad E_2 = \max\{\|\mathrm{TE1}\|, \|\mathrm{TE2}\|\},$$

where

$$\|\cdot\| \equiv \left\{ \frac{1}{s} \sum_{l=1}^{s} \left[\left[y^{(2)}_{n+j,l} - y^{(1)}_{n+j,l} \right] \Big/ \mathrm{Max}\left(1, y^{(1)}_{n+j,l}\right) \right]^2 \right\}^{1/2},$$

we can compute

$$h_i = h(\mathrm{Tol}/E_i)^{1/(i+1)}, \qquad i = 1, 2, \tag{5.7}$$

which is the predicted step that can be used with the ith order formula. [In practice we use a slightly smaller step size to minimize the likelihood of block rejection.] This enables us to compute

$$w_1 = 2h_1/4, \qquad w_2 = 3h_2/8,$$

which gives a measure of the distance that can be integrated forward per iterate computed (cf. [2, p. 442]). If $w_1 > w_2$, we accept the first order solutions and carry on to the next block. If $w_2 > w_1$, we compute the solution $y^{(2)}_{n+3}$, local extrapolation is performed at $n + 1$, $n + 2$ so that $y^{(2)}_{n+1}$, $y^{(2)}_{n+2}$ are the finally accepted solutions, and we continue forward with the second order scheme.

We denote the second order scheme symbolically by

1st order	1	2	
2nd order	3	4	5
3rd order	6	7	8

We see that error estimates for the first and second order solutions are immediately available through embedding, and an error estimate in the third order solution can be computed using an extension of the procedure previously described. Thus we can decide whether to keep the order fixed, decrease it by one or increase it by one, and this is what is normally done with linear multistep methods. Order changing for higher order equations is simplified by constructing the fourth and fifth order formulae so that the PLTE's at the end of the blocks are total derivatives. These can be estimated in the usual way by finite difference approximations and errors in lower order solutions are estimated by embedding. Using these error estimates the order changing algorithm just described extends in a natural way.

Finally, we note that once a stepsize h' has been computed using (5.7) the actual stepsize taken, $\bar{h}$, is less than the computed stepsize h' with the relationships being

$$\begin{aligned} \bar{h} &= 0.7h' \quad \text{order} = 1, \\ \bar{h} &= 0.75h' \quad \text{order} = 2, \\ \bar{h} &= 0.8h' \quad \text{order} = 3, \\ \bar{h} &= 0.85h' \quad \text{order} = 4. \end{aligned}$$

It was found to be advisable to adopt this strategy since it minimizes the likelihood of expensive block rejections; see also [9].

6. Numerical Results. In this section we shall compare our algorithms with existing DIRK methods. We will show that for the test problems considered our order and step changing strategies are satisfactory and the results obtained are competitive with those previously published for DIRK methods. We have also made comparisons on other test problems using *our* implementation of Alexander's scheme, and the results which we list in this section seem to be fairly typical. The three test problems considered are

(1) $$\begin{aligned} y_1' &= 0.01 - (0.01 + y_1 + y_2)(y_1^2 + 1001\,y_1 + 1001), \qquad y_1(0) = 0, \\ y_2' &= 0.01 - (0.01 + y_1 + y_2)(1 + y_2^2), \qquad y_2(0) = 0, \end{aligned}$$

(2) C1,

(3) C5,

where C1 and C5 are test problems taken from Enright, Hull and Lindberg [5, p. 298]. These test problems were chosen firstly because they allow us to compare our results with certain previously published ones [5] and secondly because Alexander [1] reported difficulty in solving at least one of these problems. In Table 1 we compare the performance of Alexander's strongly S-stable third order scheme [1], the third order scheme given by Cash [5, p. 295] and the third order block scheme of Section 4. For the first two methods a function evaluation always entails a back substitution for the Newton iteration scheme. However, as was explained in detail in Section 5, this is not the case for our block scheme, and so we explicitly list the number of backsolves required. Because many of our Newton iterations use an already computed function approximation (see Section 5) the number of function evaluations required by our block methods at low tolerances is very small and this is reflected in the results of Tables 1, 2. As can be seen from Table 1, the gain in efficiency of the block scheme over the other two DIRK schemes is significant. The headings in Table 1 are self-explanatory apart from Rel. Err. which gives the relative error at the end of

TABLE 1

	Tol.	Alexander [5] F^n	JAC	STEPS	Rel ERR	[5] F^n	JAC	STEPS	Rel ERR	Third Order Block F^n	BACKSOLVES	JAC	STEPS	Rel ERR
P1	10^{-2}	1208	82	30	.96 E–4	929	25	48	.21 E–3	17	120	8	32	.10 E–1
	10^{-3}	1399	110	33	.91 E–4	1093	37	51	.54 E–5	71	243	16	44	.16 E–2
	10^{-4}	2059	176	43	.34 E–5	1474	44	70	.10 E–6	179	420	23	60	.10 E–3
	10^{-5}	3378	298	63	.29 E–6	1990	64	85	.60 E–8	448	899	47	96	.55 E–5
P2	10^{-2}	799	24	24	.69 E–2	526	11	27	.47 E–2	53	247	14	60	.17 E–0
	10^{-3}	1297	26	38	.12 E–2	1338	12	68	.55 E–3	123	395	16	84	.31 E–2
	10^{-4}	2229	28	66	.96 E–4	4265	9	205	.17 E–3	336	738	12	124	.21 E–3
	10^{-5}	–				–				734	1410	13	200	.52 E–5
P3	10^{-2}	1333	78	26	.77 E–2	1577	15	64	.40 E–5	164	503	16	104	.61 E–3
	10^{-3}	1954	72	40	.42 E–5	3239	14	132	.77 E–5	463	1166	18	204	.35 E–5
	10^{-4}	3738	66	80	.97 E–6	7640	16	323	.84 E–6	1229	2685	26	400	.33 E–8

the range of integration. Finally we should emphasize most strongly that the implementations of the algorithms discussed in this paper have purposely been kept very simple and are in no way optimal. This is done so that we can easily compare actual methods themselves rather than sophisticated implementations of different methods. It is to be expected, as our preliminary results together with the results of Alexander [1] have shown, that a proper implementation of the DIRK methods discussed would give much greater efficiency than is indicated by the results of Tables 1, 2.

In Table 2 we present the results obtained using our variable order algorithm based on the ideas of the previous section. Again we see that the increase in efficiency over the schemes presented in [1], [5] is significant. We have also found that, over all tolerances our variable order algorithm is more efficient than fixed order algorithms based on the methods of Section 4 and this leads us to believe that variable step-variable order block DIRK formulae offer a promising approach to the numerical integration of stiff differential systems.

Acknowledgement. The author is grateful to Professor J. D. Lambert for many helpful suggestions and also to an anonymous referee, whose many comments greatly improved this paper.

TABLE 2
Results for variable order

	Tol.	F^n	BACKSOLVES	JAC	STEPS	Rel ERR
P1	10^{-2}	7	30	6	12	.12 E-1
	10^{-3}	65	139	20	27	.58 E-3
	10^{-4}	155	298	28	49	.86 E-4
	10^{-5}	339	751	39	97	.10 E-4
P2	10^{-2}	63	179	15	50	.37 E-2
	10^{-3}	135	489	20	97	.37 E-3
	10^{-4}	310	839	19	142	.11 E-3
	10^{-5}	738	1552	12	217	.75 E-5
P3	10^{-2}	134	260	16	59	.48 E-5
	10^{-3}	577	1466	16	237	.41 E-7
	10^{-4}	1188	2798	20	427	.23 E-7

Department of Mathematics
Imperial College
South Kensington
London S.W.7, England

1. R. ALEXANDER, "Diagonally implicit Runge-Kutta methods for stiff ordinary differential equations," *SIAM J. Numer. Anal.*, v. 14, 1977, pp. 1006–1021.

2. J. BOND & J. R. CASH, "A block method for the numerical integration of stiff systems of ordinary differential equations," *BIT*, v. 19, 1979, pp. 429–447.

3. J. C. BUTCHER, "Coefficients for the study of Runge-Kutta integration processes," *J. Austral. Math. Soc.*, v. 3, 1963, pp. 185–201.

4. J. C. BUTCHER, K. BURRAGE & F. H. CHIPMAN, *STRIDE: Stable Runge-Kutta Integrator for Differential Equations*, Report No. 20, Dept. of Mathematics, University of Auckland, New Zealand, 1979.

5. J. R. CASH, "Diagonally implicit Runge-Kutta formulae with error estimates," *J. Inst. Math. Appl.*, v. 24, 1979, pp. 293–301.

6. J. R. CASH, *Stable Recursions, with Application to the Numerical Solution of Stiff Systems*, Academic Press, London and New York, 1979.

7. M. CROUZIEUX, *Sur l'Approximation des équations Différentielles Opérationnelles Linéaires par des Méthodes de Runge-Kutta*, Ph.D. thesis, University of Paris, 1975.

8. C. W. GEAR, "Runge-Kutta starters for multistep methods," *ACM Trans. Math. Software*, v. 6, 1980, pp. 263–279.

9. A. C. HINDMARSH, *GEAR: Ordinary Differential Equation System Solver*, Rep. UCID-30001, Rev. 3, Lawrence Livermore Laboratory, Livermore, Calif., 1974.

10. K. R. JACKSON & R. SACKS-DAVIS, "An alternative implementation of variable step-size multistep formulas for stiff ODEs," *ACM Trans. Math. Software*, v. 6, 1980, pp. 295–318.

11. J. D. LAMBERT. Private communication, 1980.

12. B. LINDBERG, "Characterization of optimal stepsize sequences for methods for stiff differential equations," *SIAM J. Numer. Anal.*, v. 14, 1977, pp. 859–887.

13. W. E. MILNE, *Numerical Solution of Differential Equations*, Wiley, New York, 1953.

14. S. P. NØRSETT, *Semi-Explicit Runge-Kutta Methods*, Mathematics and Computation, No. 6, University of Trondheim, 1974.

15. A. PROTHERO & A. ROBINSON, "On the stability and accuracy of one-step methods for solving stiff systems of ordinary differential equations," *Math. Comp.*, v. 28, 1974, pp. 145–162.

16. L. F. SHAMPINE & H. A. WATTS, "A-stable implicit one-step methods," *BIT*, v. 12, 1972, pp. 252–266.

17. J. WILLIAMS & F. DE HOOG, "A class of A-stable advanced multistep methods," *Math. Comp.*, v. 28, 1974, pp. 163–177.