Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 5 (3): 411-427, 2005
ISSN 1812-5654

Implementation of s-stage Implicit Runge-Kutta Method of Order 2s for Second Order Initial Value Problems

M. Sikander Hayat Khiyal and Khalid Rashid
Department of Computer Science, Faculty of Applied Sciences,
International Islamic University, Sector H-10, Islamabad, Pakistan

Abstract: This study is concerned with the approximate solution of the special second order initial value problem. For this purpose we use the s-stage (s = 2,3,4) Implicit Runge-Kutta methods of order 2s. Fourth order method is used by two iterative technique, perfect square iterative scheme and Cooper and Butcher iterative scheme while sixth and eighth order methods are solved by using Cooper and Butcher iterative scheme. We have converted the scheme in such a way that only two function evaluations are required per iteration for fourth order method, three for sixth order method and four for eighth order method. Finally we present the numerical results.

Key words: Implicit Runge-Kutta method, iterative technique, initial value problem

INTRODUCTION

We are concerned with the approximate numerical integration of the special second order initial value problem

$$y'' = f(t, y),\ y(a) = \eta,\ y'(a) = \eta' \tag{1}$$

Where, y, f, η and η' are vectors of dimension m.

A second order system can be solved by converting it to a first order system and then applying a numerical method for first order system. If we let $x_1(t) = y(t)$, $x_2(t) = y'(t)$, $x_1(a) = \eta$, $x_2(a) = \eta'$ and introduce the notation $x = [x_1, x_2]^T$, $g = [x_2, f]^T$, $\mu = [\eta, \eta']^T$, we have a first order system

$$x' = g(t, x),\ x(a) = \mu \tag{2}$$

The s-stage Runge-Kutta method is expressed in the form

$$x_{n+1} = x_n + h\sum_{r=1}^{s} d_r g(t_n + ha_r, X_r) \tag{3}$$

where, the d_r's are constants and the terms X_r on the right hand side of (3) are given by

$$X_r = x_n + h\sum_{r=1}^{s} b_{ru} g(t_n + ha_u, X_u),\ u = 1, 2, \ldots, s \tag{4}$$

$$a_r = \sum_{u=1}^{s} b_{ru},\ u = 1, 2, \ldots, s \tag{5}$$

We consider the s-stage (s = 2, 3, 4) implicit Runge-Kutta methods of order 2s proposed by Butcher[1]. These methods are A-stable[2] and are given in Tableau form by

$$\mathbf{s=2:}\quad \begin{array}{c|cc} \frac{3-\sqrt{3}}{6} & \frac{1}{4} & \frac{3-2\sqrt{3}}{12} \\ \frac{3+\sqrt{3}}{6} & \frac{3+2\sqrt{3}}{12} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \tag{6}$$

$$\mathbf{s=3:}\quad \begin{array}{c|ccc} \frac{5-\sqrt{15}}{10} & \frac{5}{36} & \frac{10-3\sqrt{15}}{45} & \frac{25-6\sqrt{15}}{180} \\ \frac{1}{2} & \frac{10+3\sqrt{15}}{72} & \frac{2}{9} & \frac{10-3\sqrt{15}}{45} \\ \frac{5+\sqrt{15}}{10} & \frac{25+6\sqrt{15}}{180} & \frac{10+3\sqrt{15}}{45} & \frac{5}{36} \\ \hline & \frac{5}{18} & \frac{4}{9} & \frac{5}{18} \end{array} \tag{7}$$

Corresponding Author: M. Sikander Hayat Khiyal, Department of Computer Science, Faculty of Applied Sciences, International Islamic University, Sector H-10, Islamabad, Pakistan
E-mail: sikandar_hayat_khiyal@yahoo.com

$$s=4:\quad \begin{array}{c|cccc} a_1 & b_{11} & b_{12} & b_{13} & b_{14} \\ a_2 & b_{21} & b_{22} & b_{23} & b_{24} \\ a_3 & b_{31} & b_{32} & b_{33} & b_{34} \\ a_4 & b_{41} & b_{42} & b_{43} & b_{44} \\ \hline & d_1 & d_2 & d_3 & d_4 \end{array} \tag{8}$$

Where: $a_1 = \frac{1}{2}-\eta,\ a_2 = \frac{1}{2}-\eta',\ a_3 = \frac{1}{2}+\eta',\ a_4 = \frac{1}{2}+\eta,$

$b_{11} = \eta,\ b_{12} = \eta_1' - \eta_3 + \eta_4',\ b_{13} = \eta_1' - \eta_3 - \eta_4',\ b_{14} = \eta_1 - \eta_5,$

$b_{21} = \eta_1 - \eta_3' + \eta_4,\ b_{22} = \eta_1',\ b_{23} = \eta_1' - \eta_5',\ b_{24} = \eta_1 - \eta_3' - \eta_4,$

$b_{31} = \eta_1 + \eta_3' + \eta_4,\ b_{32} = \eta_1' + \eta_5',\ b_{33} = \eta_1',\ b_{34} = \eta_1 + \eta_3' - \eta_4,$

$b_{41} = \eta_1 + \eta_5,\ b_{42} = \eta_1' + \eta_3 + \eta_4',\ b_{43} = \eta_1' + \eta_3 - \eta_4',\ b_{44} = \eta_1,$

$d_1 = 2\eta_1,\ d_2 = 2\eta_1',\ d_3 = 2\eta_1',\ d_4 = 2\eta_1,$

$$\eta_1 = \frac{18-\sqrt{30}}{144},\ \eta_2 = \frac{1}{2}\left(\frac{15+2\sqrt{30}}{35}\right)^{\frac{1}{2}},$$

$$\eta_3 = \eta_2\left(\frac{4+\sqrt{30}}{24}\right),\ \eta_4 = \eta_2\left(\frac{8+5\sqrt{30}}{168}\right),$$

$$\eta_5 = \eta_2 - 2\eta_3,\ \eta_1' = \frac{18+\sqrt{30}}{144},\ \eta_2' = \frac{1}{2}\left(\frac{15-2\sqrt{30}}{35}\right)^{\frac{1}{2}},$$

$$\eta_3' = \eta_2'\left(\frac{4-\sqrt{30}}{24}\right),\ \eta_4' = \eta_2'\left(\frac{8-5\sqrt{30}}{168}\right),$$

$$\eta_5' = \eta_2' - 2\eta_3'.$$

Implicit Runge-Kutta methods are generally expensive to implement because there is a system of nonlinear equations to solve at each step whenever F(X) is nonlinear. For example, if we have a system of n ODE′s and the modified Newton method is used to solve the nonlinear equations which result, then this involves the solution of a system of sn linear equations at each step of the iteration. The local error estimate may be obtained from the equation,

$$Le_{n+1} = y_{n+1}^{(Q)} - y_{n+1}^{(0)}, \tag{9}$$

where, $y_{n+1}^{(0)}$ is the predicted value of y_{n+1} and $y_{n+1}^{(Q)}$ is the corrected value obtained on convergence of the iteration scheme for the implicit Runge-Kutta method. If this local error is acceptably small, we may then proceed. If the error test is not satisfied then we reduce the stepsize by applying the stepsize strategy and start again at the beginning. This approach for fourth order methods was also proposed by Thomas[3].

To apply the implicit Runge-Kutta method to the second order system (1), first we must convert the second order system to an equivalent first order system. However, to obtain an efficient algorithm, we exploit the structure of the resulting first order system.

On applying the implicit Runge-Kutta method to the first order system, there results a system of nonlinear algebraic equations. For the fourth order two-stage method, the resulting iteration matrix has the form $\left(I+\alpha h^2 J+\beta h^4 J^2\right)$ and is of order s. We consider an iteration scheme in which this iteration matrix is approximated by a perfect square, so that at most one matrix of order n must be factorized at each step. Unfortunately, so far it has not proved possible to adopt a similar approach for the s-stage implicit Runge-Kutta method of order 2s (s = 3 and 4). Also, the perfect square approach for the fourth order method can prove very unreliable[4]. Instead, we use another iteration scheme introduced by Cooper and Butcher[5]. We consider in detail the application of this scheme to the s-stage implicit Runge-Kutta methods of order 2s, s = 2,3,4.

PERFECT SQUARE ITERATION SCHEME

Fourth order A-stable implicit Runge-Kutta method: The fourth order two-stage implicit Runge-Kutta method for first order systems is given by (6). We discussed how, for computational convenience, we can write the non-autonomous system (2) in the form of an autonomous system x′ = F(x). Then the fourth order method (6) has the form:

$$x_{n+1} = x_n + \frac{h}{2}[F(X_1) + F(X_2)]$$

$$X_1 = x_n + \frac{h}{4}F(X_1) + \frac{3-2\sqrt{3}}{12}hF(X_2). \tag{10}$$

$$X_2 = x_n + \frac{3+2\sqrt{3}}{12}hF(X_1) + \frac{h}{4}F(X_2). \tag{11}$$

Eliminating $F(X_2)$ from (10) and (11) gives:

$$X_2 = -(3+2\sqrt{3})\,X_1 + (4+2\sqrt{3})x_n + \frac{1}{3}(3+2\sqrt{3})hF(X_1) \tag{12}$$

Let,

$$G(X_1) = X_1 - x_n - \frac{h}{4}F(X_1) - \frac{3-2\sqrt{3}}{12}hF(X_2). \tag{13}$$

Applying the modified Newton iteration gives

$$G'(X_1^{(p-1)})\left(X_1^{(p)} - X_1^{(p-1)}\right) = -G\left(X_1^{(p-1)}\right). \quad (14)$$

where, from equations (12) and (13),

$$G'(X_1) = I - \frac{h}{2}J + \frac{h^2}{12}J^2$$

and J is an approximation for the Jacobian matrix of F with respect to x. Thus the modified Newton iteration yields

$$\left(I - \frac{h}{2}J + \frac{h^2}{12}J^2\right)\left(X_1^{(p)} - X_1^{(p-1)}\right) = -X_1^{(p-1)} + x_n + \frac{h}{4}F\left(X_1^{(p-1)}\right) + \frac{3-2\sqrt{3}}{12}hF\left(X_2^{(p-1)}\right)$$

and

$$X_2^{(p-1)} = x_n + \left(3+2\sqrt{3}\right)\left[x_n - X_1^{(p-1)} + \frac{h}{3}F\left(X_1^{(p-1)}\right)\right].$$

Since matrix products are expensive, especially for large systems and any sparsity in J will be lost in J^2 and also any ill-conditioning in J will be magnified in its powers, we wish to avoid the calculation of J^2. To achieve this we may write the iteration matrix as a product of two linear (complex) factors, giving

$$(I - hr_cJ)(I - h\bar{r}_cJ)\delta = e^{(p-1)}$$

where,

$$\delta = X_1^p - X_1^{(p-1)}, e^{(p-1)} = -G(X_1^{(p-1)}), r_c = \frac{3+i\sqrt{3}}{12}$$ and $\bar{r}_c$ is

the complex conjugate of r_c.

Gladwell and Thomas[4] show how to solve this system efficiently using complex arithmetic. We do not consider this approach further, but instead we consider an alternative approach, also discussed by Gladwell and Thomas[4], in which the iteration matrix is approximated by a perfect square:

$$(I - hr_RJ)^2\delta = e^{(p-1)} \quad (15)$$

where, r_R is a real number. (The choice of r_R is discussed later in this section.) This can be solved as

$$(I - hr_RJ)Z = e^{(p-1)} \quad (16)$$

$$(I - hr_RJ)\delta = Z \quad (17)$$

Now for the second order system (1), $F(x) = [y', f(y)]^T$ and $x = [y, y']^T$. Also, suppose that $X_1 = [w_1, w_2]^T$, $e = [e_1, e_2]^T$, $\delta = [\delta_1, \delta_2]^T$, $Z = [Z_1, Z_2]^T$, so that (16) can be written in the form

$$\begin{bmatrix} I & -hr_RI \\ -hr_RJ & I \end{bmatrix}\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} = \begin{bmatrix} e_1^{(p-1)} \\ e_2^{(p-1)} \end{bmatrix} \quad (18)$$

while (17) gives

$$\begin{bmatrix} I & -hr_RI \\ -hr_RJ & I \end{bmatrix}\begin{bmatrix} \delta_1 \\ \delta_2 \end{bmatrix} = \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} \quad (19)$$

Premultiplying (18) by $\begin{bmatrix} I & hr_RJ \\ 0 & I \end{bmatrix}$ gives

$$\begin{bmatrix} I-h^2r_R^2J & 0 \\ -hr_RJ & I \end{bmatrix}\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} = \begin{bmatrix} e_1^{(p-1)} + hr_Re_2^{(p-1)} \\ e_2^{(p-1)} \end{bmatrix}$$

so that

$$(I - h^2r_R^2J)Z_1 = e_1^{(p-1)} + hr_Re_2^{(p-1)} \quad (20)$$

and

$$Z_2 = hr_RJZ_1 + e_2^{(p-1)}$$

although, as we shall see, it is not necessary to compute Z_2 explicitly.

Similarly, premultiplying (19) by $\begin{bmatrix} I & hr_RI \\ 0 & I \end{bmatrix}$ gives

$$\begin{bmatrix} I-h^2r_R^2J & 0 \\ -hr_RJ & I \end{bmatrix}\begin{bmatrix} \delta_1 \\ \delta_2 \end{bmatrix} = \begin{bmatrix} Z_1 + hr_RZ_2 \\ Z_2 \end{bmatrix}$$

and so

$$(I - h^2r_R^2J)\delta_1 = Z_1 + hr_RZ_2.$$

From the first of equations (18), we have

$$hr_RZ_2 = Z_1 - e_1^{(p-1)}$$

Thus, after computing Z_1 from (20), we must solve

$$(I - h^2r_R^2J)\delta_1 = 2Z_1 - e_1^{(p-1)}.$$

Finally from (19), we have

$$\delta_2 = \frac{1}{hr_R}(\delta_1 - Z_1).$$

Thus, one step of the iteration using the perfect square approximation is

$$\text{Solve } \left(I-h^2r_R^2J\right)Z_1=e_1^{(p-1)}+hr_Re_2^{(p-1)}$$

$$\text{Solve } \left(I-h^2r_R^2J\right)\delta_1=2Z_1-e_1^{(p-1)}$$

$$\text{Set } \delta_2 = \frac{1}{hr_R}(\delta_1 - Z_1).$$

We have to solve two systems of n linear equations at each step of the iteration. However, the matrix is the same for both systems so we do not need to perform more than one matrix factorization per step.

Now we need explicit forms for e_1 and e_2. Note that

$$\begin{bmatrix} e_1^{(p-1)} \\ e_2^{(p-1)} \end{bmatrix} = -G(X_1^{(p-1)}) = -X_1^{(p-1)}+x_n + \frac{h}{4}F(X_1^{(p-1)}) + \frac{3-2\sqrt{3}}{12}hF(X_2^{(p-1)}). \quad (21)$$

where,

$$X_2^{(p-1)} = x_n+(3+2\sqrt{3})\left[x_n - X_1^{(p-1)}+\frac{h}{3}F(X_1^{(p-1)})\right].$$

On defining, $X_1 = \left[w_1,w_2\right]^T$ and $F(X_1) = \left[w_2,f(w_1)\right]^T$ we find that

$$X_2^{(p-1)} = \begin{bmatrix} y_n+(3+2\sqrt{3})\left[y_n-w_1^{(p-1)}+\frac{h}{3}w_2^{(p-1)}\right] \\ y'_n+(3+2\sqrt{3})\left[y'_n-w_2^{(p-1)}+\frac{h}{3}f(w_1^{(p-1)})\right] \end{bmatrix}$$

and

$$F(X_2^{(p-1)}) = \begin{bmatrix} y'_n+(3+2\sqrt{3})\left[y'_n-w_2^{(p-1)}+\frac{h}{3}f(w_1^{(p-1)})\right] \\ f\left(y_n+(3+2\sqrt{3})\left[y_n-w_1^{(p-1)}+\frac{h}{3}w_2^{(p-1)}\right]\right) \end{bmatrix} \quad (22)$$

Now from (21), after a little manipulation, we have

$$e_1^{(p-1)} = y_n-w_1^{(p-1)}+\frac{h}{4}w_2^{(p-1)}+\frac{3+2\sqrt{3}}{12}\times h\left[y'_n+(3+2\sqrt{3})(y'_n-w_2^{(p-1)}+\frac{h}{3}f(w_1^{(p-1)}))\right] \quad (23)$$

$$e_2^{(p-1)} = y'_n-w_2^{(p-1)}+\frac{h}{4}f(w_1^{(p-1)})+\frac{3-2\sqrt{3}}{12}\times hf\left(y_n+\left(3+2\sqrt{3}\right)\left[y_n-w_1^{(p-1)}+\frac{h}{3}w_2^{(p-1)}\right]\right) \quad (24)$$

Finally, consider

$$x_{n+1} = x_n+\frac{h}{2}\left[F(X_1)+F(X_2)\right].$$

If $X_2 = \left[v_1,v_2\right]^T$ then $F(X_2) = \left[v_2,f(v_1)\right]^T$. Hence

$$y_{n+1} = y_n+\frac{h}{2}+(w_2+v_2) \quad (25)$$

and

$$y'_{n+1} = y'_n+\frac{h}{2}\left[f(w_1)+f(v_1)\right]. \quad (26)$$

To avoid the function evaluations $f(w_1)$ and $f(v_1)$, we proceed as follows. From equation (12), we have,

$$v_1 = \left(4+2\sqrt{3}\right)y_n-\left(3+2\sqrt{3}\right)w_1+\frac{3+2\sqrt{3}}{3}hw_2$$

and

$$v_2 = \left(4+2\sqrt{3}\right)y'_n-\left(3+2\sqrt{3}\right)w_2+\frac{3+2\sqrt{3}}{3}hf(w_1)$$

Thus

$$v_2 = \left(4+2\sqrt{3}\right)y'_n-\left(3+2\sqrt{3}\right)w_2+\frac{3+2\sqrt{3}}{3}hf(w_1) \quad (27)$$

Now from equation (10), we have

$$X_1 = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} y_n+\frac{h}{4}w_2+\frac{3-2\sqrt{3}}{12}hv_2 \\ y'_n+\frac{h}{4}f(w_1)+\frac{3-2\sqrt{3}}{12}hf(v_1) \end{bmatrix}.$$

Hence

$$w_1 = y_n+\frac{h}{4}w_2+\frac{3-2\sqrt{3}}{12}hv_2 \quad (28)$$

and

$$w_2 = y'_n+\frac{h}{4}f(w_1)+\frac{3-2\sqrt{3}}{12}hf(v_1) \quad (29)$$

By substituting the value of v_2 from equation (27) into (28), after a little manipulation, we have

$$f(w_1) = \frac{12}{h^2}y_n - w_1 + \frac{h}{2}w_2 - \frac{\sqrt{3}}{6}hy'_n \quad (30)$$

Also, by substituting the value of v_2 into equation (29), we have

$$f(v_1) = 4\left(3+2\sqrt{3}\right)\left[-w_2 + y'_n + \frac{h}{4}f(w_1)\right] \quad (31)$$

Substituting the value of v_2 from equation (27) into equation (25) gives, after a little manipulation,

$$y_{n+1} = \left(7+4\sqrt{3}\right)y_n + \left(2+\sqrt{3}\right)hw_2 - 2\left(3+2\sqrt{3}\right)w_1. \quad (32)$$

Finally, on substituting for $f(w_1)$ and $f(v_1)$ from equations (30) and (31) into (26), we obtain

$$hy'_{n+1} = hy'_n + 2\left(3+\sqrt{3}\right)hw_2 + 12\left(2+\sqrt{3}\right)y_n - 12\left(2+\sqrt{3}\right)w_1. \quad (33)$$

Thus, on convergence of the iteration (after Q iterations, say), we set

$$y_{n+1} = \left(7+4\sqrt{3}\right)y_n + \left(2+\sqrt{3}\right)hw_2^{(Q)} - 2\left(3+2\sqrt{3}\right)w_1^{(Q)}.$$

$$hy'_{n+1} = hy'_n + 2\left(3+\sqrt{3}\right)hw_2^{(Q)} + 12\left(2+\sqrt{3}\right)y_n - 12\left(2+\sqrt{3}\right)w_1^{(Q)}.$$

As we have seen, we approximate the iteration matrix by a perfect square of the form $(I-r_R h_J)^2$, where r_R is a real number. The choice $r_R = \frac{\sqrt{3}}{6}$ gives a discrepancy only in the linear term of the approximation while the choice $R_r = 1/4$ gives a discrepancy only in the quadratic term. Gladwell and Thomas[4] show that the latter choice gives desirable convergence properties for h small as may be expected. Another possibility considered by Thomas[3] is to take $r_R = [-\beta_1(u_r+u_i)]^{1/4}$, in which case the same iteration matrix can be used on transferring to the fourth order direct hybrid method, provided it is not necessary to change the stepsize. However, Thomas found that in practice, this choice requires the use of a smaller initial stepsize and hence more function evaluations are needed overall. For this reason, we will not consider this choice further. Indeed, in providing our test results, we only use the choice $r_R = 1/4$.

To obtain a local error estimate we use the approach in which the estimate Le_{n+1} of the local error in the predicted value $y^{(0)}_{n+1}$ is obtained from equation (9)

$$Le_{n+1} = y_{n+1}^{(Q)} - y_{n+1}^{(0)}$$

Since the Runge-Kutta method is fourth order accurate, we require $y^{(0)}_{n+1}$ to be a third order approximation to $y(t_{n+1})$ so that the estimate will be asymptotically correct.

The predictor we propose to use for the algorithm is based on Hermite interpolation. The Hermite interpolant of degree three to (t_n, y_n, y'_n), $(t_{n-1}, y_{n-1}, y'_{n-1})$, $Q_{3,n}(t)$ say, may be evaluated at $t = t_{n+1}$, to give

$$y_{n=1}^{(0)} = -4y_n + 5y_{n-1} + 4hy'_n + 2hy'_{n-1} \quad (34)$$

(This formula is derived in Appendix 4 by Khiyal[6]) The use of (34) implies that, initially, we must carry out two steps before attempting to form the error estimate. To start the iteration in the algorithm, we must predict values of w_1 and hw_2. We use the predictor proposed by Gladwell and Thomas[4] based on the Hermite interpolant. We take

$$w_1^{(0)} = Q_{3,n}\left(t_n + \frac{3-\sqrt{3}}{6}h\right) \quad \text{and} \quad w_2^{(0)} = Q'_{3,n}\left(t_n + \frac{3-\sqrt{3}}{6}h\right)$$

(These values can be obtained from equations (A4.9) and (A4.10) of Appendix 4, respectively of Khiyal[6]. They cannot be used on the first step and so, on this step, we use:

$$w_1^{(0)} = y_0 + \frac{3-\sqrt{3}}{6}hy'_0,\ hw_2^{(0)} = hy'_0$$

After carrying out two steps with the implicit Runge-Kutta method, we form an estimate of the local error. If the local error test is not satisfied, we reduce the stepsize and start again at the beginning. If the error test is satisfied then the estimated value of the stepsize, $\bar{h}$, obtained from

$$\bar{h} = h\left[\frac{TOL}{2\|Le_{n+1}\|}\right]^{\frac{1}{R+1}} \quad (35)$$

can be used for the next step. In equation (35), R is order of the predeictor used. Suppose that h is the current stepsize. If $\frac{\bar{h}}{h} < \rho_2$, we continue with the current stepsize h and if $\rho_2 \le \frac{\bar{h}}{h} \le \rho_3$, we increase the stepsize to $\bar{h}$. However, if $\frac{\bar{h}}{h} < \rho_3$ we reject the estimate and decrease the stepsize to $\bar{h}$ but not less than $\rho_1 h$. This strategy is just the same as discussed by Khiyal[6]. Note that the factors ρ_1, ρ_2 and ρ_3 are taken to be 0.2, 2.0 and 5.0, respectively.

COOPER AND BUTCHER ITERATION SCHEME

s-stage 2s-order implicit Runge-Kutta methods: We have obtained a perfect square iteration scheme for the two-step fourth order A-stable Runge-Kutta method applied to second order systems. However, we were not able to find a corresponding perfect cube iteration scheme for the three-stage sixth order implicit Runge-Kutta method. Moreover, the perfect square iteration scheme for the fourth order method is not always reliable. For these reasons, in this section, we consider an iteration scheme proposed by Cooper and Butcher[5] which sacrifices the superlinear convergence of the modified Newton method for reduced linear algebra costs. This iteration scheme requires the solution of s systems of n linear equations at each step of the iteration and the coefficient matrix is the same for each system.

We have shown that for computational convenience the nonautonomous system (2) can be written in the form of an autonomous system and then we write the Runge-Kutta method in autonomous form. Suppose the first order autonomous initial value problem is

$$x' = F(x),\ x(0) = x_0 \tag{36}$$

Then the s-stage Runge-Kutta method is given by

$$x_{n+1} = x_n + h\sum_{R=1}^{s} d_R F(X_R) \tag{37}$$

where, $X_1, X_2, ..., X_s$ satisfy the sR equations

$$X_R = x_n + h\sum_{j=1}^{s} a_{Rj} F(X_j)\ R=1,2,3,...s \tag{38}$$

We may simplify our notation by using direct sums and direct products. Let $\bar{X} = X_1 \oplus X_2 \oplus ... \oplus X_s$ and $\bar{x} = x_n \oplus x_n \oplus x_n \oplus ... \oplus x_n$ be column vectors and let $\bar{F}(\bar{X}) = F(X_1) \oplus F(X_2) \oplus ... \oplus F(X_s)$. Then equation (38) may be represented by

$$\bar{X} = \bar{x} + h(A \otimes I)\bar{F}(\bar{X}) \tag{39}$$

Where, $(A \otimes I)$ is the direct product of A with the $s \times s$ identity matrix and, in general,

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \cdots & a_{1s}B \\ a_{21}B & a_{22}B & \cdots & a_{2s}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{s1}B & a_{s2}B & \cdots & a_{ss}B \end{bmatrix}$$

Where $A = (a_{ij})$. The equation (39) may be solved by the modified Newton method but because the Jacobian of F is expensive to evaluate, it is computed only occasionally. Suppose that $\tilde{J}$ is an approximation for this Jacobian. Then the modified Newton iteration scheme is

$$\left(I - h(A \otimes \tilde{J})\right)\bar{Z}^{(p)} = \bar{G}^{(p-1)}$$
$$\bar{X} = \bar{X}^{(p-1)} + \bar{Z}^{(p)}$$

where:

$\bar{G}^{(p-1)} = \bar{x} + h(A \otimes I)\bar{F}(\bar{X}^{(p-1)}) - \bar{X}^{(p-1)}$, p = 1,2,3,... Note that in practice the Jacobian is kept constant for several time steps. At each iteration, we need to solve a system of linear equations.

Let B and S be real invertible matrices and let L be a strictly lower triangular matrix and r a real constant. Cooper and Butcher[5] propose the following iteration scheme for solving (39)

$$\left(I - rhI \otimes \tilde{J}\right)\bar{Z}^{(p)} = (L \otimes I)\bar{Z}^{(p)} + \left(BS^{-1} \otimes I\right)\bar{G}^{(p-1)}\ p = 1, 2, 3.... \tag{40}$$

where,

$$(S \otimes I)\bar{Z}^{(p)} = \bar{X}^{(p)} - \bar{X}^{(p-1)} \tag{41}$$

$\bar{G}^{(p)}$ is the residual,

$$\bar{G}^{(p)} = \bar{x} + h(A \otimes I)\bar{F}(\bar{X}^{(p)}) - \bar{X}^{(p)},\quad p = 1,2,3.... \tag{42}$$

and $\tilde{J}$ is an approximation for the Jacobian matrix of F with respect to x. Note that $\left(I - rhI \otimes \tilde{J}\right)$ is block diagonal and that $(L \otimes I)$ is strictly lower triangular. This means that each step of the iteration requires the solution of s sets of n linear equations and when the solution of the jth set of equations is being found, the solutions of the previous j-1 sets of equations may be used. For large s, the linear algebra costs for the Cooper and Butcher iteration scheme are much less per iteration than for the modified Newton method, for which a system of sn linear equations has to be solved at each step of the iteration. There are three matrices L, B and S and the parameter r which may be chosen in many ways. w is the relaxation parameter. Cooper and Butcher[5] choose w to give the best rate of convergence for the iteration scheme when the method is applied to the scalar linear test problem $x' = cx$, where c is real and negative.

For each s-stage method of order 2s there is a matrix S such that:

$$S^{-1}AS = \bar{A} = \bar{A}_1 \oplus \bar{A}_2 \oplus \ldots \oplus \bar{A}_M$$

a real block diagonal matrix. Here M is an integer defined by

$$M = \begin{cases} \frac{s}{2}, & \text{if s is even} \\ \frac{s+1}{2} & \text{if s is odd} \end{cases}$$

The submatrices $\{\bar{A}\}$ are chosen to have the form

$$\bar{A}_i = \begin{bmatrix} \bar{a}_i & \bar{a}_i - \bar{b}_i \\ \bar{a}_i + \bar{b}_i & \bar{a}_i \end{bmatrix}, i = 1,2,3,..., M$$

except that, when s is odd, $\bar{A}_M = [\bar{a}_M]$. Also $L = L_1 \oplus L_2 \oplus \ldots \oplus L_M$ and $B = B_1 \oplus B_2 \oplus \ldots \oplus B_M$ have the same block diagonal form. When s is odd, $L_M = [0]$ and $B_M = [\beta]$.

In introduction the s-stage implicit Runge-kutta methods of order 2s (s = 2,3,4) are given, in tableau form, by (6), (7) and (8), respectively. We take the matrices L and B to be the same as those given by Cooper and Butcher[5]. That is, we take

$$L_i = \begin{bmatrix} 0 & 0 \\ -w_i \frac{r^2 - 2\bar{a}_i r + \bar{b}_i}{(\bar{a}_i - \bar{b}_i) r} & 0 \end{bmatrix}, \quad i = 1, 2,..,M$$

$$B_i = \begin{bmatrix} w_i & w_i \frac{r - \bar{a}_i}{\bar{a}_i + \bar{b}_i} \\ w_i \frac{r - \bar{a}_i}{\bar{a}_i - \bar{b}_i} - w_i^2 \frac{\bar{a}_i r - \bar{b}_i^2}{r(\bar{a}_i - \bar{b}_i)} & w_i + w_i^2 \frac{(\bar{a}_i^2 r - \bar{b}_i^2)(\bar{a}_i - r)}{r(\bar{a}_i^2 - \bar{b}_i^2)} \end{bmatrix},$$

i = 1, 2,...., M

When s = 2 the method of order 2s has the matrix of coefficients

$$A = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} - \frac{\sqrt{3}}{6} \\ \frac{1}{4} + \frac{\sqrt{3}}{6} & \frac{1}{4} \end{bmatrix}$$

We may take $S = I$ as suggested by Cooper and Butcher[5]. Then $A = \bar{A}$. Following Cooper and Butcher, we take $r = \frac{\sqrt{3}}{6}$. The implementation of these methods is discussed later.

When s = 3 the method of order 2s has the matrix of coefficients:

$$A = \begin{bmatrix} \frac{5}{36} & \frac{2}{9} - \frac{\sqrt{15}}{15} & \frac{5}{36} - \frac{\sqrt{15}}{30} \\ \frac{5}{36} + \frac{\sqrt{15}}{24} & \frac{2}{9} & \frac{5}{36} - \frac{\sqrt{15}}{24} \\ \frac{5}{36} + \frac{\sqrt{15}}{30} & \frac{2}{9} + \frac{\sqrt{15}}{15} & \frac{5}{36} \end{bmatrix}$$

The matrix S is chosen so that

$$\bar{A} = S^{-1}AS = \begin{bmatrix} \bar{a}_1 & \bar{a}_1 - \bar{b}_1 & 0 \\ \bar{a}_1 + \bar{b}_1 & \bar{a}_1 & 0 \\ 0 & 0 & \bar{a}_2 \end{bmatrix} = \bar{A}_1 \oplus \bar{A}_2$$

The elements of $\bar{A}$ may be obtained by noting that $\det(rI - \bar{A}) = \det(rI - A)$ where $\det(rI - \bar{A}) = r^3 - (2\bar{a}_1 + \bar{a}_2) r^2 + (2\bar{a}_1\bar{a}_2 + \bar{b}_1^2) r - \bar{a}_2\bar{b}_1^2$

and

$$\det(rI - A) = r^3 - \frac{1}{2}r^2 + \frac{1}{10}r - \frac{1}{120}$$

Then we have $2\bar{a}_1 + \bar{a}_2 = \frac{1}{2}$, $2\bar{a}_1\bar{a}_2 + \bar{b}_1^2 = \frac{1}{10}$ and $\bar{a}_2\bar{b}_1^2 = \frac{1}{120}$. By numerical evaluation these give

$$\bar{a}_1 \approx 0.142342788, \bar{b}_1 \approx 0.196731007, \bar{a}_2 \approx 0.215314423$$

The matrix S may be obtained by noting that its columns are a linearly independent set of eigenvectors of the matrix $(\bar{a}_1 I - A)^2$. These eigenvectors are obtained by numerical calculation. Since s = 3 is odd, we have $L_2 = [0]$ and $B_2 = [\beta]$ where, $\beta = \frac{a_1 + r}{a_1 + \bar{a}_2}$ and $a_1 = \frac{\bar{a}_1 r - \bar{b}_1^2}{\bar{a}_1 - r}$ These values are given by Cooper and Butcher[5]. Again we take $r = \bar{b}_1$ when we consider the implementation of this method later in this section.

When s = 4 the method of order 2s has the matrix of coefficients:

$$A = \begin{bmatrix} \eta_1 & \eta_1' - \eta_3 + \eta_4' & \eta_1' - \eta_3 - \eta_4' & \eta_1 - \eta_5 \\ \eta_1 - \eta_3' + \eta_4 & \eta_1' & \eta_1' - \eta_5' & \eta_1 - \eta_3' - \eta_4 \\ \eta_1 + \eta_3' + \eta_4 & \eta_1' + \eta_5' & \eta_1' & \eta_1 + \eta_3' + \eta_4 \\ \eta_1 - \eta_5 & \eta_1' + \eta_3 + \eta_4' & \eta_1' + \eta_3 - \eta_4' & \eta_1 \end{bmatrix}$$

where:

$$\eta_1 = \frac{18-\sqrt{30}}{144}, \eta_2 = \frac{1}{2}\left(\frac{15+2\sqrt{30}}{35}\right)^{\frac{1}{2}},$$

$$\eta_3 = \eta_2\left(\frac{4+\sqrt{30}}{24}\right), \eta_4 = \eta_2\left(\frac{8+5\sqrt{30}}{168}\right),$$

$$\eta_5 = \eta_2 - 2\eta_3, \eta_1' = \frac{18+\sqrt{30}}{144}, \eta_2' = \frac{1}{2}\left(\frac{15-2\sqrt{30}}{35}\right)^{\frac{1}{2}},$$

$$\eta_3' = \eta_2'\left(\frac{4-\sqrt{30}}{24}\right), \eta_4' = \eta_2'\left(\frac{8-5\sqrt{30}}{168}\right),$$

$\eta_5' = \eta_2' - 2\eta_3'$. The matrix S is chosen so that

$$\bar{A} = S^{-1}AS = \begin{bmatrix} \bar{a}_1 & \bar{a}_1\text{-}\bar{b}_1 & 0 & 0 \\ \bar{a}_1 + \bar{b}_1 & \bar{a}_1 & 0 & 0 \\ 0 & 0 & \bar{a}_2 & \bar{a}_2\text{-}\bar{b}_2 \\ 0 & 0 & \bar{a}_2 + \bar{b}_2 & \bar{a}_2 \end{bmatrix}$$

The elements of the matrix $\bar{A}$ may be obtained by noting that $\det(rI-\bar{A}) = \det(rI-A)$ where:

$$\det(rI-\bar{A}) = r^4 - 2(\bar{a}_1+\bar{a}_2)r^3 + \left(\bar{b}_1^2 + 4\bar{a}_1\bar{a}_2 + \bar{b}_2^2\right)r^2 - 2(\bar{a}_1\bar{b}_2^2 + \bar{a}_2\bar{b}_1^2)r + \bar{b}_1^2\bar{b}_2^2$$

and

$$\det(rI-A) = r^4 - \frac{1}{2}r^3 + \frac{3}{28}r^2 - \frac{1}{84}r + \frac{1}{1680}$$

Thus we have $2(\bar{a}_1+\bar{a}_2) = 1/2$, $\left(\bar{b}_1^2 + 4\bar{a}_1\bar{a}_2 + \bar{b}_2^2\right) = \frac{3}{28}$, $2(\bar{a}_1\bar{b}_2^2 + \bar{a}_2\bar{b}_1^2) = 1/84$ and $\bar{b}_1^2\bar{b}_2^2 = \frac{1}{1680}$. By numerical evaluation, we obtain $\bar{a}_1 \approx 0.0915662403, \bar{a}_2 \approx 0.158433760$, $\bar{b}_1 \approx 0.147520224, \bar{b}_2 \approx 0.165384116$.

Then the matrix S is obtained by noting that its first two columns are a linearly independent set of eigenvectors of $(\bar{a}_1 I-A)^2$ and the third and the fourth columns are a linearly independent set of eigenvectors of $(\bar{a}_2 I-A)^2$. Again these eigenvectors are obtained by numerical calculation. As suggested by Cooper and Butcher, we take $r = \bar{b}_1$ to obtain a better rate of convergence.

Implementation aspects: Now we consider the implementation of the s-stage methods with this iteration scheme for second order systems of the form (1). (When s = 2 the implementation of the Cooper and Butcher iteration scheme was discussed by Gladwell and Thomas[4], who gave an algorithm.). Note that in (40)

$$\left[I\text{-}rh\tilde{I} \otimes \tilde{J}\right] = \begin{bmatrix} I\text{-}rh\tilde{J} & 0 & \cdots & 0 \\ 0 & I\text{-}rh\tilde{J} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & I\text{-}rh\tilde{J} \end{bmatrix}$$

for the first order system (36). Then from (40), we have

$$\begin{bmatrix} I\text{-}rh\tilde{J} & 0 & \cdots & 0 \\ 0 & I\text{-}rh\tilde{J} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & I\text{-}rh\tilde{J} \end{bmatrix} \bar{Z}^{(p)} = (L \otimes I)\bar{Z}^{(p)} + (BS^{-1} \otimes I)\bar{G}^{(p-1)} \quad (43)$$

Let $\bar{Z} = \left[\bar{Z}_1^T, \bar{Z}_2^T, \ldots, \bar{Z}_s^T\right]^T$, $\bar{G} = \left[\bar{G}_1^T, \bar{G}_2^T, \ldots, \bar{G}_s^T\right]^T$ where

$\bar{Z}_1 = \left[Z_1, Z_1'\right]^T$ and $\bar{G}_1 = \left[G, G_1'\right]^T$ and

$$BS^{-1} = \bar{B} = \begin{bmatrix} \bar{b}_{11} & \bar{b}_{12} & \cdots & \bar{b}_{1s} \\ \bar{b}_{21} & \bar{b}_{22} & \cdots & \bar{b}_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{b}_{s1} & \bar{b}_{s2} & \cdots & \bar{b}_{ss} \end{bmatrix}$$

Then equation (43) becomes

$$\begin{bmatrix} \left(I\text{-}rh\tilde{J}\right)\bar{Z}_1^{(p)} \\ \left(I\text{-}rh\tilde{J}\right)\bar{Z}_2^{(p)} \\ \vdots \\ \left(I\text{-}rh\tilde{J}\right)\bar{Z}_s^{(p)} \end{bmatrix} = \begin{bmatrix} 0 \\ I_{21}\bar{Z}_1^{(p)} \\ \vdots \\ I_{21}\bar{Z}_{s-1}^{(p)} \end{bmatrix} + \begin{bmatrix} \bar{b}_{11}\bar{G}_1^{(p-1)} & +\bar{b}_{12}\bar{G}_2^{(p-1)} + & \cdots & +\bar{b}_{1s}\bar{G}_s^{(p-1)} \\ \bar{b}_{21}\bar{G}_1^{(p-1)} & +\bar{b}_{22}\bar{G}_2^{(p-1)} + & \cdots & +\bar{b}_{2s}\bar{G}_s^{(p-1)} \\ & \vdots & & \vdots \\ \bar{b}_{s1}\bar{G}_1^{(p-1)} & +\bar{b}_{s2}\bar{G}_2^{(p-1)} + & \cdots & +\bar{b}_{ss}\bar{G}_s^{(p-1)} \end{bmatrix} \quad (44)$$

where $I_{ii-1} = 0$ when i is odd.

For a second order system, we have $\left(\text{I-rh}\tilde{\text{J}}\right) = \begin{bmatrix} \text{I} & \text{-rhI} \\ \text{-rhJ} & \text{I} \end{bmatrix}$

where J is an approximation for the Jacobian of f with respect y and (44) can be written in the form

$$\begin{bmatrix} \text{I} & \text{-rhI} \\ \text{-rhJ} & \text{I} \end{bmatrix} \bar{Z}_R^{(p)} = 1_{RR-1}\bar{Z}_{R-1}^{(p)} + \sum_{i=1}^{s} \bar{b}_{Ri}\bar{G}_i^{(p-1)} = \bar{g}_R^{(p-1)},$$

R = 1,2,3,...,s (45)

where $\bar{g}_R = \left[g_R, g'_R\right]^T$, R = 1, 2, 3..., s. Thus,

$$\begin{bmatrix} \text{I} & \text{-rhI} \\ \text{-rhJ} & \text{I} \end{bmatrix} \begin{bmatrix} Z_R^{(p)} \\ Z'^{(p)}_R \end{bmatrix} = \begin{bmatrix} g_R^{(p-1)} \\ g'^{(p-1)}_R \end{bmatrix}$$

R=1,2,3...,s (46)

Now premultiplying (46) by $\begin{bmatrix} \text{I} & \text{rhI} \\ 0 & \text{I} \end{bmatrix}$, we obtain

$$\begin{bmatrix} \text{I-r}^2\text{h}^2\text{J} & 0 \\ \text{-rhJ} & \text{I} \end{bmatrix} \begin{bmatrix} Z_R^{(p)} \\ Z'^{(p)}_R \end{bmatrix} = \begin{bmatrix} g_R^{(p-1)} + \text{rhg}'^{(p-1)}_R \\ g'^{(p-1)}_R \end{bmatrix},$$

R=1,2,3...,s

From the first of these equations, we have

$$\left(\text{I-r}^2\text{h}^2\text{J}\right)Z_R^{(p)} = g_R^{(p-1)} + \text{rhg}'^{(p-1)}_R,\ \text{R} = 1, 2, 3..., \text{s} \quad (47)$$

Now from the first of (46), we have

$$Z_R^{(p)} - \text{rh}Z'^{(p-1)}_R = g_R^{(p-1)},\ \text{R} = 1, 2, 3..., \text{s}$$

and so we may set

$$Z'^{(p)}_R = \left[Z_R^{(p)} - g_R^{(p-1)}\right] \Big/ \text{rh},\ \text{R} = 1, 2, 3..., \text{s} \quad (48)$$

Next we calculate $g_R^{(p-1)}$ and $g'^{(p-1)}_R$, R = 1,2,3,..., s . First consider the residual $\bar{G}_R^{(p-1)}$ given by (42)

$$\bar{G}_R^{(p-1)} = \bar{x} + h\left(A \otimes I\right)\bar{F}\left(\bar{X}^{(P-1)}\right) - \bar{X}^{(P-1)}$$

Written in component form, this becomes

$$\begin{bmatrix} \bar{G}_1^{(p-1)} \\ \bar{G}_2^{(p-1)} \\ \vdots \\ \bar{G}_S^{(p-1)} \end{bmatrix} = \begin{bmatrix} x_n \\ x_n \\ \vdots \\ x_n \end{bmatrix} + h \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1s} \\ a_{21} & a_{22} & \cdots & a_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ a_{s1} & a_{s2} & \cdots & a_{ss} \end{bmatrix} \times$$

$$\begin{bmatrix} F\left(X_1^{(p-1)}\right) \\ F\left(X_2^{(p-1)}\right) \\ \vdots \\ F\left(X_s^{(p-1)}\right) \end{bmatrix} - \begin{bmatrix} X_1^{(p-1)} \\ X_2^{(p-1)} \\ \vdots \\ X_s^{(p-1)} \end{bmatrix}.$$

or, equivalently,

$$\bar{G}_R^{(p-1)} = x_n + h\sum_{i=1}^{s} a_{Ri}\, F\left(X_i^{(p-1)}\right) - X_R^{(p-1)}$$

R = 1, 2, 3, ..., s. (49)

Now making the substitutions

$$x_n = \left[y_n, y'_n\right]^T,\ X_R = \left[w_R, w'_R\right]^T,\ F\left(X_R\right) = \left[w'_R, f\left(w_R\right)\right]^T$$

R = 1, 2, 3, ..., s. then from (49), we get

$$\bar{G}_R^{(p-1)} = \begin{bmatrix} G_R \\ G'_R \end{bmatrix} = \begin{bmatrix} y_n + h\sum_{i=1}^{s} a_{Ri} w'^{(p-1)}_i - w_R^{(p-1)} \\ y'_n + h\sum_{i=1}^{s} a_{Ri} f\left(w_i^{(p-1)}\right) - w'^{(p-1)}_R \end{bmatrix}$$

R = 1,2,...,s. (50)

Substituting the values of $\bar{G}_R$, R = 1,2...,S from (50) into (45), we have

$$\begin{bmatrix} g_R^{(p-1)} \\ g'^{(p-1)}_R \end{bmatrix} = 1_{RR-1} \begin{bmatrix} Z^{(p)}{}_{R-1} \\ Z'^{(p)}_{R-1} \end{bmatrix}$$

$$+ \sum_{i=1}^{s} \bar{b}_{Ri} \begin{bmatrix} y_n + h\sum_{j=1}^{s} a_{ii} w'j^{(p-1)} - w_i^{(p-1)} \\ y'_n + h\sum_{j=1}^{s} a_{ii} f\left(w_j^{(p-1)}\right) - w'^{(p-1)}_i \end{bmatrix}$$

R = 1,2,...,s

where, l_{ii-1} when i is odd. These equations may be rewritten as

$$g_R^{(p-1)} = 1_{RR-1} Z_{R-1}^{(p)} + \sum_{i=1}^{s} \bar{b}_{Ri} y_n$$

$$+ h\sum_{i=1}^{s}\sum_{j=1}^{s} \bar{b}_{Ri} a_{ij} w_j'^{(p-1)} - \sum_{j=1}^{s} \bar{b}_{Ri} w_j^{(p-1)} \quad (51)$$

and

$$g'^{(p-1)}_R = 1_{RR-1} z'^{(p)}_{R-1} + \sum_{i=1}^{s} \bar{b}_{Ri} y'_n$$

$$+ h\sum_{i=1}^{s}\sum_{j=1}^{s} \bar{b}_{Ri} a_{ij} f\left(w^{(p-1)}{}_j\right) - \sum_{i=1}^{s} \bar{b}_{Ri} w_j^{(p-1)} \quad (52)$$

Finally we calculate $w_R^{(p)}$ and $w'^{(p)}_R$, R = 1,2,...,s. Consider the equation (41)

$$\bar{X}^{(p)} = \bar{X}^{(p-1)} + \left(S \otimes I\right)\bar{Z}^{(p)}.$$

Suppose S has the coefficients $\{S_{ij}\}$, i, j = 1,2 , ..., s. Then we have

$$\begin{bmatrix} X_1^{(p)} \\ X_2^{(p)} \\ \vdots \\ X_s^{(p)} \end{bmatrix} = \begin{bmatrix} X_1^{(p-1)} \\ X_2^{(p-1)} \\ \vdots \\ X_s^{(p-1)} \end{bmatrix} + \begin{bmatrix} S_{11}S_{12}\ldots S_{1s} \\ S_{21}S_{22}\ldots S_{2s} \\ \vdots \\ S_{s1}S_{s2}\ldots S_{ss} \end{bmatrix} \times \begin{bmatrix} Z_1^{(p)} \\ Z_2^{(p)} \\ \vdots \\ Z_s^{(p)} \end{bmatrix}$$

Thus, $X_R^{(p)} = X_R^{(p-1)} + \sum_{i=1}^{s} S_{Ri}\bar{Z}_i^{(p)}$, R=1,2...,s.

By substituting $X_R = \left[w_R w'_R\right]^T$ and $\bar{Z}_R = \left[Z_R Z'_R\right]^T$ R = 1,2,…,s, we obtain

$$W_R^{(p)} = W_R^{(p-1)} + \sum_{i=1}^{s} S_{Ri}Z_i^{(p)},\ R = 1,2,....,s, \tag{53}$$

$$W'^{(p)}_R = W'^{(p-1)}_R + \sum_{i=1}^{s} S_{Ri}Z'^{(p)}_i,\ R = 1,2,...,s, \tag{54}$$

Hence one step of the iteration is

Solve $\left(I - r^2h^2J\right)Z_R^{(p)} = g_R^{(p-1)} + rhg'^{(p-1)}_R$ R = 1,2,...,s,

Calculate $Z'^{(p)}_R = \left[Z_R^{(p)} - g_R^{(p-1)}\right] \Big/ rh$, R=1,2,....s.

Calculate $W_R^{(p)} = W_R^{(p-1)} + \sum_{i=1}^{s} S_{Ri}Z_i^{(p)}$, R = 1,2,....s.

Calculate $W'^{(p)}_R = W'^{(p-1)}_R + \sum_{i=1}^{s} S_{Ri}Z'^{(p)}_R$, R = 1,2,....s.

where, $g_R^{(p-1)}, g'^{(p-1)}_R$ are given by (51) and (52), respectively.

On convergence of the iteration, y_{n+1} and y'_{n+1} are obtained by substituting

$X_n = \left[y_n y'_n\right]^T$ and $F(X_R) = \left[w'_R f(w_R)\right]^T$

R = 1,2,… s into (46) giving

$$y_{n+1} = y_n + h\sum_{R=1}^{s} d_R w'_R,\ R = 1,2,...,s. \tag{55}$$

$$y'_{n+1} = y'_n + h\sum_{R=1}^{s} d_R f(w_R),\ R = 1,2,...,s. \tag{56}$$

Note that, to calculate y'_{n+1} we need to evaluate the functions $f(w_R)$, R= 1, 2,...s. We would like to avoid having to evaluate these functions. This and other computational aspects are discussed separately below for each of the cases, s = 2,3,4.

Case s = 2. Two-stage fourth order method: The two-stage fourth order implicit Runge-Kutta method is given by (6). For this method equations (55) become

$$y_{n+1} = y_n + \frac{h}{2}(w'_1 + w'_2) \tag{57}$$

$$y'_{n+1} = y'_n + \frac{h}{2}\left[f(w_1) + f(w_2)\right] \tag{58}$$

Now we form $f(w_1)$ and $f(w_2)$ from (38). From equation (38) with R = 1, we have

$$X_1 = x_n + \frac{h}{4}F(X_1) + \frac{3-2\sqrt{3}}{12}hF(X_2)$$

or, equivalently,

$$w_1 = y_n + \frac{h}{4}w'_1 + \frac{3-2\sqrt{3}}{12}hw'_2, \tag{59}$$

$$w'_1 = y'_n + \frac{h}{4}f(w_1) + \frac{3-2\sqrt{3}}{12}hf(w_2). \tag{60}$$

Similarly from equation (38) with R = 2, we obtain

$$w'_2 = y_n + \frac{3+2\sqrt{3}}{12}hw'_1 + \frac{h}{4}w'_2, \tag{61}$$

$$w'_2 = y'_n + \frac{3+2\sqrt{3}}{12}hf(w_1) + \frac{h}{4}f(w_2). \tag{62}$$

By subtracting (60) from (62), we obtain

$$\frac{h}{2}\left[f(w_1) + f(w_2)\right] = \sqrt{3}(w'_2 - w'_1)$$

and substituting into (58), gives

$$y'_{n+1} = y'_n + \sqrt{3}(w'_2 - w'_1)$$

Thus after convergence of the iteration (after Q iterations, say), we form

$$y_{n+1} = y_n + \frac{h}{2}\left(w_1'^{(Q)} + w_2'^{(Q)}\right),\ y'_{n+1} = y'_n + \sqrt{3}\left(w_1'^{(Q)} - w_2'^{(Q)}\right),$$

To start the iteration, we must predict the values of w_1, w_2, w'_1 and w'_2. We predict $w_1^{(0)}$ and $w_2^{(0)}$ by using the interpolating polynomial $Q_{3,n}(t)$ of Khiyal[6]. That is, we take $w_1^{(0)} = Q_{3,n}\left(t_n + \frac{3-\sqrt{3}}{6}h\right)$ and $w_2^{(0)} = Q_{3,n}\left(t_n + \frac{3+\sqrt{3}}{6}h\right)$.

This predictor cannot be used on the first step and so on this step, we propose to use

$$w_1^{(0)} = y_0 + \frac{3-\sqrt{3}}{6}hy'_{0,} \tag{63}$$

$$w_2^{(0)} = y_0 + \frac{3+\sqrt{3}}{6}hy'_{0,} \tag{64}$$

Having predicted $w_1^{(0)}$ and $w_2^{(0)}$, we may use these values in (59) and (61), to predict $w'^{(0)}_1$ and $w'^{(0)}_2$. Now by eliminating w'_1 form (59) and (61) we get w'_2, while by eliminating w', we obtain w'_1. This gives

$$hw_1'^{(0)} = -2\sqrt{3}y_n + 3w_1^{(0)} - \left(3-2\sqrt{3}\right)w_2^{(0)} \tag{65}$$

$$hw_2'^{(0)} = 2\sqrt{3}y_n - \left(3+2\sqrt{3}\right)w_1^{(0)} + 3w_2^{(0)} \tag{66}$$

In our codes, we use the same convergence strategy and action on divergence of the iteration step as discussed for the fourth order implicit Runge-Kutta method with a perfect square iteration scheme. To form the error estimate we use the formula given by (9) while the predictor is given by (34). Note that the cost of the Cooper and Butcher iteration scheme is the same as the cost of the perfect square iteration scheme (assuming both schemes require the same number of iterations) because we have to solve two systems of n linear equations at each step of the integration for both schemes. However, the matrix is the same for both systems for both schemes so we do not need to perform more than one matrix factorisation per step.

Case s = 3 three-stage sixth order method: The three-stage sixth order implicit Runge-Kutta method is given by (7). For this method equation (55) and (56) becomes

$$y_{n+1} = y_n + \frac{h}{18}\left(5w'_1 + 8w'_2 + 5w'_3\right) \tag{67}$$

$$y'_{n+1} = y'_n + \frac{h}{18}\left[5f(w_1) + 8f(w_2) + 5f(w_3)\right] \tag{68}$$

Now we form $f(w_R)$, R = 1,2,3 from equation (38). By substituting the coefficients of method (7) in equation (38) for R = 1,2,3 , we obtain

$$w_1 = y_n + \frac{5}{36}hw'_1 + \frac{10-3\sqrt{15}}{45}hw'_2 + \frac{25-6\sqrt{15}}{180}hw'_3, \tag{69}$$

$$w'_1 = y'_n + \frac{5}{36}hf(w_1) + \frac{10-3\sqrt{15}}{45}hf(w_2) + \frac{25-6\sqrt{15}}{180}hf(w_3). \tag{70}$$

$$w_2 = y_n + \frac{10+3\sqrt{15}}{72}hw'_1 + \frac{2}{9}hw'_2 + \frac{10-3\sqrt{15}}{72}hw'_3 \tag{71}$$

$$w'_2 = y'_n + \frac{10+3\sqrt{15}}{72}hf(w_1) + \frac{2}{9}hf(w_2) + \frac{10-3\sqrt{15}}{72}hf(w_3) \tag{72}$$

$$w_3 = y_n + \frac{25+6\sqrt{15}}{180}hw'_2 + \frac{10+3\sqrt{15}}{45}hw'_2 + \frac{5}{36}hw'_3, \tag{73}$$

$$w'_3 = y'_n + \frac{25+6\sqrt{15}}{180}hf(w_1) + \frac{10+3\sqrt{15}}{45}hf(w_2) + \frac{5}{36}hf(w_3) \tag{74}$$

To find $f(w_1)$ we eliminate $f(w_2)$ and $f(w_3)$ from the set of three equations (70), (72) and (74). Similarly we can find $f(w_2)$ or $f(w_3)$ by eliminating the other two values from the same set of equations. After a little manipulation, we obtain

$$hf(w_1) = -6y'_n + 5w'_1 - \frac{4}{3}\left(3-\sqrt{15}\right)w'_2 + \frac{1}{3}\left(15-4\sqrt{15}\right)w'_3 \tag{75}$$

$$hf(w_2) = 3y'_n - \frac{5}{6}\left(3+\sqrt{15}\right)w'_1 + 2w'_2 - \frac{5}{6}\left(3+\sqrt{15}\right)w'_3, \tag{76}$$

$$hf(w_3) = -6y'_n + \frac{1}{3}\left(15+4\sqrt{15}\right)w'_1 - \frac{4}{3}\left(3+\sqrt{15}\right)w'_2 + 5w'_3 \tag{77}$$

Finally we substitute the values of $hf(w_1)$, $hf(w_2)$ and $hf(w_3)$ from equations (75), (76) and (77) into (68) and simplify to give

$$y'_{n+1} = y'_n + \frac{1}{3}\left[5w'_1 - 4w'_2 + 5w'_3\right].$$

Thus after convergence of the iteration (after Q iterations, say), we form

$$y_{n+1} = y_n + \frac{h}{18}\left[5w'^{(Q)}_1 + 8w'^{(Q)}_2 + 5w'^{(Q)}_3\right] \quad (78)$$

$$y'_{n+1} = y'_n + \frac{1}{3}\left[5w'^{(Q)}_1 - 4w'^{(Q)}_2 + 5w'^{(Q)}_3\right] \quad (79)$$

To start the iteration, we must predict the values of $w_1, w_2, w_3, w'_1, w'_2$ and w'_3. We predict $w_1^{(Q)}$, $w_2^{(Q)}$, $w_3^{(Q)}$ by using the interpolating polynomial $Q_{5,n}(t)$ given by (A4.12) in Appendix 4 of Khiyal[6]. That is, we take

$$w_1^{(0)} = Q_{5,n}\left(t_n + \frac{5-\sqrt{15}}{10}h\right),\ w_2^{(0)} = Q_{5,n}\left(t_n + \frac{h}{2}\right) \text{ and}$$

$$w_3^{(0)} = Q_{5,n}\left(t_n + \frac{5+\sqrt{15}}{10}h\right)$$ However this predictor cannot be used on the first two steps. Instead, on the first step, we propose to use

$$w_1^{(0)} = y_0 + \frac{5-\sqrt{15}}{10}hy'_0 \quad (80)$$

$$w_2^{(0)} = y_0 + \frac{h}{2}y'_0, \quad (81)$$

$$w_3^{(0)} = y_0 + \frac{5+\sqrt{15}}{10}hy'_0 \quad (82)$$

On the second step, we take $w_1^{(0)} = Q_{3,n}\left(t_n + \frac{5-\sqrt{15}}{10}h\right)$,

$w_2^{(0)} = Q_{3,n}\left(t_n + \frac{h}{2}\right)$ and $w_3^{(0)} = Q_{3,n}\left(t_n + \frac{5+\sqrt{15}}{10}h\right)$. These can be obtained from equation (A4.9) in Appendix 4 of Khiyal[6].

Having predicted $w_1^{(0)}$, $w_2^{(0)}$ and $w_3^{(0)}$, we may use these values in the equations obtained from (69), (71) and (73). To find w'_1, we eliminate w'_2 and w'_3 from the set of three equations (69), (71) and (73). Similarly we can find w'_2 or w'_3 by eliminating the other two values from the same set of equations. After a little manipulation, we obtain

$$hw'_1 = -6y_n + 5w_1 - \frac{4}{3}\left(3-\sqrt{15}\right)w_2 + \frac{1}{3}\left(15-4\sqrt{15}\right)w_3 \quad (83)$$

$$hw'_2 = 3y_n - \frac{5}{6}\left(3+\sqrt{15}\right)w_1 + 2w_2 + \frac{5}{6}\left(3-\sqrt{15}\right)w_3, \quad (84)$$

$$hw'_3 = -6y_n + \frac{1}{3}\left(15+4\sqrt{15}\right)w_1 - \frac{4}{3}\left(3+\sqrt{15}\right)w_2 + 5w_3 \quad (85)$$

To form the error estimate we use the formula given by (9). This requires $y_{n+1}^{(0)}$ to be a fifth order approximation to $y(t_{n+1})$. For this purpose, we use the interpolating polynomial of degree five, $Q_{5,n}(t)$ say, which passes through the points (t_n, y_n, y'_n), $(t_{n-1}, y_{n-1}, y'_{n-1})$ and $(t_{n-2}, y_{n-2}, y'_{n-2})$. This polynomial may be evaluated at $t = t_{n+1}$, to give

$$y_{n+1} = -18y_n + 9y_{n-1} + 10y_{n-2} + h\left(9y'_n + 18y'_{n-1} + 3y'_{n-2}\right) \quad (86)$$

(This can be derived from equation (A4.12) of Appendix 4 of Khiyal[6] The use of (86) implies that we must carry out three steps before attempting to form the error estimate.

Case s = 4 four-stage eighth order method: The four-stage eighth order implicit Runge-Kutta method is given by (8). For this method, equation (55) and (56) becomes

$$y_{n+1} = y_n + h\left(m_1 + m_2\right) \quad (87)$$

$$y'_{n+1} = y'_n + h\left(m_3 + m_4\right) \quad (88)$$

Where:

$$m_1 = \frac{18-\sqrt{30}}{72}\left(w'_1 + w'_4\right),$$

$$m_2 = \frac{18+\sqrt{30}}{72}\left(w'_2 + w'_3\right),$$

$$m_3 = \frac{18-\sqrt{30}}{72}\left(f\left(w_1\right) + f\left(w_4\right)\right),$$

$$m_4 = \frac{18+\sqrt{30}}{72}\left(f\left(w_2\right) + f\left(w_3\right)\right),$$

Now we form $f(w_R)$, R = 1,2,3,4 from equations (38). By substituting the coefficients of method (8) in equations (38) for R = 1,2,3,4, we obtain

$$w_1 = y_n + \eta_1 hw'_1 + (\eta'_1 - \eta_3 + \eta'_4)hw'_2 + (\eta'_1 - \eta_3 - \eta'_4)hw'_3 + (\eta_1 - \eta_5)hw'_4, \quad (89)$$

$$w'_1 = y'_n + \eta_1 hf(w_1) + (\eta'_1 - \eta_3 + \eta'_4)hf(w_2) + (\eta'_1 - \eta_3 - \eta'_4)hf(w_3) + (\eta_1 - \eta_5)hf(w_4) \quad (90)$$

$$w_2 = y_n + (\eta_1 - \eta'_3 + \eta_4)hw'_1 + \eta'_1 hw'_2 + (\eta'_1 - \eta'_5)hw'_3 + (\eta_1 - \eta'_3 - \eta_5)hw'_4, \quad (91)$$

$$w'_2 = y'_n + (\eta_1 - \eta'_3 + \eta_4)hf(w_1) + \eta'_1 hf(w_2) + (\eta'_1 - \eta'_5)hf(w_3) + (\eta_1 - \eta'_3 + \eta_4)hf(w_4) \quad (92)$$

$$w_3 = y_n + (\eta_1 + \eta'_3 + \eta_4)hw'_1 + (\eta'_1 + \eta'_5)hw'_2 + \eta'_1 hw'_3 (\eta_1 + \eta'_3 - \eta_4)hw'_4, \quad (93)$$

$$w'_3 = y'_n + (\eta_1 + \eta'_3 + \eta_4)hf(w_1) + (\eta'_1 + \eta'_5)hf(w_2) + \eta'_1 hf(w_3) + (\eta_1 + \eta'_3 - \eta_4)hf(w_4) \quad (94)$$

$$w_4 = y_n + (\eta_1 + \eta_5)hw'_1 + (\eta'_1 + \eta_3 + \eta'_4)hw'_2 + (\eta'_1 + \eta_3 - \eta'_4)hw'_3 + \eta_1 hw'_4 \quad (95)$$

$$w'_4 = y'_n + (\eta_1 + \eta_5)hf(w_1) + (\eta'_1 + \eta_3 + \eta'_4)hf(w_2) + (\eta'_1 + \eta_3 - \eta'_4)hf(w_3) + \eta_1 hf(w_4) \quad (96)$$

Here the values of η'_1, η_1 are those given for method (8). From equations (90), (92), (94) and (96), after a little manipulation, we obtain

$$h[f(w_2)+f(w_3)] = n_1 / (4\eta'_3\eta_3 - \eta_5\eta'_5), \quad (97)$$

and

$$h[f(w_1)+f(w_4)] = n_2 / (4\eta'_3\eta_3 - \eta_5\eta'_5), \quad (98)$$

where, $n_1 = [2\eta'_3(w'_4 - w'_1) - \eta_5(w'_3 - w'_2)]$ and $n_2 = [2\eta_3(w'_3 - w'_2) - \eta'_5(w'_4 - w'_1)]$. Now we substitute the values of $f(w_2)+f(w_3)$ and $f(w_4)+f(w_1)$ from (97) and (98) into (88) and using the values of η_i, η'_i given in (8), we have

$$y_{n+1} = y_n - \frac{\sqrt{3}}{18}(n_3 + n_4)$$

where, $n_3 = (3+\sqrt{30})(15-2\sqrt{30})^{\frac{1}{2}}(w'_1 - w'_4)$ and

$$n_4 = (3-\sqrt{30})(15+2\sqrt{30})^{\frac{1}{2}}(w'_2 - w'_3)$$

Thus after convergence of the iteration (after Q iterations, say), we have

$$y_{n+1} = y_n + h(n_5 + n_6), \quad (99)$$

$$y'_{n+1} = y'_n + \frac{\sqrt{3}}{18}(n_7 + n_8) \quad (100)$$

where,

$$n_5 = \frac{18-\sqrt{30}}{72}\left(w'^{(Q)}_1 + w'^{(Q)}_4\right),$$

$$n_6 = \frac{18+\sqrt{30}}{72}\left(w'^{(Q)}_2 + w'^{(Q)}_3\right),$$

$$n_7 = (3+\sqrt{30})(15-2\sqrt{30})^{\frac{1}{2}}\left(w'^{(Q)}_1 - w'^{(Q)}_4\right),$$

$$n_8 = (3-\sqrt{30})(15+2\sqrt{30})^{\frac{1}{2}}\left(w'^{(Q)}_2 - w'^{(Q)}_3\right)$$

To start the iteration, we must predict the values of w_1, w_2, w_3, w_4, w'_1, w'_2, w'_3 and w'_4. We predict $w_1^{(0)}$, $w_2^{(0)}$, $w_3^{(0)}$, $w_4^{(0)}$ by using the interpolating polynomial $Q_{7,n}(t)$ given by equation (A4.13) of Appendix 4 of Khiyal[6]. That is, we take

$$w_1^{(0)} = Q_{7,n}\left(t_n + \left(\frac{1}{2} - \eta_2\right)h\right),$$

$$w_2^{(0)} = Q_{7,n}\left(t_n + \left(\frac{1}{2} - \eta'_2\right)h\right),$$

$$w_3^{(0)} = Q_{7,n}\left(t_n + \left(\frac{1}{2} + \eta'_2\right)h\right),$$

and

$$w_4^{(0)} = Q_{7,n}\left(t_n + \left(\frac{1}{2} + \eta_2\right)h\right).$$

However this predictor cannot be used on the first three steps. Instead, on the first step, we propose to use

$$w_1^{(0)} = y_0 + \left(\frac{1}{2} - \eta_2\right)hy'_0 \quad (101)$$

$$w_2^{(0)} = y_0 + \left(\frac{1}{2} - \eta_2'\right) hy_0' \quad (102)$$

$$w_3^{(0)} = y_0 + \left(\frac{1}{2} + \eta_2'\right) hy_0' \quad (103)$$

$$w_4^{(0)} = y_0 + \left(\frac{1}{2} + \eta_2\right) hy_0'. \quad (104)$$

On the second step, we take

$$w_1^{(0)} = Q_{3,n}\left(t_n + \left(\frac{1}{2} - \eta_2\right)h\right),$$

$$w_2^{(0)} = Q_{3,n}\left(t_n + \left(\frac{1}{2} - \eta_2'\right)h\right),$$

$$w_3^{(0)} = Q_{3,n}\left(t_n + \left(\frac{1}{2} + \eta_2'\right)h\right),$$

and

$$w_4^{(0)} = Q_{3,n}\left(t_n + \left(\frac{1}{2} + \eta_2\right)h\right).$$

These can be obtained from equation (A4.9) of Appendix 4 of Khiyal[6]. On the third step, we take

$$w_1^{(0)} = Q_{5,n}\left(t_n + \left(\frac{1}{2} - \eta_2\right)h\right),$$

$$w_2^{(0)} = Q_{5,n}\left(t_n + \left(\frac{1}{2} - \eta_2'\right)h\right),$$

$$w_3^{(0)} = Q_{5,n}\left(t_n + \left(\frac{1}{2} + \eta_2'\right)h\right),$$

and

$$w_4^{(0)} = Q_{5,n}\left(t_n + \left(\frac{1}{2} + \eta_2\right)h\right),$$

These can be obtained from equation (A4.12) of Appendix 4 of Khiyal[6].

Having predicted $w_1^{(0)}$, $w_2^{(0)}$, $w_3^{(0)}$, $w_4^{(0)}$, we may use these values in the equations obtained from (89), (91), (93) and (95). After a little manipulation, we obtain the following equations

$$hw_4' = (m_1 + m_2 + m_3)/(\eta_5'\eta_5 - 4\eta_4'\eta_4) \quad (105)$$

$$hw_3' = (m_4 + m_5 + m_6)/2'\eta_4 \quad (106)$$

$$hw_2' = -4\sqrt{105}\left[2\eta_3'(w_4 - w_1) - \eta_5(w_3 - w_2)\right] - hw_3' \quad (107)$$

$$hw_1' = -4\sqrt{105}\left[2\eta_3(w_3 - w_2) - \eta_5'(w_4 - w_1)\right] - hw_4' \quad (108)$$

where,

$$m_1 = (\eta_5' - 2\eta_4')y_n - \eta_5' w_1 + 2\eta_4' w_2$$

$$m_2 = 4\sqrt{105}(w_3 - w_2)(n_1 + n_2 + n_3)$$

$$m_3 = 4\sqrt{105}(w_4 - w_1)(n_4 + n_5 + n_6)$$

$$m_4 = y_n - w_1 - \eta_5 hw_4'$$

$$m_5 = -4\sqrt{105}(w_3 - w_2)\left[2\eta_3\eta_1 - \eta_5(\eta_1' - \eta_3 + \eta_4')\right]$$

$$m_6 = -4\sqrt{105}(w_4 - w_1)\left[2\eta_3'(\eta_1' - \eta_3 + n_4') - \eta_5'\eta_1\right]$$

$$n_1 = 4\eta_4'\eta_3(\eta_1 - \eta_3' + \eta_4),$$

$$n_2 = -2\eta_4'\eta_1'\eta_5 - 2\eta_5'\eta_1\eta_3,$$

$$n_3 = \eta_5'\eta_5(\eta_1' - \eta_3 + \eta_4'),$$

$$n_4 = 4\eta_4'\eta_1'\eta_3' + \eta_1\eta_5'^2,$$

$$n_5 = -2\eta_4'\eta_5'(\eta_1 - \eta_3' + \eta_4),$$

$$n_6 = -2\eta_5'\eta_3'(\eta_1' - \eta_3 + \eta_4'),$$

Thus we first calculate hw_4' from (105) and then hw_3' from (106). Finally we calculate hw_2' and hw_1' from (107) and (108), respectively.

At the end of the fourth step, we form an estimate of the error. To do so, we use the formula given by (9). In this case, we require $y^{(0)}_{n+1}$ to be a seventh order approximation to $y(t_{n+1})$ for this purpose, we use the interpolating polynomial of degree seven, $Q_{7,n}(t)$ say, interpolating to (t_n, y_n, y_n'),

$(t_{n-1}, y_{n-1}, y_{n-1}')(t_{n-2}, y_{n-2}, y_{n-2}')$ and $(t_{n-3}, y_{n-3}, y_{n-3}')$.

This polynomial may be evaluated at $t = t_{n+1}$ to give

$$y_{n+1}^{(0)} = \frac{1}{3}(-128y_n - 108y_{n-1} + 192y_{n-2} + 47y_{n-3}) + 4h(4y_n' + 18y_{n-1}' + 12y_{n-2}' + y_{n-3}')$$

(This can be derived from equation (A4.13) of Appendix 4. of Khiyal[6]) The use of (109) implies that we must carry out four steps with the implicit Runge-Kutta method before attempting to form the error estimate.

Numerical results: We are mainly concerned with solving oscillatory stiff initial value problem. We have tried a number of explicit scalar (nonstiff) test problems of the form (1). Thet give similar results and so we restrict our attention to one oscillatory example.

Example 1: $y''+\sinh(y) = 0$, $y(0) = 1$, $y'(0) = 0$.

This is a pure oscillation problem whose solution has maximum amplitude unity and period approximately six.

To verify that our techniques work for systems, we use as a test problem a moderately stiff system of two equations.

Example 2: $y''_1+\sinh(y_1+y_2) = 0$, $y_1(0) = 1$, $y'_1(0) = 0$

$y''_2+10^4y_2 = 0$, $y_2(0) = 10^{-8}$, $y'_2(0) = 0$

For this example we have deliberately introduce coupling from the stiff (linear) equation. Our intention here is that the stiff oscillatory component y_2 should be present only at the noise level as otherwise we would expect to choose the stepsize to resolve y_2. We have also tried other test problems. For example, we have tried the initial condition $y_2(0) = 10^{-8}$, $y_2(0) = 10^{-6}$, $y_2(0) = 10^{-4}$ and $y_2(0) = 10^{-2}$. However, in such problems, y_2 is not insignificant and, in general, a small stepsize has to be used to resolve accurately the rapidly oscillating solution component. These problems are then not really stiff since the small stepsize is required for accuracy, rather than stability, considerations. Thus we restrict ourselves to solving example 2.

Both examples 1 and 2 have been solved for $t\in[0,6]$. The stepsize is chosen initially to be $h = 1$ as this is more or less on scale for the problems. However, the stepsize is increase or reduce automatically by the code depending on the local error test and iterations convergence or divergence.

The error at the end point is obtained by comparing the computed solution with the solution obtained by using a fixed step code with a small stepsize for example 1 and for the first equation of example 2. For the second equation of example 2 we have used the exact solution. We denote the error at the end point by MAXERR where for the scalar equation it is |Error at t = 6| and for the system it is ||Error at t = 6||∞. It should be noted that the global error at the end points may give only a rough indication of the accuracy of each solution as, for oscillatory problems, a large global error may be caused by a small phase error and the approximate solution may be a good one. However, corresponding sets of results have been obtained also for ranges [0,1], [0,2],..., [0,5]. For given values of TOL and PMAX, the errors in the computed solution at each of the end points are of similar size to those obtained with $t\in[0,6]$ and the relative performance of the methods is similar.

We present the results for the cases where the maximum number of iterations perimitted for the methods, PMAX, is 5.

We present some statistics on the performance of the methods. The notation used in the Tables throughout this section is as follows.

- Number of evaluations of the differential equations right hand side f, FCN;
- Number of evaluations of the Jacobian $\frac{\partial f}{\partial y}$, JAC;
- Number of iterations overall, NIT;
- Number of iterations on steps where the iteration converge, NSIT;
- Number of steps overall, NST;
- Number of successful steps to complete the integration, NSST;
- Number of LU factorizations of the iteration matrix, FACT.

The mnemonic used for the method in the tables are as follows:

- Fourth order Implicit Runge-Kutta method given by (6) with perfect square iteration scheme, O4PS;
- Fourth order Implicit Runge-Kutta method given by (6) with Cooper and Butcher iteration scheme, O4CB;
- Sixth order Implicit Runge-Kutta method given by (7) with Cooper and Butcher iteration scheme, O6CB;
- Eighth order Implicit Runge-Kutta method given by (8) with Cooper and Butcher iteration scheme, O8CB;

For the fourth order method we can use two types of interpolating polynomials, namely the cubic polynomial $P_{3,n}(t)$ and the quartic polnomial $P_{4,n}(t)$ to find the back values on a change of stepsize. Almost results with both interpolants are same. Thus we present the results by using cubic interpolating polnomial.

For the sixth order method we can use two types of interpolating polynomials, namely fifth degree interpolating polynomial $P_{5,n}(t)$ and sixth degree interpolating polnomial $P_{6,n}(t)$ to find the back values on a change of stepsize. Again the results with both interpolants are almost the same. Thus we present the results by using fifth degree interpolating polnomial.

For the eighth order method we can use two types of interpolating polynomials, namely seventh degree interpolating polynomial $P_{7,n}(t)$ and eighth degree interpolating polnomial $P_{8,n}(t)$ to find the back values on a change of stepsize. Again the results with both interpolants are almost the same. Thus we present the results by using seventh degree interpolating polnomial. Result for example 1 are given in Table 1, 2, 3 and 4 with Tol = 10^{-4}, 10^{-6}, 10^{-8} and 10^{-10}, respectively. Result for example 2 are given in Table 5, 6, 7 and 8 with Tol = 10^{-4}, 10^{-6}, 10^{-8} and 10^{-10}, respectively..

We have also tested the example given by Kramarz[7].

Example 3: $y'' = 2498y+4998z$, $y(0) = 2$, $y'(0) = 0$.

$z'' = -2499y-4999z$, $z(0) = -1$, $z'(0) = 1$.

Table 1: Comparison of methods for example 1 with Tol = 10^{-4}

Method	MAXERR	FCN	JAC	NIT	NSIT	NST	NSST
O4PS	5.486×10^{-4}	158	1	79	59	42	33
O4CB	1.420×10^{-3}	56	1	28	28	6	6
O6CB	2.194×10^{-4}	81	1	27	27	6	6
O8CB	1.221×10^{-4}	152	1	38	33	13	12

Table 2: Comparison of methods for example 1 with Tol = 10^{-6}

Method	MAXERR	FCN	JAC	NIT	NSIT	NST	NSST
O4PS	1.902×10^{-5}	510	1	255	212	129	111
O4CB	9.074×10^{-5}	130	1	65	60	13	12
O6CB	8.539×10^{-6}	279	1	93	79	41	37
O8CB	6.766×10^{-7}	288	1	72	62	26	24

Table 3: Comparison of methods for example 1 with Tol = 10^{-8}

Method	MAXERR	FCN	JAC	NIT	NSIT	NST	NSST
O4PS	1.925×10^{-5}	1386	1	693	639	346	323
O4CB	6.260×10^{-8}	684	4	342	315	99	90
O6CB	2.933×10^{-6}	630	1	210	191	100	96
O8CB	8.897×10^{-6}	6704	1	1676	1641	775	768

Table 4: Comparison of methods for example 1 with Tol = 10^{-10}

Method	MAXERR	FCN	JAC	NIT	NSIT	NST	NSST
O4PS	1.830×10^{-7}	4668	1	2334	2263	1166	1136
O4CB	9.461×10^{-8}	4656	1	2328	2293	775	768
O6CB	2.969×10^{-8}	3825	1	1275	1241	775	768
O8CB	6.630×10^{-7}	52600	1	13150	13100	6154	6144

Table 5: Comparison of methods for example 2 with Tol = 10^{-4}

Method	MAXERR	FCN	JAC	NIT	NSIT	NST	NSST	FAC
O4PS	5.561×10^{-4}	698	2	349	265	293	254	52
O4CB	1.420×10^{-3}	56	1	28	28	6	6	1
O6CB	2.194×10^{-4}	81	1	27	27	6	6	1
O8CB	1.266×10^{-4}	284	2	71	56	16	13	4

Table 6: Comparison of methods for example 2 with Tol = 10^{-6}

Method	MAXERR	FCN	JAC	NIT	NSIT	NST	NSST	FAC
O4PS	1.921×10^{-5}	900	1	450	373	252	215	49
O4CB	9.074×10^{-5}	130	1	65	60	13	12	2
O6CB	2.454×10^{-6}	249	1	83	73	26	24	3
O8CB	9.707×10^{-7}	1192	3	298	280	80	74	7

Table 7: Comparison of methods for example 2 with Tol = 10^{-8}

Method	MAXERR	FCN	JAC	NIT	NSIT	NST	NSST	FAC
O4PS	1.409×10^{-8}	1796	1	898	856	446	431	21
O4CB	3.713×10^{-8}	668	3	334	311	89	82	8
O6CB	2.620×10^{-7}	2196	2	732	709	177	171	7
O8CB	5.251×10^{-8}	6804	1	1701	1666	775	768	8

Table 8: Comparison of methods for example 2 with Tol = 10^{-10}

Method	MAXERR	FCN	JAC	NIT	NSIT	NST	NSST	FAC
O4PS	4.046×10^{-10}	6164	1	3082	2941	1538	1476	90
O4CB	9.420×10^{-8}	4658	1	2329	2294	775	768	8
O6CB	1.030×10^{-8}	4008	1	1336	1302	775	768	8
O8CB	3.909×10^{-7}	52204	1	13051	13001	6154	6144	11

Table 9: Comparison of methods for example 3 with Tol = 10^{-4}

Method	MAXERR	FCN	JAC	NIT	NSIT	NST	NSST	FAC
O4PS	1.543×10^{-3}	6610	1	3305	2897	2104	1900	304
O4CB	2.559×10^{-2}	338	1	169	159	34	32	2
O6CB	2.810×10^{-2}	4155	2	1385	1359	398	392	7
O8CB	6.675×10^{-4}	5224	1	1306	1275	519	512	6

Table 10: Comparison of methods for example 3 with Tol = 10^{-6}

Method	MAXERR	FCN	JAC	NIT	NSIT	NST	NSST	FAC
O4PS	1302×10^{-5}	22940	1	11470	11058	6153	5946	275
O4CB	7.089×10^{-4}	14902	3	7451	7410	3792	3781	11
O6CB	2.090×10^{-3}	12765	3	4255	4222	1994	1985	10
O8CB	7.544×10^{-5}	9820	1	2455	2422	1032	1025	7

Table 11: Comparison of methods for example 3 with Tol = 10^{-8}

Method	MAXERR	FCN	JAC	NIT	NSIT	NST	NSST	FAC
O4PS	1.005×10^{-7}	74758	100	37379	35369	19104	18099	1305
O4CB	4.816×10^{-4}	67084	4	33542	33500	14608	14595	14
O6CB	9.131×10^{-5}	24510	1	8170	8124	4106	4096	9
O8CB	2.506×10^{-7}	19968	1	4992	4958	2055	2048	8

Table 12: Comparison of methods for example 3 with Tol = 10^{-10}

Method	MAXERR	FCN	JAC	NIT	NSIT	NST	NSST	FAC
O4PS	8.000×10^{-9}	230060	1	115030	113412	58015	57206	1010
O4CB	4.198×10^{-7}	188562	3	94281	94233	31870	31857	14
O6CB	1.542×10^{-5}	55350	1	18450	18405	8202	8193	8
O8CB	9.074×10^{-7}	69384	1	17346	17302	8202	8193	10

Exact solution of this problem is $y(x) = 2\cos x$, $z = -\cos x$. We have find the solution for $x = 4\pi$, with initial $h = \pi/4$. Results for example 3 are given in Table 9, 10, 11 and 12 with Tol = 10^{-4}, 10^{-6}, 10^{-8} and 10^{-10}, respectively.

REFERENCES

1. Butcher, J.C.., 1964. Implicit Runge-Kutta processes. Math. Comput. (MTAC)., 18: 50-64.
2. Dahlquist, G., 1963. A special stability problem for linear multistep methods. BIT., 3: 27-43.
3. Thomas, R.M., 1987. Efficient fourth order P-stable formulae. BIT., 27: 599-614.
4. Gladwell, I. and R.M. Thomas, 1990. Efficiency of methods for second order problems. IMA J. Numer. Anal., 10: 181-207.
5. Cooper, G.J. and J.C. Butcher, 1983. An iteration scheme for implicit Runge-Kutta methods. IMA J. Numer. Anal., 3: 127-140.
6. Khiyal, M.S.H., 1991. Efficient algorithms based on direct hybrid methods for second order initial value problems. Ph.D. Thesis, The University of Manchester Institute of Science and Technology, Manchester.
7. Kramarz, L., 1980. Stability of collocation methods for the numerical solution of $y'' = f(x,y)$. BIT., 20: 215-222.