Parallel Implicit Runge-Kutta Methods for Stiff ODEs

Bashir M. S. Khalaf **Abdulhabib A. A. Murshid**
College Of Education
University of Mosul, Iraq

Received on: 20/08/2002 **Accepted on: 12/10/2002**

ABSTRACT

The main objective of this paper is to develop and construct numerical algorithms for solving stiff system of ordinary differential equations (ODEs) which are suitable for running on parallel computers (MIMD computers).Semi-parallel implicit Runge-Kutta methods have been derived and parallel predictor - corrector methods are developed.

Keywords: stiff ordinary differential equations (ODEs), Runge-Kutta methods, parallel implicit methods, predictor - corrector methods.

طرائق رنج – كوتا الضمنية المتوازية لحل المعادلات التفاضلية الاعتيادية الصلبة

بشير محمد صالح خلف **عبدالحبيب مرشد**

كلية التربية

جامعة الموصل

تاريخ الاستلام: 2002/08/20 **تاريخ القبول: 2002/10/12**

الملخص

هدف هذا البحث الرئيسي هو تطوير وتركيب خوارزميات عددية وتركيبها لحل المعادلات التفاضلية الاعتيادية الصلبة الملائمة للتنفيذ في الحاسبات المتوازية (حاسباتMIMD). تم في هذا البحث اشتقاق طرائق ضمنية متوازية من نوع طرائق رنج – كوتا. وكذلك تم تطوير طرائق متوازية من نوع التخمين– والتصحيح.

الكلمات المفتاحية: المعادلات التفاضلية الاعتيادية الصلبة، طرائق رنج – كوتا، طرائق ضمنية متوازية، طرائق التخمين– والتصحيح.

1- Arithmetic Mean and Geometric Mean Runge-Kutta Methods:

The general form of an r-step arithmetic mean (AM) Runge-Kutta method is:

$$K_1 = f(y_n),\ K_i = (y_n + h_n \sum_{j=1}^{i-1} b_{ij} K_j \qquad (1a,b)$$

and

$$y_{n+1} = y_n + \sum_{i=1}^{r} w_i K_i, \qquad \sum_{i=1}^{r} w_i = 1 \qquad (1c)$$

with appropriate values of the b's and w's . If instead of the weighted arithmetic mean in eq. (1) we use the geometric mean, then the geometric mean (GM) Runge-Kutta formula is [2]:

$$y_{n+1} = y_n + \prod_{i=1}^{r} KP_i, \qquad \sum_{i=1}^{r} P_i = 1 \qquad (1d)$$

An example of the AM- Runge-Kutta formula is the 4^{th} -order formula generated by the symbol matrix [2]

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 \\ 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1/6 & 1/3 & 1/3 & 1/6 \end{bmatrix}_{AM}$$

And an example of the GM-Runge-Kutta formula is the 4th-order formula generated by the symbol matrix

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 0 \\ 7/24 & 5/24 & 0 & 0 \\ -1/4 & -59/20 & 21/5 & 0 \\ 1/6 & -1/3 & 1 & 1/6 \end{bmatrix}_{AM}$$

If, while maintaining the order of accuracy of the formula, we can choose some of the w's , e.g. w, , to vanish and the subsequent k's do not depend on k;, then an economical formula is obtained. An example of such formula is the 4th-order formula generated by the symbol matrix

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 1/21 & 0 & 0 & 0 \\ -\infty & \infty & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1/6 & 4/6 & 0 & 1/6 \end{bmatrix}_{AM}$$

We can get the formula:

$k_i = f(x_n, y_n)$, $k_2 = f(x_n + h/2, y_n + h/2k_1)$, $k_4 = f(X_n + h, y_n + hk_1)$ (2a)

and

$y_{n+1} = y_n + l/6h(k_1 + 4k_2 + k_4)$ (2b)

In this formula, we notice that while it is effectively a 3-stage formula generated by the symbol matrix.

$$\begin{bmatrix} 0 & 0 & 0 \\ 1/2 & 0 & 0 \\ -\infty & \infty & 0 \\ 1 & 0 & 0 \\ 1/6 & 4/6 & 1/6 \end{bmatrix}_{AM}$$

The quantity K_4 does not depend on k_2 and thus allows us to compute k_2 and k_4 simultaneously on a parallel computer. By replacing h by - h in the formula (2) we obtain the corresponding implicit formula as:

$L_1 = f(x_{n+l}, y_{n+l})$,

$L_2 = f(x_{n+1} - l/2h, y_{n+1} - h/2\, L_1)$,

$L_4 = f(X_{n+1} - h, y_{n+1} - h, y_{n+1} - h\, L_1)$, (3)

And

$y^*_{n+1} == Y_n + h/6\,(L_1 + 4L_2 + L_4)$

Now we will try to find other RK formulas of the type given by eqs (2) and (3) and other forms:

2- Semi-Parallel Implicit 3-Stage AM-Runge-Kutta Method:

The 3-stage AM-Runge-Kutta formula is of the form:

$k_1 = f(x_n, y_n)$ (4a)

$k_2 = f(x_n + a_1 h, y_n + a_1 h k_1)$, (4b)

$k_3 = f(x_n, + h, y_n + a_2 h k_1 + a_3 h k_2)$, (4c)

and

$y_{n+l} = y_n + h\,(w_1 k_1 + w_2 k_2 + w_3 k_3)$ (4d)

By setting $a_2 + a_3 = 1$ and by comparing the r.h.s. of equation (4d) with the Taylor series expansion for $y(X_{n+1})$, the following equations of conditions were obtained :

$h^2\, f.fy : w_2 B_1 + w_3 (a_2 + a_3) = ½$,

$h^3\, f^2 fyy$: 1/2 W2ai2 + 1/2 W3 (a2 + a3)2 = 1/6 , (5)

$h^3\, ff^2 y : w_3 a_1 a_3 = 1/6$

For the purpose of parallel computation we require that $a_3 = 0$, therefore, by solving this system, for w_2 and w_3 in terms of a_1 and a_2 we get:

$$W_2 = \frac{-3a_2 + 2}{6a_1(a_1 - a_2)} \qquad \text{and} \qquad W_3 = \frac{-3a_1 - 2}{6a_{21}(a_1 - a_2)}$$

Since Wi+w2+W3=l and by setting a_1 = Vi, we find that the solution is:

$a_1 = 1/2$, $a_2 = 1$, $w_1 = 1/6$, $w_2 = 2/3$, $w_3 == 1/6$

The resulting formula is thus;

$k_1 = f(x_n, y_n)$,

$k_2 = f(x_n + ½h, y_n + ½hk_l)$,

$k_3 = f(x_n + h, y_n + hk_1)$

and

$y_{n+1} = y_{n+l} + 1/6\,h(k_1 + 4k_2 + k3)$ (6)

which is a semi-parallel 3-stage AM-Runge-Kutta method, because evaluations of k_2 and k_4 are independent, so they can be computed simultaneously, but sequentially to k_1 , so that the formula is semiparallel explicit Runge-Kutta (SPERK) method .

Since we are solving stiff ODEs[8], we require implicit forms. By replacing h by- h (i.e.backward integration) in the formula (6) we obtain the corresponding implicit formula as:

$$\left.\begin{array}{l} L_1 = f(x_{n+1}, y_{n+1}), \quad L_2 = f(x_{n+1} - \frac{1}{2}h, y_{n+1} - \frac{1}{2}hL_1), \\ L_4 = f(x_{n+1} - h, y_{n+1} - hL_1) \end{array}\right\} \qquad (7)$$

And

$$y^*_{n+1} = y_n + \frac{h}{6}(L_1 + 4h_2 + L_4),$$

In formula (7) we see that computing y_{n+1} is depending on itself so that the method is Implicit Runge-Kutta (IRK). Also, we note that L_4 does not depend on L_2, so they can be computed in parallel (i.e. simultaneously) using two different processors. But simultaneous evaluation of L_2 and L_4 is sequential to evaluation of L_1, therefore, the method is a semi-parallel IRK (SPIRK) method.

The parallel predictor-corrector form of (6) and (7) is:

$$\left.\begin{aligned}
& y_{n+1}^{p} = y_{n}^{c} + \frac{h}{6}(K_1^c + K_2^c + K_4^c) \\
& y_{n+1}^{p} = y_{n}^{p} + \frac{h}{6}(L_1^p + L_2^p + L_4^p) \\
& \text{Where} \\
& K_1^c = f(x_n, y_n^c), \quad L_1^p = f(x_{n+1}, y_{n+1}^p) \\
& K_2^c = f(x_n + \frac{1}{2}h, y_n^c + \frac{h}{2}K_1^c), \; L_2^p = f(x_{n+1} - \frac{h}{2}, y_{n+1}^p - \frac{h}{2}L_1^p) \\
& K_4^c = f(x_n + h, y_n^c + hK_1^c), \quad L_4^p = f(x_{n+1} - h, y_{n+1}^p - hL_1^p)
\end{aligned}\right\} \quad (8)$$

The computation diagram is:

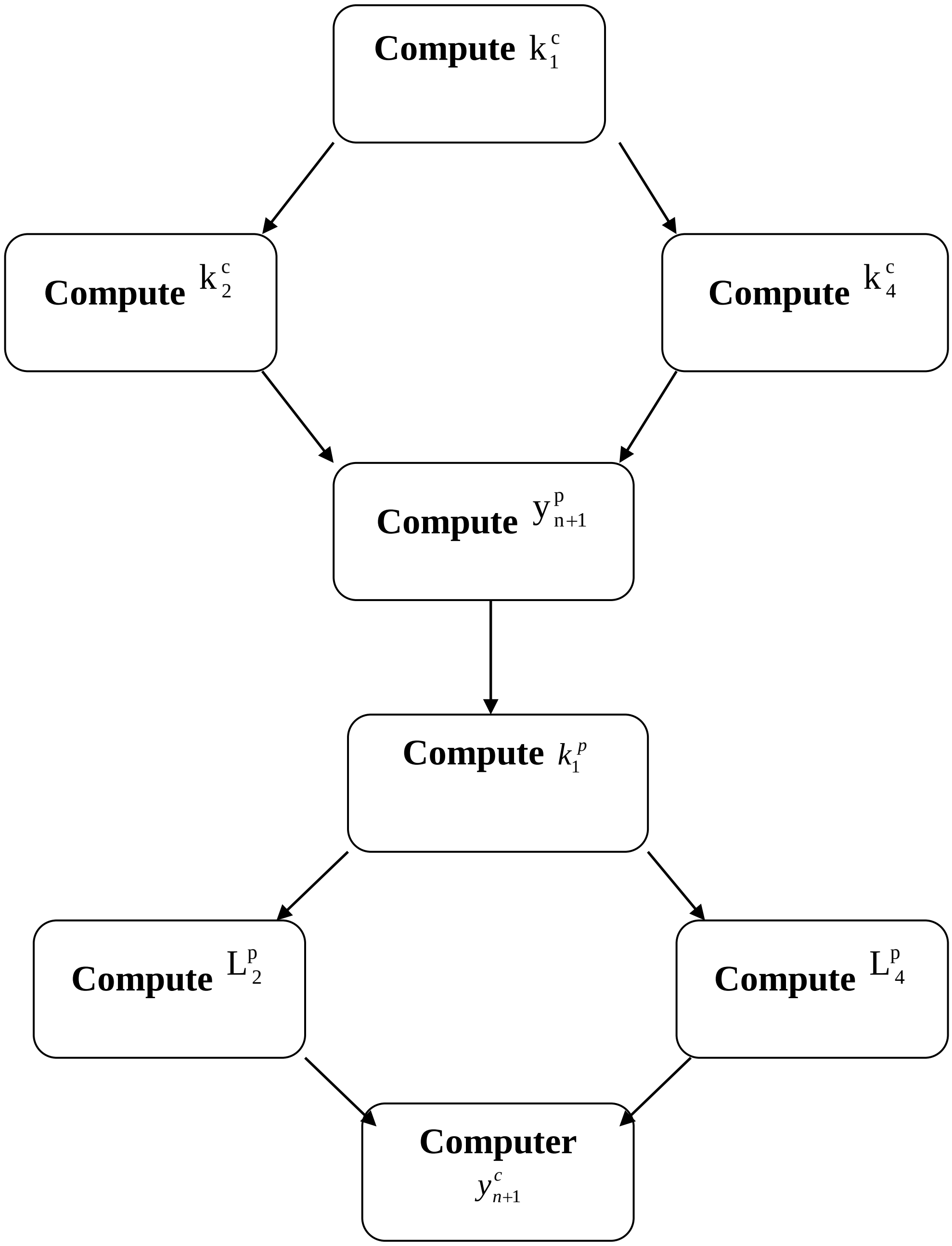


Fig. (1) The computation diagram of (8).

3- Semi-Parallel Implicit Geometric Mean Runge-Kutta (SPIGMRK) Method:

Now we repeat the analysis of equations (4) with (4d) replaced by

Yn+1 = yn + h $\mathbf{K_1^{(1-p-q)}}$ $K_2^p K_4^q$, 0<p, q<l (9)

to get a 2-stae SPIGMRK formula .

The Taylor series for y(xn+1) are :

h^2ffy : pa1 + q(a_2+a_3) = 1/2 (10a)

h^3f^2fyy: l/2pa_1^2 + l/2q(a_2 + a_3)^2 =1/6 (l0b)

h^3ff^2y:Pqa_1 (a_2 + a_3) + l/2p(p - l) a_1^2 + qa_1 a_3

+ l/2q(q - 1)(a_2 + a_3)^2 = 1/6

By Putting a_3 = 0, we get:

Pa_1+qa_2= l/2 (11a)

Pa_1^2 +q a_1^2 = l/3 (11b)

Pqa_1a_2 l/2P(P-1)a_1^2 = l/2q(q-1)a_1^2 = l/6 (11c)

Unlike the linear case, (10c) is not immediately violated with the choice a_3 =0 Solving this system for p and q from (11a) and (11b) in terms of a_1 and a_2 we have:

$$P = \frac{-3a_2 + 2}{6a_1(a_1 - a_2)} \quad \text{and} \quad q = \frac{3a_1 - 2}{6a_2(a_1 - a_2)} \qquad \mathbf{(12)}$$

Therefore with a_1 and a_2 chosen to satisfy (11a) and (11b) result in (11c) is not being satisfied. The resulting formula is thus order 2. As an example of the solution, setting a_1 = 2/3 and a_2= 1 we obtain p=3/4 and q=0. The resulting formula is given by:

k_1=f (x_n, y_n), k_2=f(x_n+2/3/h, y_n+2/3hk_1), k_3=f (x_n+h, y_n + hk_1) and

$y_{n+1} = y_n + hk_1^{1/4} k_2^{3/4}$

This is a parallel 2-stage formula since k_3 dose not need to be evaluated. The corresponding implicit formula is:

L_1=f(x_{n+1},y_{n+1}), L_2=f(x_{n+1}-2/3h,y_{n+1}-2/3hL_1), L_3=f(x_{n+1}-h,y_{n+1}-hL_1)

and

$$y^{*}_{n+1} = y_n + hL_1^{1/4}L_2^{3/4}$$

4. Another semi–parallel implicit 2-stage GM–Runge–Kutta Formula:

Now, we try to find the solution to satisfy equations (11a) and (11c) but not (11b), we have :

$q\, a_2 = ½ - pa_1$

and substituting into (11c) we get :

$Pa_1(1/2-pa_1)+1/2p(p-1)a_1^2+½(1/2-pa_1)(1/2-pa_1-a_2)=1/6$ (15)

Putting $Pa_1 = 0$

-eq.n. (11a) becomes $q\, a_2 = 1/2$, -eqn (15) becomes $a_2 = -1/6$

an example of the solution is:

$P=0$, $a_1 = 0$, $a_2 = -1/6$, $q=-3$

which leads to the following formula:

$K_1 = f(x_n+1/6h , y_n - 1/6hk_1)$

And

$y_{n+1} = y_n + hk_1^4 k_3^{-3}$ (16)

which is a semi-parallel 2-stage GM-Runge-Kutta method. The corresponding implicit formula for this method is:

$L_1=f(x_n,y_n)$, $L_3=f(x_n-1/6h, y_n+1/6hL_1)$,

And

$$y^{*}_{n+1} = y_n + hL_1^{4}L_3^{-3} \quad (17)$$

Form (17) is considered as a parallel mode because the calculation of L_2 is not needed.

5-semi-parallel implicit 3rd order Runge-Kutta method:

In similar way as described previously, we can obtain another (SPIRK) of order three in the following form:

$k_1= f(x_n, y_n)$, $k_2 = (x_n+1/4h, y_{n+1}-1/4hL_1)$,

$L_3=f(x_{n+1}-h, y_{n+1}-hL_1)$

And

$$y_{n+1} = y_n + \frac{h}{12}(3K_1 + 4K_2 + 5K_1) \qquad (18)$$

The corresponding implicit formula is:

$L_1=(x_{n+1},y_{n+1})$, $L_2=(x_{n+1}-1/4h,y_{n+1}-1/4hL_1)$, $L_3=(x_{n+1}-h,y_{n+1}-hL_1)$,

And

$$y_{n+1} = y_n + \frac{h}{12}(3L_1 + 4L_2 + 5L_1) \qquad (19)$$

6-Fourth order parallel predictor- corrector RK (PPCRK) method:

The general form of fourth order Runge-kutta method is:

$k_1 = f(x_n, y_n)$,
$k_2 = f(x_n+c_2h, y_n+c_2hk_1)$,
$k_3 = f(x_n+c_3h, y_n+h(a_{31}k_2+(c_3-a_{31})k_1))$,
$k_4 = f(x_n+c_4h, y_n+h(a_{41}k_2+ a_{42}k_3+(c_4-a_{41}-a_{42})k_1))$,
$y_{n+1} = y_n+ h(w_1k_1+w_2k_2+w_3k_3+w_4k_4)$

By using series expansion , one obtains after rather complicated calculations , the following system [35]

$w_1 + w_2 + w_3 + w_4 = 1$, $w_3c_2a_{31}+w_4(c_3a_{42} + c_2a_{41})=1/6$,
$w_2c_2 + w_3c_3+w_4c_4 = 1/2$, $w_3c_2a_{31}+w_4c_4(c_3a_{42} + c_2a_{41})=1/8$,
$w_2c_2^2 + w_3c_3^2+w_4c_4^2 = 1/3$, $w_3c_2^2a_{31}+w_4(c_3^2a_{42} + c_2^2a_{41})=1/12$,
$w_2c_2^3 + w_3c_3^3+w_4c_4^3 = 1/4$, $w_4c_2a_{42}= 1/24$,

We have eight equation in ten unknowns , and hence we can choose two quantities arbitrarily. If we assume that $c_2=c_3$, which seems rather natural, we find:

$c_2=c_3=½$, $c_4=1$, $w_1=w_4=1/6$,$w_2+w_3 = 2/3$, $a_{41}+a_{42} = 1$, $w_3a_{31}=1/6$

If we further choose $w_2=w_3$ and for the purpose of parallel computation we require $a_{42} = 0$ we get:

$w_2 = w_3 = 1/3$, $a_{41} = 1$, $+a_{31}=1/2$

Thus we have the final formula:

$$
\begin{aligned}
&k_1 = f(x_n, y_n),\\
&k_2 = f(x_n+1/2h,\ y_n+1/2hk_1),\\
&k_3 = f(x_n+1/2h,\ y_n+1/2hk_2), \qquad (20)\\
&k_4 = f(x_n+h,\ y_n+hk_2),\\
&y_{n+1} = y_n + h/6\,(k_1 + 2k_2 + 2k_3 + k_4)
\end{aligned}
$$

This is a (SPERK) method of order four:

From (20) we can get a semi-parallel implicit RK (SPIRK) method of order four by backward integration process a follows:

$$
\begin{aligned}
&L_1 = f(x_{n+1}, y_{n+1}),\\
&L_2 = f(x_{n+1} - \frac{1}{2}h,\ y_{n+1} - \frac{1}{2}hL_1),\\
&L_3 = f(x_{n+1} - \frac{1}{2}h,\ y_{n+1} - \frac{1}{2}hL_2),\\
&L_4 = f(x_{n+1} - h,\ y_{n+1} - hL_2), \qquad (21)\\
&y_n = y_{n+1} - \frac{h}{6}(L_1 + 2L_2 + 2L_3 + L_4),
\end{aligned}
$$

Rearranging this formula we get:

$$y_{n+1} = y_n + \frac{h}{6}(L_1 + 2L_2 + 2L_3 + L_4)$$

In this implicit method L_3 and L_4 are independent and can be computed in parallel. To get a parallel predictor-corrector RK (PPCRK) mode of the methods (20) and (21) , we can write them in the following form:

$y^p_{n+2} = y^p_{n+1} + \frac{h}{6}(K^p_1 + 2K^p_2 + 2K^p_3 + K^p_4)$, predictor from

$y^c_{n+1} = y^c_n + \frac{h}{6}(L^p_1 + 2L^p_2 + 2L^p_3 + L^p_4)$,corrector from

Where

$K_1^p = f(x_{n+1}, y_{n+1}^p),$ $\qquad L_1^p = f(x_{n+1}, y_{n+1}^p),$

$$K_2^p = f(x_{n+1} + \frac{1}{2}h, y_{n+1}^p + \frac{1}{2}hK_1^p),$$

$$L_2^p = f(x_{n+1} - \frac{1}{2}h, y_{n+1}^p - \frac{1}{2}hL_1^p),$$

$$K_3^p = f(x_{n+1} + \frac{1}{2}h, y_{n+1}^p + \frac{1}{2}hK_2^p),$$

$$L_3^p = f(x_{n+1} - \frac{1}{2}h, y_{n+1}^p - \frac{1}{2}hL_2^p),$$

$$K_4^p = f(x_{n+1} + h, y_{n+1}^p + hK_2^p),$$

$$L_4^p = f(x_{n+1} - h, y_{n+1}^p - hL_2^p),$$

Where y_1^p is computed in advance

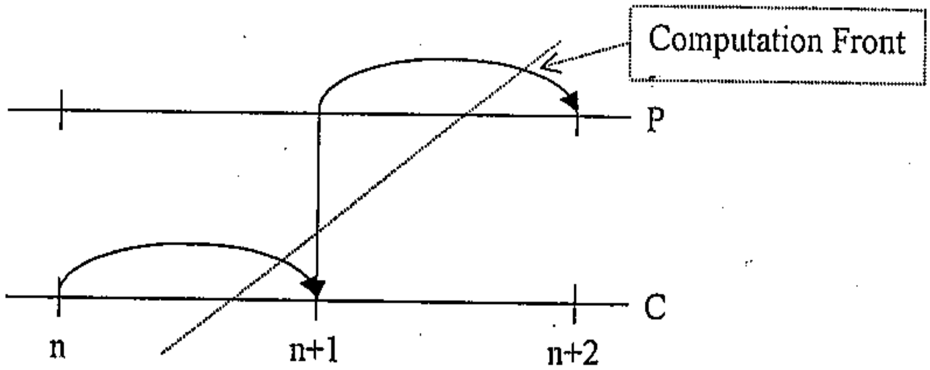


Fig. (2): Information flow in the PPCRK scheme.

The problem with this PPCRK mode is that: the predictor formula always compute y^p_{n+2} depending on predicted values and because we are solving stiff ODEs this means that the predictor formula will be unstable Hence, we require a modification of the PPCRK mode , so that the predictor formula depends on corrected values ,we can develop a formula which repeats the

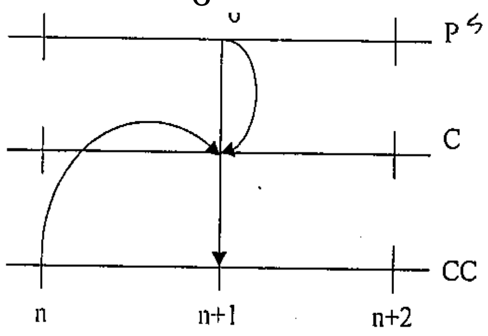
correction twice: (y_1 , y_2 and y_3 can be computed first sequentially).
First calculation stage:

$$y^{c}_{n+1} = y^{cc}_{n} + \frac{h}{6}(L^{p}_{1} + 2L^{p}_{2} + 2L^{p}_{3} + L^{p}_{4})$$

Second calculation stage:

$$y^{p}_{n+2} = y^{c}_{n+1} + \frac{h}{6}(K^{c}_{1} + 2K^{c}_{2} + 2K^{c}_{3} + K^{c}_{4})$$

$$y^{cc}_{n+1} = y^{cc}_{n} + \frac{h}{6}(L^{c}_{1} + 2L^{c}_{2} + 2L^{c}_{3} + L^{c}_{4})$$



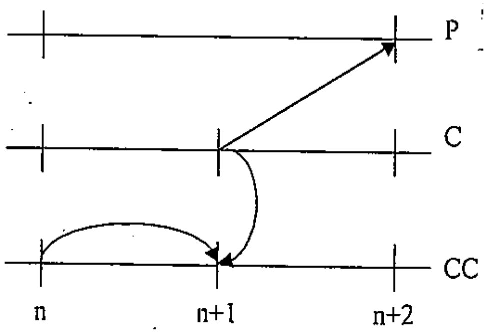


Fig.(3): Diagram of the first calculation stage.

Fig.(4): Diagram of the second calculation stage.

7-PPCRK Method for a system of first order ODEs:

Runge-Kutta method can be applied directly to differential equations of higher order, taking , for example , the equation $y''=f(x,y,y')$, we put $y'=z$ and obtain the following system of first-order equation:

$y'=z$, $z'=f(x,y,z)$

This is a special case of:

$y'= F(x,y,z)$,

$z'= G(x,y,z)$,

(22)

which can be integrated (using method 6):

$K_1 = F(x_n, y_n, z_n)$, $L_1 = G(x_n, y_n, z_n)$,

$$K_2 = F(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hK_1, z_n + \frac{1}{2}hL_1),$$
$$L_2 = G(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hK_1, z_n + \frac{1}{2}hL_1),$$
$$K_3 = F(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hK_2, z_n + \frac{1}{2}hL_2),$$
$$L_2 = G(x_n + \frac{1}{2}h, y_n + \frac{1}{2}hK_2, z_n + \frac{1}{2}hL_2), \qquad (23)$$
$$K_4 = F(x_n + h, y_n + hK_2, z_n + hL_2),$$
$$L_4 = G(x_n + h, y_n + hK_2, z_n + hL_2),$$
$$y_{n+1} = y_n + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4),$$
$$z_{n+1} = z_n + \frac{h}{6}(L_1 + 2L_2 + 2L_3 + L_4),$$

The corresponding implicit formula is:

$m_1 = F(x_{n+1}, y_{n+1}, z_{n+1})$, $S_1 = G(x_{n+1}, y_{n+1}, z_{n+1})$,

$$m_2 = F(x_{n+1} - \frac{1}{2}h, y_{n+1} - \frac{1}{2}hm_1, z_{n+1} - \frac{1}{2}hS_1),$$
$$S_2 = G(x_{n+1} - \frac{1}{2}h, y_{n+1} - \frac{1}{2}hm_1, z_{n+1} - \frac{1}{2}hS_1),$$
$$m_3 = F(x_{n+1} - \frac{1}{2}h, y_{n+1} - \frac{1}{2}hm_2, z_{n+1} - \frac{1}{2}hS_2),$$
$$S_3 = G(x_{n+1} - \frac{1}{2}h, y_{n+1} - \frac{1}{2}hm_2, z_{n+1} - \frac{1}{2}hS_2), \qquad (24)$$
$$m_4 = F(x_{n+1} - \frac{1}{2}h, y_{n+1} - \frac{1}{2}hm_2, z_{n+1} - \frac{1}{2}hS_2),$$
$$S_4 = G(x_{n+1} - \frac{1}{2}h, y_{n+1} - \frac{1}{2}hm_2, z_{n+1} - \frac{1}{2}hS_2),$$

$$y^*_{n+1} = y_n + \frac{h}{6}(m_1 + 2m_2 + 2m_3 + m_4),$$

$$z^*_{n+1} = z_n + \frac{h}{6}(S_1 + 2S_2 + 2S_3 + S_4),$$

We can modify the algorithms (23) and (24) to parallel predictor-corrector RK (PPCRK) version as follows:

1) We compute y_1 and z_1 by using (13) in advance, then
2) We compute the following in parallel:

$$y^p_{n+2} = y^p_{n+1} + \frac{h}{6}(K^p_1 + 2K^p_2 + 2K^p_3 + K^p_4),$$

$$z^p_{n+1} = z^p_n + \frac{h}{6}(L^p_1 + 2L^p_2 + 2L^p_3 + L^p_4),$$

$$y^c_{n+1} = y^c_n + \frac{h}{6}(m^p_1 + 2m^p_2 + 2m^p_3 + m^p_4), \qquad (25)$$

$$z^c_{n+1} = z^c_n + \frac{h}{6}(S^p_1 + 2S^p_2 + 2S^p_3 + S^p_4),$$

Where p means using predicted values and c denotes the using of corrected values.

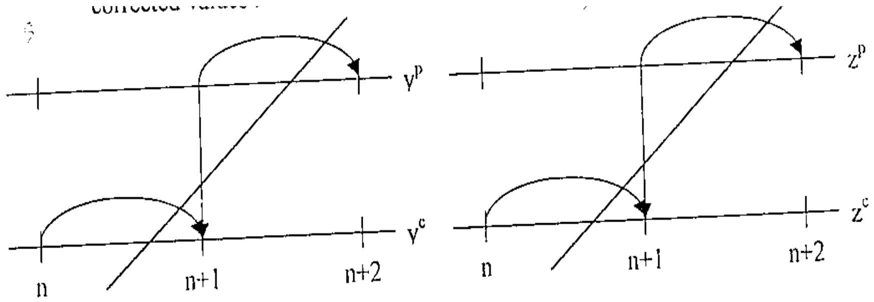


Fig. (5): Information flow in the PPCRK scheme (25).

Processor y^p compute values of y^p_{n+2}, n=0,1,…
Processor z^p compute values of z^p_{n+2}, n=0,1,…
Processor y^c compute values of y^c_{n+2}, n=0,1,…
Processor z^c compute values of z^c_{n+2}, n=0,1,…

The message passing between the processors will be as follows:

Processor y^p computes values of y^p_{n+2} , where the values of y^p_{n+1} and z^p_{n+1} are ready at processor y^p from last step, at the same time processor z^p computes the value of z^p_{n+2} as the values of y^p_{n+1} and z^p_{n+1} are ready from last step. At the same time, processor y^c computes the value of y^c_{n+1}, as the values of y^c_n , z^c_n , y^p_{n+1} and z^p_{n+1} ready from last step . At the same time, processor z^c computes values of z^c_{n+1} as the values of y^c_n , z^c_n , y^p_{n+1} and y^p_{n+1} are known from last step , now ,

1- Processor y^p sends the values of y^p_{n+2} to the processors z^p , y^c and z^c and they wait antil the message will be received.

2- Processor z^p sends the values of z^p_{n+2} to the processors y^p , y^c and z^c and they wait antil the message will be received.

3- Processor y^c sends the values of y^c_{n+1} to the other processors and the process will wait for the receiving the value.

4- Processor z^c sends the values of z^c_{n+1} to the other processors.

Now all processors y^p , z^p , y^c and z^c have the necessary values to start the calculation of the next step.Since we are dealing with only stiff ODEs , the predictor mode of the method should depend on the corrector mode , hence the formula can be adapted to the following form:

First stage calculation:

$$y^c_{n+1} = y^{cc}_n + \frac{h}{6}(m^p_1 + 2m^p_2 + 2m^p_3 + m^p_4)$$

$$z^c_{n+1} = z^{cc}_n + \frac{h}{6}(S^p_1 + 2S^p_2 + 2S^p_3 + S^p_4)$$

Where

$$m_1^p = F(x_{n+1}, y_{n+1}^p, z_{n+1}^p),$$
$$S_1^p = G(x_{n+1}, y_{n+1}^p, z_{n+1}^p),$$
$$m_2^p = F(x_{n+1} - \frac{h}{2}, y_{n+1}^p - \frac{h}{2} hm_1^p, z_{n+1}^p - \frac{h}{2} S_1^p),$$
$$S_2^p = G(x_{n+1} - \frac{h}{2}, y_{n+1}^p - \frac{h}{2} hm_1^p, z_{n+1}^p - \frac{h}{2} S_1^p),$$
$$m_3^p = F(x_{n+1} - \frac{h}{2}, y_{n+1}^p - \frac{h}{2} hm_2^p, z_{n+1}^p - \frac{h}{2} S_2^p),$$
$$S_3^p = G(x_{n+1} - \frac{h}{2}, y_{n+1}^p - \frac{h}{2} hm_2^p, z_{n+1}^p - \frac{h}{2} S_2^p),$$
$$m_4^p = F(x_{n+1} - h, y_{n+1}^p - hhm_2^p, z_{n+1}^p - hS_2^p),$$
$$S_4^p = G(x_{n+1} - h, y_{n+1}^p - hm_2^p, z_{n+1}^p - hS_2^p),$$

Second stage calculation:

$$y_{n+2}^p = y_{n+1}^p + \frac{h}{6}(K_1^c + 2K_2^c + 2K_3^c + K_4^c)$$
$$z_{n+2}^p = z_{n+1}^c + \frac{h}{6}(L_1^c + 2L_2^c + 2L_3^c + L_4^c)$$
$$y_{n+1}^{cc} = y_n^{cc} + \frac{h}{6}(m_1^c + 2m_2^c + 2m_3^c + m_4^c)$$
$$z_{n+1}^{cc} = z_n^{cc} + \frac{h}{6}(S_1^c + 2S_2^c + 2S_3^c + S_4^c)$$

Where

$$K_1^c = F(x_{n+1}, y_{n+1}^c, z_{n+1}^c),\ L_1^c = G(x_{n+1}, y_{n+1}^c, z_{n+1}^c),$$
$$K_2^c = F(x_{n+1} - \frac{1}{2}h, y_{n+1}^c - \frac{1}{2}hK_1^c, z_{n+1}^c - \frac{1}{2}hL_1^c),$$
$$L_2^c = G(x_{n+1} - \frac{1}{2}h, y_{n+1}^p - \frac{1}{2}hK_1^c, z_{n+1}^c - \frac{1}{2}hL_1^p),$$

$$K_2^c = F(x_{n+1} - \frac{1}{2}h, y_{n+1}^c - \frac{1}{2}hK_1^c, z_{n+1}^c - \frac{1}{2}hL_1^c),$$
$$L_2^c = G(x_{n+1} - \frac{1}{2}h, y_{n+1}^p - \frac{1}{2}hK_1^c, z_{n+1}^c - \frac{1}{2}hL_1^p),$$
$$K_3^c = F(x_{n+1} - \frac{1}{2}h, y_{n+1}^c - \frac{1}{2}hK_2^c, z_{n+1}^c - \frac{1}{2}hL_2^c),$$
$$L_3^c = G(x_{n+1} - \frac{1}{2}h, y_{n+1}^p - \frac{1}{2}hK_2^c, z_{n+1}^c - \frac{1}{2}hL_2^p),$$
$$K_4^c = F(x_{n+1} - h, y_{n+1}^c - hK_2^c, z_{n+1}^c - hL_2^c),$$
$$L_4^c = G(x_{n+1} - h, y_{n+1}^p - hK_2^c, z_{n+1}^c - hL_2^p),$$
$$m_1^c = F(x_{n+1}, y_{n+1}^c, z_{n+1}^c),\ S_1^c = G(x_{n+1}, y_{n+1}^c, z_{n+1}^c),$$
$$m_2^c = F(x_{n+1} - \frac{h}{2}, y_{n+1}^c - \frac{h}{2}m_1^c, z_{n+1}^c - \frac{h}{2}S_1^c),$$
$$S_2^c = G(x_{n+1} - \frac{h}{2}, y_{n+1}^c - \frac{h}{2}m_1^c, z_{n+1}^c - \frac{h}{2}S_1^c),$$
$$m_3^c = F(x_{n+1} - \frac{h}{2}, y_{n+1}^c - \frac{h}{2}m_1^c, z_{n+1}^c - \frac{h}{2}S_2^c),$$
$$S_3^c = G(x_{n+1} - \frac{h}{2}, y_{n+1}^c - \frac{h}{2}m_1^c, z_{n+1}^c - \frac{h}{2}S_2^c),$$
$$m_4^c = F(x_{n+1} - h, y_{n+1}^c - hm_1^c, z_{n+1}^c - hS_2^c),$$
$$S_4^c = G(x_{n+1} - h, y_{n+1}^c - hm_1^c, z_{n+1}^c - hS_2^c),$$

In two processors computer, we can assign the calculations of first stage as follows: y^c_{n+1} to first processor and z^c_{n+1} to second processor, and the calculation of second stage as follows: y^c_{n+2} and z^c_{n+2} first processor and y^{cc}_{n+1} and z^{cc}_{n+1} to second processor.

Fig.(6): The mapping of the calculations for the two processors is given below:

P_1		P_2
1^{st} stage calculations: Calculates		1^{st} stage calculations: Calculates
1. m_1^p		1. S_1^p
sends m_1^p	$\rightarrow$	receives m_1^p
receive S_1^p	$\leftarrow$	sends S_1^p
2. m_2^p		2. S_2^p
sends m_2^p	$\rightarrow$	receives m_2^p
receive S_2^p	$\leftarrow$	sends S_2^p
3. m_3^p		3. S_3^p
sends m_3^p	$\rightarrow$	receives m_3^p
receive S_3^p	$\leftarrow$	sends S_3^p
4. m_4^p		4. S_4^p
sends m_4^p	$\rightarrow$	receives m_4^p
receive S_4^p	$\leftarrow$	sends S_4^p
5. y_{n+1}^c		5. z_{n+1}^c
sends y_{n+1}^c	$\rightarrow$	receives y_{n+1}^c
receive z_{n+1}^c	$\leftarrow$	sends z_{n+1}^c
2^{nd} stage calculations: Calculates		2^{nd} stage calculations: Calculates
1. $K_1^c, L_1^c, K_2^c, L_2^c, K_3^c, L_3^c, K_4^c, L_4^c$	$\rightarrow$ $\leftarrow$	$m_1^c, S_1^c, m_2^c, S_2^c, m_3^c, S_3^c, m_4^c, S_4^c$
2. y_{n+2}^p and z_{n+2}^p,		2. y_{n+1}^{cc} and z_{n+1}^{cc}
sends y_{n+2}^p and z_{n+2}^p	$\rightarrow$ $\leftarrow$	sends y_{n+1}^{cc} and z_{n+1}^{cc}

8.PPCGMRK Method of Order Four:

A new fourth order GM-Runge-Kutta method is of the form [81]:

$K_1 = f(x_n, y_n)$

$K_2 = f(x_n+c_2h, y_n+c_2hk_1)$,

$K_3 = f(x_n+c_3h, y_n+h(a_{31}k_1+ a_{32}k_2))$,

$K_4 = f(x_n+c_4h, y_n+h(a_{41}k_1+ a_{42}k_2+ a_{43}k_3))$,

And

$$y_{n+1} = y_n + \frac{h}{3}(\sqrt{k_1k_2} + \sqrt{k_2k_3} + \sqrt{k_3k_4}) \tag{27}$$

We know that $c_3=a_{31}+a_{32}$ and $c_4=a_{41}+a_{42}+a_{43}$, in particular by setting $a_{31}+a_{32}=1/2$, $a_{41}+a_{42}+a_{43}=1$ and by comparing the r.h.s. of equation (27) with the Taylor series expansion for $y(x_{n+1})$, the following six equation of conditions were obtained:

h^2ffy: $-192c_2+96=0$ (28a)

h^3ff^2y: $-108-48a_{43}- 24c_2-96c_2a_{42}-192c_2a_{32}+48c_2^2=0$ (28b)

h^3ff^2y: $24-96c_2^2=0$ (28c)

h^4ff^3y: $18+12a_{43}+3c_2+24c_2a_{42}-96c_2a_{32}a_{43}$
$+6c_2^2-48c_2^2a_{32}-246c_2^3=0$ (28d)

h^4f^2fyfyy: $108-60a_{43}-6c_2-96c_2a_{32}-12c_2^2$
$-48c_2^2a_{42}-96c_2^2a_{32}+48c_2^3=0$ (28e)

and

h^4f^3fyyy: $4-32_2^3=0$ (28f)

Now we shall combine equation (28) with the arithmetic mean (AM) formula of the form:

$$y_{n+1} = y_n+h (w_1 k_1 + w_2 k_2 + w_3 k_3 + w_4 k_4) \tag{29}$$

where $w_1+w_2+w_3+w_4=1$, which uses the similar function evaluation k_i (i = 1,2,3,4) as given by equation (26)

It can be established that the equation of condition for the fourth order terms are given as follows:

$$h^2ffy : w_2c_2 + w_3c_3 + w_4c_4 = 1/2 \tag{30a}$$

$$h^3ff^2y : w_3c_2a_{32} + w_4(a_{42}c_2 + a_{43}c_3) = 1/6 \tag{30b}$$

$$h^3 ff^2 yy: \frac{w_3 c_2^2}{2} + \frac{w_3 c_3^2}{2} + \frac{w_4 c_4^2}{2} = 1/6 \qquad (30c)$$

$$h^4 f^3 fyyy: \frac{w_2 c_2^3}{6} + \frac{w_3 c_3^3}{6} + \frac{w_4 c_4^3}{6} = 1/24 \qquad (30d)$$

$$h^4 ff^3 y: w_4 a_{43} c_2 a_{32} = 1/24 \qquad (30e)$$

and

h^4f^2fyfyy:

$$\frac{w_2 c_2^2 a_{32}}{2} + w_3 c_2 a_{32} c_3 + \frac{w_4 a_{42} c_2^2}{2} + \frac{w_4 a_{43} c_3^2}{6} + w_4 c_4 (a_{42} c_2 + a_{43} c_3) = 1/6 \qquad (30f)$$

In obtaining equations (28a) – (28f) we have previously set $a_{31}+a_{32}= ½$ and $a_{41}+a_{42}+a_{43}=1$, thus we still have one degree of freedom for the nine unknown variables , if we choose equation (30c) to be the ninth equation , then the resulting AM and GM formulae will be of third order when applied to a general problem of the form:

$$y'(x)= f(x,y(x)) \qquad (31)$$

Thus, by substituting $c_3= 1/2$ and $c_4=1$ where appropriate, the remaining seven equations of conditions for the AM and GM formulae are given as follows:

AM:

h^2ffy: $1/2w_2 + 1/2w_3 + w_4 = 1/2$ (32a)

h^3ff^2y: $1/3w_2a_{32} + w_4(1/2a_{42} + 1/2a_{43})= 1/6$ (32b)

h^4ff^3y: $1/2w_4a_{43}a_{32} = 1/24$ (32c)

and

GM:

h^2ffy: $-192c_2 = 96$ (32e)

h^3ff^2y: $8a_{32} + 4a_{42} + 4a_{43} = 9$ (32f)

and

h^4ff^3y: $2a_{32} – 2a_{42} - 2a_43 + 8a_{32}a_{43} = 3$ (32g)

For the purpose pf parallel computation we shall assume that $a_{43}=0$, So these equations are then solved simultaneuously, we get:

$c_2=1/2$, $a_{31}=-3/4$, $a_{32}=5/4$, $a_{42}=-1/4$, $a_{43}=0$, $w_1=1/11$, $w_2=17/33$, $w_3=10/33$, $w_4=1/11$

Thus the method is given in its final form by:

$k_1 = f(x_n, y_n)$,

$k_2 = f(x_n+1/2h, y_n+1/2hk_1)$,

$k_3 = f(x_n+1/2h, y_n+h(-3/4k_1 + 5/4k_2))$,

(33)

$k_4 = f(x_n+h, y_n+h(5/4k_1-1/4k_2))$,

and

$$y^{*}_{n+1} = y_n + \frac{h}{33}(3k_1 + 17k_2 + 10k_3 + 3k_4)\text{(AM-mode)} \quad (34)$$

$$y^{**}_{n+1} = y_n + \frac{h}{3}(\sqrt{k_1k_2} + \sqrt{k_2k_3} + \sqrt{k_3k_4})\text{(GM-mode)} \quad (35)$$

Again k_3 and k_4 are independent tasks , hence they acan be computed in parallel . Anyhow , evaluation of K_1,K_2 and (parallel evaluation of K_3 and K_4) are sequential processes which means that speed-up factor S_p of the solution will remain small ($1\leq Sp<2$) we have called these type of methods as semi-parallel explicit methods.

Since we are dealing with stiff differential equations, we need that the integration method to be implicit , so we can conert formula (34) and (35) to implicit forms by backward integration (i.e. using $-h, h>0$).

Again we see in the implicit forms that K_3 and K_4 are defined independently , so that they can be evaluated at the same time using two different processors but their parallel evaluation should be done sequentially with evaluation of K_1 and K_2 , which means that the new implicit methods are semi-parallel. We can produce parallel versions of the methods by arranging them in parallel PCRK forms.

9-Numerical Example:

Consider the second order stiff equation:

$$\frac{d^2y}{dx^2} = 1001\frac{dy}{dx} + 1000y = 0, \qquad x \in (0,1) \quad (40)$$

The general solution of (40) is

$Y(x) = Ae^{-x} + Be^{-1000X}$

If we impose the initial conditions Y(0) = 1 , y'(0)=-1, the exact solution is:

$Y(x) = e^{-x}$

We now try to solve (40) with this initial condition using the 4th order Runge-Kutta method. The ystem can be rewritten as a first-order system[23]

$$\begin{aligned} \frac{dy_1}{dx} &= y_2, \qquad y_1(0) = 1 \\ \frac{dy_2}{dx} &= -1001y_2 - 1000y_1, \quad y_2(0) = -1 \end{aligned} \quad (41)$$

Results of applications of the methods of section (2), (4), (6) and (8) for the above problem are given in tables (1), (2), (3) and (4) respectively (h=0.1).

Table (1) Results of method of section (2)

X	Corr.Y_1	Corr.Y_2	Exact Y_1	Exact Y_2	Error 1	Error 2
0.1	0.9050226	-0.9050226	0.9048374	-0.9048374	1.852×10^{-4}	-1.852×10^{-4}
0.2	08190455	-08190455	0.8187308	-0.8187308	3.147×10^{-4}	-3.147×10^{-4}
0.3	0.7412362	-0.7412362	0.74087182	-0.7408182	4.18×10^{-4}	-4.18×10^{-4}
0.4	0.6708187	-0.6708187	0.6703200	-0.6703200	4.987×10^{-4}	-4.987×10^{-4}
0.5	0.6070909	-0.6070909	0.6065307	-0.6005307	5.602×10^{-4}	-5.602×10^{-4}
0.6	0.5494173	-0.5494173	0.4588116	-0.4588116	6.057×10^{-4}	-6.057×10^{-4}
0.7	0.4972227	-0.4972227	0.4965853	-0.4965853	6.374×10^{-4}	-6.374×10^{-4}
0.8	0.4499865	-0.4499865	0.4493290	-0.4493290	6.575×10^{-4}	-6.575×10^{-4}
0.9	0.4072378	-0.4072378	0.4065697	-0.4065697	6.681×10^{-4}	-6.681×10^{-4}
1	0.3685502	-0.3685502	0.3678794	-0.3678794	6.708×10-4	-6.708×10^{-4}

TABLE (2), Result of method of section (4)

X	Corr.Y_1	Corr.Y_2	Exact Y_1	Exact Y_2	Error 1	Error 2
0.1	0.9048387	-0.9048387	0.9048374	-0.9048374	-1.3×10^{-6}	1.3×10^{-6}
0.2	08187323	-08187323	0.8187308	-0.8187308	-1.5×10^{-6}	1.5×10^{-6}
0.3	0.7408200	-0.7408200	0.7408182	-0.7408182	-1.8×10^{-6}	1.8×10^{-6}
0.4	0.6703219	-0.6703219	0.6703200	-0.6703200	-1.9×10^{-6}	1.9×10^{-6}
0.5	0.6065327	-0.6065327	0.6065307	-0.6005307	-2×10^{-6}	2×10^{-6}
0.6	0.5488137	-0.5488137	0.5488116	-0.4588116	-2.1×10^{-6}	2.1×10^{-6}
0.7	0.4965874	-0.4965874	0.9465853	-0.4965853	-2.1×10^{-6}	2.1×10^{-6}
0.8	0.4493311	-0.4493311	0.449390	-0.4493290	-2.1×10^{-6}	2.1×10^{-6}
0.9	0.465718	-0.465718	0.4065697	-0.4065697	-2.1×10^{-6}	2.1×10^{-6}
1	0.3678815	-0.3678815	0.3678794	-0.3678794	-2.1×10^{-6}	2.1×10^{-6}

TABLE (3) Results of method of section (6)

X	Corr.Y_1	Corr.Y_2	Exact Y_1	Exact Y_2	Error 1	Error 2
0.1	0.9048258	-0.9048258	0.9048374	-0.9048374	-1.16×10^{-5}	1.16×10^{-5}
0.2	08187165	-08187165	0.8187308	-0.8187308	-1.43×10^{-5}	1.43×10^{-5}
0.3	0.7408020	-0.7408020	0.74087182	-0.7408182	-1.62×10^{-5}	1.62×10^{-5}
0.4	0.6703023	-0.6703023	0.6703200	-0.6703200	-1.77×10^{-5}	1.77×10^{-5}
0.5	0.6065119	-0.6065119	0.6065307	-0.6005307	-1.88×10^{-5}	1.88×10^{-5}
0.6	0.5487922	-0.5487922	0.4588116	-0.4588116	-1.94×10^{-5}	1.94×10^{-5}
0.7	0.4965655	-0.4965655	0.4965853	-0.4965853	-1.98×10^{-5}	1.98×10^{-5}
0.8	0.4493090	-0.4493090	0.4493290	-0.4493290	-2×10^{-5}	2×10^{-5}
0.9	0.4065497	-0.4065497	0.4065697	-0.4065697	-2×10^{-5}	2×10^{-5}
1	0.3678598	-0.3678598	0.3678794	-0.3678794	-1.96×10^{-5}	1.96×10^{-5}

Table (4) Results of method of section (8)

X	Corr.Y_1	Corr.Y_2	Exact Y_1	Exact Y_2	Error 1	Error 2
0.1	0.9048123	-0.9048123	0.9048374	-0.9048374	-2.51×10^{-5}	2.51×10^{-5}
0.2	0.816941	-0.816941	0.8187308	-0.8187308	-3.67×10^{-5}	3.67×10^{-5}
0.3	0.7407724	-0.7407724	0.74087182	-0.7408182	-4.58×10^{-5}	4.58×10^{-5}
0.4	0.6702671	-0.6702671	0.6703200	-0.6703200	-5.29×10^{-5}	5.29×10^{-5}
0.5	0.6064724	-0.6064724	0.6065307	-0.6005307	-5.83×10^{-5}	5.83×10^{-5}
0.6	0.5487496	-0.5487496	0.4588116	-0.4588116	-6.2×10^{-5}	6.2×10^{-5}
0.7	0.4965207	-0.4965207	0.4965853	-0.4965853	-6.46×10^{-5}	6.46×10^{-5}
0.8	0.4492628	-0.4492628	0.4493290	-0.4493290	-6.62×10^{-5}	6.62×10^{-5}
0.9	0.4065029	-0.4065029	0.4065697	-0.4065697	-6.68×10^{-5}	6.68×10^{-5}
1	0.3678127	-0.3678127	0.3678794	-0.3678794	-6.67×10^{-5}	6.67×10^{-5}

Corr. = Corrected, $Error_1$= Corr. Y_1-Exact Y_1,
$Error_2$=Corr. Y_2-Exact Y_2

In each table, the first column gives values of the independent variable x, the second and the third columns give the corresponding corrected values of y_1 and y_2 respectively computed by implicit mode of the method being used, the fourth and the fifth columns give the corresponding values y_1 and y_2 computed from the exact solution of the problem, the sixth and seventh columns give the corresponding errors of the numerical solutions of y_1 and y_2 respectively. The two last columns of each table make clear the effectiveness of the newly developed methods in solving stiff ODEs since we are using fixed large stepsize and the integration method is of fixed order.

11- Conclusion

We have developed several semi-parallel explicit Rung-Kutta method for solving ODEs using AM and GM techniques. Then we converted these methods into SPIRK methods by converting the direction of integration. SPIRK methods are suitable for integrating stiff ODEs. Also we have developed PPCRK methods by advancing the predictor mode one-step, and we have modified the PPCRK methods for a system of stiff ODEs.

REFERENCES

[1] S. D. Contc and C. de Boor, "Elementary numerical analysis a algorithm approach"; International Student Edition, London, 1981.

[2] D.J. Evans and B. V. Sanugi, "A parallel Runge-Kutta integration method", Parallel computing 11 (1989), 245-251.

[3] M.A. Franklin, "Parallel solution of ordinary differential equations", IEEE Trans. On Computers Vol. C-27, No.5(1978), 413-420.

[4] C.E. Froberg, "Introduction to numerical analysis", Addison-Wesley, 1969.

[5] C.W. Gear, "Rubg-Kutta started for multistep methods", ACM Trans. Math. Software, Vol. 6 (1980) 263-279.

[6] S.K. Goshal, M. Gupta and V. Rajarman, "A parallel multisrep predictor-corrector algorithm for solving ordinary differential equations", J. of Parallel and Distributed Computing, 6 (1989), 636-648.

[7] W. L. Miranker, "A survey of parallelism in numerical analysis", SIAM Review, Vol. 13, No.4 (1971), 547.

[8] Murshid A.A.A. (2000); An Investigation of Numerical Algorithms for solcing stiff ODEs suitable for parallel computers, Ph. D. Thesis, Mosul University.

[9] A. Sandu et al, "Benchmarking stiff ODEs solvers for atmospheric chemistry problems-I: implicit vs explicit", Atmospheric environment, Vol. 31, No. 12 (1988), 991-995.