International Journal of Latest Engineering and Management Research (IJLEMR)
ISSN: 2455-4847
www.ijlemr.com || Volume 09 – Issue 03 || March 2024 || PP. 58-63

Techniques in Implementing Runge –Kutta Method

Pham Thi Thuy[1]
[1]*University of Education, Thai Nguyen University, Vietnam*

Abstract: The paper aims to do a survey on the techniques used to implement implicit and explicit Runge – Kutta methods to approximate an initial value problem. This is helpful for the problem of high stiffness which appears in many applications, but it challenges a numerical approximation technique to solve. In fact, this problem can be solved by several approaches, such as the predictor-corrector approach, or an adaptive step-size by pair of different order methods. Numerical experiments prove the efficiency of these approaches.
Keywords: Runge – Kutta, Implementation, predictor-corrector, adaptive step-size, stiffness.

I. INTRODUCTION

Let consider the ODE with initial condition

$$y' = f(t,y), a \le t \le b, y(a) = \alpha. \quad (1)$$

A Runge-Kutta method of order p is given by

$$w_{n+1} = w_n + \sum_{i=1}^{s} b_i k_i \quad (\forall n, 0 \le n \le N)(2)$$

$$k_r = hf\left(t_n + c_j h, w_n + \sum_{r=1}^{s} a_{jr} k_r\right), \forall r = 1,2,\dots,s,$$

$$h = \frac{b-a}{N}, a = t_0 < t_1 < \dots < t_N = b, w_n \approx y(t_n), \forall 1 \le n \le N.$$

The Butcher's table ([2]) is

c_1	a_{11}	a_{12}	...	a_{1s}
c_2	a_{21}	a_{22}	...	a_{2s}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
c_s	a_{s1}	a_{s2}	...	a_{ss}
	b_1	b_2	...	b_s

In Sections II and III below, we introduce two approaches for implementing implicit and explicit Runge-Kutta method where matrix $A = \left(a_{ij}\right)_{1 \le i,j \le s}$ in Butcher's table is upper triangular or not.

II. PREDICTOR – CORRECTOR APPROACH

The equivalent form of (2) in matrix block is given by

$$w_{n+1} = w_n + \mathbf{U}\mathbf{k}$$
$$\mathbf{k} = h\boldsymbol{F}(t_n\mathbf{1} + h\mathbf{Q}, w_n\mathbf{1} + A\mathbf{k}) \quad (3)$$

where

$$\mathbf{U} = (b_1, b_2, \dots, b_s), \mathbf{Q} = (c_1, c_2, \dots, c_s)^T,$$
$$\mathbf{1} = (1, \dots, 1)^{\mathrm{T}}, \mathbf{k} = (k_1, k_2, \dots, k_s)^{\mathrm{T}} \in \mathbb{R}^s,$$
$$\boldsymbol{F}(\boldsymbol{z}, \boldsymbol{u}) = \boldsymbol{F}((z_1, z_2, \dots, z_s)^T, (u_1, u_2, \dots, u_s)^T) = \left(f(z_1, u_1), f(z_2, u_2), \dots, f(z_s, u_s)\right)^T.$$

The Newton method is a not bad option to formulate **k** in (3) assuming the differentiable of $\boldsymbol{F}$. However, the cost of derivative computation is an obstacle of this approach. We then improve the performance by only solving **k** using the Newton method and then using the predictor-corrector approach toconstruct**k** recurrently for other repeatation at the current time step t_n as

$$\mathbf{k}^{(q+1)} = h\boldsymbol{F}\left(t_n\mathbf{1} + h\mathbf{Q}, w_n\mathbf{1} + A\mathbf{k}^{(q)}\right), \forall q \ge 0, \quad (4)$$

This approach works properly since the sequence in (4) converges when $\boldsymbol{F}$ is differentiable. For implicit Runge-Kutta method, the method (3) with s-stage and p-order has $s < p$. So it benefits the amount of computation. This helpsin reducing the round of error and saving time by an efficient computation.

For illustrations, consider the following cases in which a class of implicit Runge-Kutta method formed by using the Gaussian quadrature in [3, 4].

Case of $s = 2, p = 4$:The Butcher's table is

$$\begin{array}{c|cc} \frac{1}{2}-\frac{\sqrt{3}}{6} & \frac{1}{4} & \frac{1}{4}-\frac{\sqrt{3}}{6} \\ \frac{1}{2}+\frac{\sqrt{3}}{6} & \frac{1}{4}+\frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \qquad (5)$$

Case of$s = 3, p = 6$:The Butcher's table is

$$\begin{array}{c|ccc} \frac{1}{2}-\frac{\sqrt{15}}{10} & \frac{5}{36} & \frac{2}{9}-\frac{\sqrt{15}}{15} & \frac{5}{36}-\frac{\sqrt{15}}{30} \\ \frac{1}{2} & \frac{5}{36}+\frac{\sqrt{15}}{24} & \frac{2}{9} & \frac{5}{36}-\frac{\sqrt{15}}{24} \\ \frac{1}{2}+\frac{\sqrt{15}}{10} & \frac{5}{36}+\frac{\sqrt{15}}{30} & \frac{2}{9}+\frac{\sqrt{15}}{15} & \frac{5}{36} \\ \hline & \frac{5}{18} & \frac{4}{9} & \frac{5}{18} \end{array} \qquad (6)$$

The algorithm below shows the procedure step by step.
ALGORITHM: Predictor-Corrector approach.

Input: $\boldsymbol{F}, \mathbf{Q}, \mathbf{U}, A, a, b,$the initial value $\alpha,$and number of time-steps$N,$ the maximum iteration in the predictor-corrector process $mmax$.

Output: the approximation w_n for $0 \le n \le N$.
Step 1. Set step size $h \coloneqq (b-a)/N$, and initiate $n = 0$: Take $t_0 \coloneqq a, w_0 \coloneqq \alpha$.

Step 2.While $n \le N$ do:
Compute

$$B \coloneqq I_s - h\boldsymbol{F_u}(t_n\mathbf{1} + h\mathbf{Q}, w_n\mathbf{1})A.$$
$$P \coloneqq h\boldsymbol{F}(t_n\mathbf{1} + h\mathbf{Q}, w_n\mathbf{1})\mathbf{1} + h^2\boldsymbol{F}(t_n\mathbf{1} + h\mathbf{Q}, w_n\mathbf{1})\mathbf{U}.$$

Set $\boldsymbol{\theta}_{\mathbf{1}}^{(n)} \coloneqq \mathbf{k}^{(n)}$to be the solution $\mathbf{k}^{(n)}$of the system $B\mathbf{k}^{(\boldsymbol{n})} = P$.

For i from 1 to $(mmax - 1)$,set$\boldsymbol{\theta}_{\mathrm{i}+1}^{(n)} \coloneqq h\boldsymbol{F}\left(t_n\mathbf{1} + h\mathbf{Q}, w_n\mathbf{1} + A\boldsymbol{\theta}_{\mathrm{i}}^{(n)}\right)$.
Set $n \coloneqq n+1, t_n \coloneqq t_n + h, w_n \coloneqq w_n + \mathbf{U}\boldsymbol{\theta}_{mmax}^{(n)}$.

Step 3. Get output $\{(t_n, w_n)\}_{0 \le n \le N}$.

In this algorithm, the Newton method performs the first step at each time step$t_n,$ the predictor-corrector method then is applied in the steps follow to improve the exactness of the approximation at the current time step. This combination works properly and is efficient in reducing the computational cost and getting more and more accurate approximation. The proof of advantages for the implementation is shown in Numerical experiments below. Here, IRK6_PC and IRK4_PC are method of orders 6 and 4, respectively, for the predictor-corrector approach. The comparison is made with methods the implicit Runge-Kutta or order 6, IRK6, and normal Runge-Kutta method of order 4, RK4, and order 6, RK6. The efficiency is shown in the performance of the method.

Example 1([1]).The initial value problem

$$y' = \left(\frac{1}{t} - 40\right)y + 40t^2 + t, t \in [\ln 2\,, 5], y(\ln 2) = \frac{\ln 2}{2^{40}} + \ln^2 2.$$

has the exact solution of the problem is $y = t^2 + te^{-40t}$. The error at $t_N = 5$ is shown in table below for each method. The stiffness is very high since the term te^{-40t} having a very large negative coefficent in power. This causes challenge a usual method, especialy when the step-size is moderately small. In the table, we see tremendous error in calculation, such as method RK4 and RK6 even when the step size is very small, but the stable is clearer and clearer for the other methods, especially the class of the predictor-corrector methods.

IRK6_PC $N = 10,30,40,70;$ $mmax = 10$	**IRK4_PC** $N = 10,30,40,70;$ $mmax = 10$	**IRK6** $N = 10,20,30,40,70;$ $mmax = 10, tol = 0.001$	**RK4** $N = 10,30,40;$	**RK6** $N = 10,40,70;$
7.7×10^{33}, 1.31×10^{-1}, 3.698×10^{-3}, 3.483×10^{-6}	2.35×10^{39}, 5.12×10^{-1}, 1.62×10^{-2}, 1.964×10^{-5}	1.324×10^{-1}, 3.46×10^{-2}, 1.443×10^{-2}, 7.486×10^{-3}, 2.021×10^{-3}	2.143×10^{32}, 1.167×10^{39}, 2.574×10^{30}, 2.895×10^{-3}	3.34×10^{51}, 2.97×10^{45}, 2.915×10^{-2}
$2.5s, 4.9s, 5.84s, 9.21s$	$2.47s, 5.08s, 5.8s, 9.3s$	$3.1s, 4.3s, 5.3s, 6.3s, 9.9s$	$1.33s, 1.4s, 1.42s, 1.43s$	$1.2s, 1.35s, 1.36s$

Exmple 2([2])**.** The initial value problem

$$y' = -10y + 10\cos t - \sin t, t \in [0,4], y(0) = 2.$$

The exact solution of the problem is $y = \cos(t) + e^{-10t}$. The absolute error at $t_N = 4$ is shown in table below for each method. This problem has a high stiffness since the term e^{-10t} having a large negative coefficent in power. Similar to Example 1, result also shows that the efficiency in the class of the precdictor-corrector methods.

IRK6_PC $N = 10,20,30,70;$ $mmax = 10$	**IRK4_PC** $N = 10,20,30;$ $mmax = 10$	**IRK6** $N = 10,20,30;$ $mmax = 10, tol = 0.001$	**RK4** $N = 10,20,30;$	**RK6** $N = 10,20,30;$
1.004×10^{-2}, 1.538×10^{-6}, 3.801×10^{-9}	4.24×10^{-2}, 1.628×10^{-5}, 5.123×10^{-6}	3.972×10^{-2}, 7.104×10^{-3} 2.727×10^{-3}	9.517×10^{6}, 3.982×10^{-3}, 4.607×10^{-4}	1.275×10^{10}, 3.065×10^{-4}, 6.712×10^{-4}
$0.83s, 1.16s, 1.55s$	$0.81s, 1.18s, 1.53s$	$1.14s, 1.37s, 1.72s$	$0.45s, 0.41s, 0.42s$	$0.42s, 0.44s, 0.46s$

Example 3([5])**.** Given the initial value problem

$$y' = (t + 2t^3)y^3 - ty, t \in [0,2], y(0) = 1/3.$$

The exact solution of the problem is $y = \left(3 + 2t^2 + 6e^{t^2}\right)^{-1/2}$. The absolute error at $t_N = 2$ is shown in table below for each method. The stiffness of the problem is not too small in comparing to that in two previous examples. However, the efficiency and the advantages still appear for the class of the predictor-corrector methods.

IRK6_PC $N = 10,20,30,70;$ $M = 10$	**IRK4_PC** $N = 10,20,30;$ $M = 10$	**IRK6** $N = 10,20,30;$ $M = 10, tol = 0.001$	**RK4** $N = 10,20,30;$	**RK6** $N = 10,20,30;$
1.915×10^{-9}, 2.978×10^{-11}, 2.612×10^{-12}, 1.6×10^{-14}	1.82×10^{-7}, 1.064×10^{-8}, 2.075×10^{-9}	1.464×10^{-3}, 3.628×10^{-4} 1.606×10^{-4}	6.458×10^{-6}, 3.73×10^{-7}, 7.16×10^{-8}	1.982×10^{-4}, 1.033×10^{-4}, 6.991×10^{-5}
$0.92s, 1.27s, 1.59s, 9.1s$	$0.88s, 1.25s, 1.6s$	$0.98s, 1.33s, 1.69s$	$0.48s, 0.5s, 0.51s$	$1.52s, 1.58s, 1.6s$

III. ADAPTIVE APPROACH

We consider a Runge - Kutta method of order 1:

$$w_1 = w_0 + b_1 h F_1, \qquad (7)$$

Where $w_0 = \alpha, F_1 = f(t_0, w_0)$, and the coefficient b_1 is selected appropriately. Another Runge -Kutta method of order 2 is applied

$$w_1^* = w_0 + h(b_1^* F_1 + b_2^* F_2), \qquad (8)$$

Where $F_2 = f(t_0 + c_2 h, w_0 + a_2 h)$, and coefficients b_1^*, b_2^*, c_2, a_2 are selected such that (7) and (8) are identical to their Taylor's approximation of the corresponding the orders 1 and 2. We have

$$y(t_0 + h) = w_0 + hf(t_0, w_0) + \frac{h^2}{2} f'(t_0, w_0) + \frac{h^3}{6} f''(\xi, y(\xi)), \xi \in (t_0, t_0 + h),$$

$$f'(t_0, w_0) = f_t(t_0, w_0) + f_y(t_0, w_0) y'(t_0) = f_t(t_0, w_0) + f_y(t_0, w_0) f(t_0, w_0).$$

By Taylor's expansion up to order 2,

$$F_2 = f(t_0 + c_2 h, w_0 + a_2 h) = f(t_0, w_0) + h\big(c_2 f_t(t_0, w_0) + a_2 f_y(t_0, w_0)\big) + o(h^2).$$

So, $w_1^* = w_0 + h(b_1^* + b_2^*) f(t_0, w_0) + h^2 b_2^* [c_2 f_x(t_0, w_0) + a_2 f_y(t_0, w_0)]$. (9)

Identifying (8) and (9) yields

$$\begin{cases} b_1^* + b_2^* = 1 \\ b_2^* c_2 = \dfrac{1}{2} \\ b_2^* a_2 = \dfrac{1}{2} f(t_0, y_0) \end{cases}$$

This system has many solutions for b_1^*, b_2^*, c_2, and a_2. Each provides us a method in the class we aims to construct. The local truncation error of (7) and (8), respectively, are

$$\tau_1(h) = \frac{y(t_0 + h) - w_1}{h} = O(h),$$

$$\tau_1^*(h) = \frac{y(t_0 + h) - w_1^*}{h} = O(h^2).$$

So,

$$\tau_1(h) = (y(t_0 + h) - w_1)/h = \frac{1}{h}[(y(t_0 + h) - w_1^*) + (-w_1 + w_1^*)] = \tau_1^*(h) + \frac{1}{h}(w_1^* - w_1).$$

Since $\tau_1^*(h)$ has a high order than $\tau_1(h)$, $\tau_1(h) \approx (w_1^* - w_1)/h$.

There is a constant M such that $\tau_1(h) \approx Mh$. If the step size is adjusted to $qh, q \le 1$,then

$$\tau_1(qh) \approx M(qh) \approx \frac{q}{h}(w_1^* - w_1).$$

If we require a bound of error $\varepsilon > 0$ that $|\tau_1(h)| \le \varepsilon$, the coefficient for adjustment the step size is

$$q \le \frac{\varepsilon h}{|w_1^* - w_1|}.$$

The following algorithm describes this procedure of the method.

ALGORITHM: Adaptive step size approach

Input: $f, a, b, \alpha, N, \varepsilon, hmin$ (minimum step size allow), and $hmax$ (maximum step size allow).

Output: Sequence of approximation $\{(t_n, w_n)\}_{0 \le n \le N}$.

Step 1. Initiate the procedure.
$n := 0, t_n := a; w_n := \alpha, h = hmax$.
OUTPUT (t_n, w_n, h).

Step 2. Repeat steps 3-9 until a break.

Step 3.$F_1 := hf(t_n, w_n)$;
$F_2 := hf(t_n + h, w_n + F_1)$;

Step 4.$R := \frac{1}{h}|F_1 - F_2|$.

Step 5. If $R < \varepsilon$ then
$n := n + 1$;
$t_n := t_n + h$;
$w_n := w_n + F_1$;
OUTPUT (t_n, w_n, h).

Step 6.$q := 0.5\varepsilon / R$.

Step 7. If $q \le 0.1$ then $h := 0.1h$;
elseif $q \ge 4$ then $h := 4h$;
else $h := qh$;

Step 8. If $h > hmax$ then $h := hmax$;

Step 9. If $x \geq b$ then break;
elseif $x + h > b$ then $h := b - x$;
elseif $h < hmin$ then
OUTPUT("Minimum *h* exceeded. Procedure fails!")
Break.

STOP.

Adaptive time step method produces an approximation with very small error but with low cost of computation. The efficiency of the method is proved in the numerical experiment below.

Example 4([6]).Consider the IVP: $y' = y - x^2 + 1, 0 \leq x \leq 1.5, y(0) = 0.5$.
Take the tolerance $\varepsilon = 0.06$, and $hmax = 0.25, hmin = 0.001$. The exact solution of the IVP is

$$y(x) = (x+1)^2 - \frac{e^x}{2}.$$

x_i	$y(x_i)$	h_i	w_i by Euler method	Absolute Error $\lvert y(x_i) - w_i \rvert$	Estimate R_i of $\tau_{i+1}(h)$
0.0000	0.500000000	0.250000000	0.500000000	0.000000000	0.000000000
0.0538	0.582794479	0.053760000	0.580640000	0.002154479	0.038874931
0.1002	0.657785768	0.046465317	0.653950647	0.003835121	0.033077835
0.1474	0.737159458	0.047198815	0.731541041	0.005618417	0.032950817
0.1956	0.821355070	0.048128707	0.813831849	0.007523220	0.032891885
0.2447	0.910691785	0.049164849	0.901128510	0.009563276	0.032825413
0.2950	1.005543717	0.050325000	0.993789002	0.011754715	0.032748526
0.3467	1.106357364	0.051633470	1.092240543	0.014116821	0.032658685
0.3998	1.213672443	0.053121691	1.196999503	0.016672940	0.032552499
0.4546	1.328151023	0.054831085	1.308699260	0.019451763	0.032425305
0.5114	1.450619466	0.056817490	1.428130294	0.022489172	0.032270529
0.5706	1.582130720	0.059158240	1.556299747	0.025830973	0.032078608
0.6326	1.724060435	0.061963938	1.694523328	0.029537107	0.031835113
0.6980	1.878262202	0.065399118	1.844573775	0.033688427	0.031517273
0.7677	2.047332893	0.069720829	2.008934690	0.038398203	0.031087073
0.8430	2.235100017	0.075356719	2.191266978	0.043833039	0.030476336
0.9261	2.447605733	0.083080387	2.397351338	0.050254395	0.029550478
1.0206	2.695371328	0.094465512	2.637259976	0.058111352	0.028005519
1.1339	2.999727117	0.113336275	2.931441691	0.068285426	0.024998301
1.2863	3.417376961	0.152334306	3.334464986	0.082911975	0.017172727
1.5000	4.009155465	0.213738253	3.907282585	0.101872880	0.011356559

The result shows the errors of the approximation obtained from the method with the time-step changes adaptively to fulfill the tolerance. The error is small and could prove the correction of the method.

IV. CONCLUSION

Two approaches introduced in this paper are effective and advantageous, especially in treating the problem with high stiffness. The benefits from class of implicit Runge-Kutta techniques are developed and improved with these approaches. Numerical experiments show that these approaches work properly and efficient. They also represent the advance of the predictor-corrector class of method to highly stiff problems.

V. ACKNOWLEDGEMENTS

This work is a part of the research project supported by University of Education – Thai Nguyen University.

REFERENCES

[1]. Butcher, John. C., Practical Runge - Kutta methods for scientific computation, *The ANZIAM Journal*, 50(03), 2009, 333-342, doi: 10.1017/S1446181109000030.

[2]. Butcher, John. C., *Numerical Methods for Differential Equations and Applications, 2nd Edition* (John Wiley Chichester, 2008).
[3]. Richard L. Burden, J. Douglas Faires, *Numerical Analysis*, 9th Edition (Brooks/Cole, Cengage Learning, 2011)**.**
[4]. Luther, H. A., An explicit sixth-order Runge-Kutta formula, *Mathematics of Computation,* 22, 1968, 434-436.
[5]. Nasarudin A.A., Ibrahim Z.B., Rosali H., On the Integration of Stiff ODEs Using Block Backward Differentiation Formulas of Order Six, Symmetry 2020, 12, 952, doi: 10.3390/sym12060952.
[6]. John C. Butcher, *Numerical Methods for Ordinary Differential Equations*, 2nd Edition, (John Wiley & Sons, Ltd, 2008).