Mathematics and Computer Science
2019; 4(3): 68-75
http://www.sciencepublishinggroup.com/j/mcs
doi: 10.11648/j.mcs.20190403.12
ISSN: 2575-6036 (Print); ISSN: 2575-6028 (Online)

A New Seventh Order Runge-kutta Family: Comparison with the Method of Butcher and Presentation of a Calculation Software

Hippolyte Séka, Assui Richard Kouassi

Department of Mathematics and Informatics, Institut National Polytechnique Houphouet-Boigny, Yamoussoukro, Ivory Coast

Email address:
hippolyte.seka@inphb.ci (H. Séka), r.assui@yahoo.fr (A. R. Kouassi)

To cite this article:
Hippolyte Séka, Assui Richard Kouassi. A New Seventh Order Runge-kutta Family: Comparison with the Method of Butcher and Presentation of a Calculation Software. *Mathematics and Computer Science*. Vol. 4, No. 3, 2019, pp. 68-75.
doi: 10.11648/j.mcs.20190403.12

Received: September 5, 2019; **Accepted**: September 24, 2019; **Published**: October 14, 2019

Abstract: This paper is in the context of the numerical resolution of ordinary differential equations. Most equations are unsolved in the analytic aspect. The goal is to find among the existing methods, the best method of numerical resolution. Also to facilitate the implementation of methods by introducing a calculation software. To do this, we use the Runge-Kutta method which is one of the best methods of numerical resolutions. That is why a family of Runge-Kutta methods of order 7 is presented. This family depends on the parameter b8 and contains the well known method of Butcher [8] (b8 =77/1440). To obtain convincing results, we compare methods according to the values of b8 with those of Butcher. The stability region is also studied to essentially perceive the numerical behavior that manifests itself when the steps of discretization tend to 0. The study shows that the stability region of this method does not depend on the coefficient b8. To get the values of b8, Java programming is used. Finally, to facilitate the implementation of the resolution, very simple software for numerical resolution of the ordinary differential equations is given. This software is designed for all students, also for all those who have no basis in numerical analysis and java programming to be able to find a solution approached with error estimate to an ordinary differential equation.

Keywords: Ordinary Differential Equation, Runge-Kutta, Stability Region, Java Programming

1. Introduction

The RK method is one of the best methods for numerically solving ODE. Several method of order ≥ 4 have been discovered. [1, 3, 4-7]. However the search for better methods is always up to date. In this paper, a new RK7 depending on the coefficients b_8 is presented. We show that this family contains the Butcher method [8]. The stability region is independent of b_8. By java programming, using an example we determine some values of b8 for which methods are better compared to that of Butcher [8]. Finally, a presentation of calculation software in Java programming is made to determine an approximate value of the ordinary differential equation. This paper will be organized by the following plan: section 2 RK7 method for ODEs, section 3 Stability region, section 4 Numerical analysis, section 5 The RK7 java calculator, section 6 conclusion.

2. RK7 Method for ODEs

Let's solve the following ODE.

$$y' = f(x, y(x)), \tag{1}$$

with $x_0 \leq x \leq x_n$ and the initial condition $y(x_0) = y_0$. Consider the nodes:

$$x_0, x_1 = x_0 + h, x_2 = x_0 + 2h, \ldots, x_n = x_0 + nh, \tag{2}$$

where h is the fixed step size. The goal is to determine $y_1, y_2, y_3, \cdots, y_n$ using the new RK7 family method as defined by Butcher tableau (see Figure 1).

0									
$\frac{1}{6}$	$\frac{1}{6}$								
$\frac{1}{3}$	0	$\frac{1}{3}$							
$\frac{1}{2}$	$\frac{1}{8}$	0	$\frac{3}{8}$						
$\frac{2}{11}$	$\frac{148}{1331}$	0	$\frac{150}{1331}$	$\frac{-56}{1331}$					
$\frac{2}{3}$	$\frac{-404}{243}$	0	$\frac{-170}{27}$	$\frac{4024}{1701}$	$\frac{10648}{1701}$				
$\frac{6}{7}$	$\frac{2466}{2401}$	0	$\frac{1242}{343}$	$\frac{-19176}{16807}$	$\frac{-51909}{16807}$	$\frac{1053}{2401}$			
0	$\frac{1}{576b_8}$	0	0	$\frac{1}{105b_8}$	$\frac{-1331}{279552b_8}$	$\frac{-9}{1024b_8}$	$\frac{343}{149760b_8}$		
1	$\frac{-71}{32}+\frac{-270b_8}{11}$	0	$\frac{-195}{22}$	$\frac{32}{7}$	$\frac{29403}{3584}$	$\frac{-729}{512}$	$\frac{1029}{1408}$	$\frac{270b_8}{11}$	
	$\frac{77}{1440}-b_8$	0	0	$\frac{32}{105}$	$\frac{1771561}{6289920}$	$\frac{243}{2560}$	$\frac{16807}{74880}$	b_8	$\frac{11}{270}$

Figure 1. *Butcher Tableau for RK7 method family depending on parameter b_8.*

and

$$y_{i+1} = y_i + \mathrm{h}\left(\left(\frac{77}{1440}\right) - b_8\right)k_1 + \mathrm{h}\left(\frac{32}{105}\right)k_4 + \mathrm{h}\left(\frac{1771561}{6289920}\right)k_5 + \mathrm{h}\left(\frac{243}{2560}\right)k_6 + \mathrm{h}\left(\frac{16807}{74880}\right)k_7 + \mathrm{h}(b_8)k_8 + \mathrm{h}\left(\frac{11}{270}\right)k_9 \quad (3)$$

with

$$k_1 = f(x_i, y_i) \quad (4)$$

$$k_2 = f\left(\left(\frac{1}{6}\right)\mathrm{h} + x_i, \left(\frac{1}{6}\right)\mathrm{h}k_1 + y_i\right) \quad (5)$$

$$k_3 = f\left(\left(\frac{1}{3}\right)\mathrm{h} + x_i, \left(\frac{1}{3}\right)\mathrm{h}k_2 + y_i\right) \quad (6)$$

$$k_4 = f\left(\left(\frac{1}{2}\right)\mathrm{h} + x_i, \left(\frac{1}{8}\right)\mathrm{h}k_1 + \left(\frac{3}{8}\right)\mathrm{h}k_3 + y_i\right) \quad (7)$$

$$k_5 = f\left(\left(\frac{2}{11}\right)\mathrm{h} + x_i, \left(\frac{148}{1331}\right)\mathrm{h}k_1 + \left(\frac{150}{1331}\right)\mathrm{h}k_3 - \left(\frac{56}{1331}\right)\mathrm{h}k_4 + y_i\right) \quad (8)$$

$$k_6 = f\left(\left(\frac{2}{3}\right) + x_i, \left(-\frac{404}{243}\right)\mathrm{h}k_1 - \left(\frac{170}{27}\right)\mathrm{h}k_3 + \left(\frac{4024}{1701}\right)\mathrm{h}k_4 + \left(\frac{10648}{1701}\right)\mathrm{h}k_5 + y_i\right) \quad (9)$$

$$k_7 = f\left(\left(\frac{6}{7}\right) + x_i, \left(\frac{2466}{240}\right)\mathrm{h}k_1 + \left(\frac{1242}{343}\right)\mathrm{h}k_3 - \left(\frac{19176}{16807}\right)\mathrm{h}k_4 - \left(\frac{51909}{16807}\right)\mathrm{h}k_5 + \left(\frac{1053}{2401}\right)\mathrm{h}k_6 + y_i\right) \quad (10)$$

$$k_8 = f\left(x_i, \left(\frac{1}{576b_8}\right)\mathrm{h}k_1 + \left(\frac{1}{105b_8}\right)\mathrm{h}k_4 - \left(\frac{1331}{279552b_8}\right)\mathrm{h}k_5 - \left(\frac{9}{1024b_8}\right)\mathrm{h}k_6 + \left(\frac{343}{149760b_8}\right)\mathrm{h}k_7 + y_i\right) \quad (11)$$

$$k_9 = f\left(h + x_i, \left(\left(-\frac{71}{32}\right) - \left(\frac{270b_8}{11}\right)\right)\mathrm{h}k_1 - \left(\frac{195}{22}\right)\mathrm{h}k_3 + \left(\frac{32}{7}\right)\mathrm{h}k_4 + \left(\frac{29403}{3584}\right)\mathrm{h}k_5 - \left(\frac{729}{512}\right)\mathrm{h}k_6 + \left(\frac{1029}{1408}\right)\mathrm{h}k_7 + \left(\frac{270b_8}{11}\right)\mathrm{h}k_8 + y_i\right) \quad (12)$$

If $b_8 = \frac{77}{1440}$ then we find the particular method of Runge-Kutta of order 7 defined in the book [8] on page 196.

3. Stability Region

The determination of the stability region has been developed in the literature. [9-11]. Thus the following differential equation is used:

$$y' = -\lambda y(t) \quad (13)$$

with $y(0)$ and $\lambda \in \mathrm{C}$ whose solution is $y(t) = y(0)e^{-\lambda t}$.

Let's study the case of the scheme (13):

$$k_1 = -\lambda y_i \quad (14)$$

$$k_2 = \frac{\lambda}{6} y_i(h\lambda - 6) \tag{15}$$

$$k_3 = \frac{-\lambda}{18} y_i(h^2\lambda^2 - 6h\lambda + 18) \tag{16}$$

$$k_4 = \frac{\lambda}{48} y_i(h^3\lambda^3 - 6h^2\lambda^2 + 24h\lambda - 48) \tag{17}$$

$$k_5 = \frac{\lambda}{7986} y_i(7h^4\lambda^4 + 8h^3\lambda^3 - 132h^2\lambda^2 + 1452h\lambda - 7986) \tag{18}$$

$$k_6 = \frac{-\lambda}{1458} y_i(8h^5\lambda^5 + 81h^4\lambda^4 - 72h^3\lambda^3 + 324h^2\lambda^2 - 972h\lambda + 1458) \tag{19}$$

$$k_7 = \frac{\lambda}{43218} y_i(104h^6\lambda^6 + 1170h^5\lambda^5 + 225h^4\lambda^4 + 4536h^3\lambda^3 - 15876h^2\lambda^2 + 37044h\lambda - 43218) \tag{20}$$

$$k_8 = \frac{-\lambda y_i(h^7\lambda^7 + 20h^6\lambda^6 + 90h^5\lambda^5 + 181440b_8)}{181440b_8} \tag{21}$$

$$k_9 = \frac{\lambda}{7392} y_i(h^8\lambda^8 + 7h^7\lambda^7 - 114h^6\lambda^6 - 666h^5\lambda^5 - 812h^4\lambda^4 + 1232h^3\lambda^3 - 3696h^2\lambda^2 + 7392h\lambda - 7392) \tag{22}$$

Using (3), we obtain:

$$y_{i+1} = \frac{1}{181440} y_i(h^9\lambda^9 + 6h^8\lambda^8 - 36h^7\lambda^7 + 252h^6\lambda^6 - 1512h^5\lambda^5 + 7560h^4\lambda^4 - 30240h^3\lambda^3 + 90720h^2\lambda^2 - 181440h\lambda + 181440) \tag{23}$$

Let

$$z = h\lambda \tag{24}$$

We obtain

$$y_n = \left(\frac{1}{181440}\right)^n y_0(z^9 + 6z^8 - 36z^7 + 252z^6 - 1512z^5 + 7560z^4 - 30240z^3 + 90720z^2 - 181440z + 181440)^n \tag{25}$$

We write $z = x + iy$, the stability region such as $\left|\frac{y_n}{y_0}\right| < 1$. (see Figure 2):

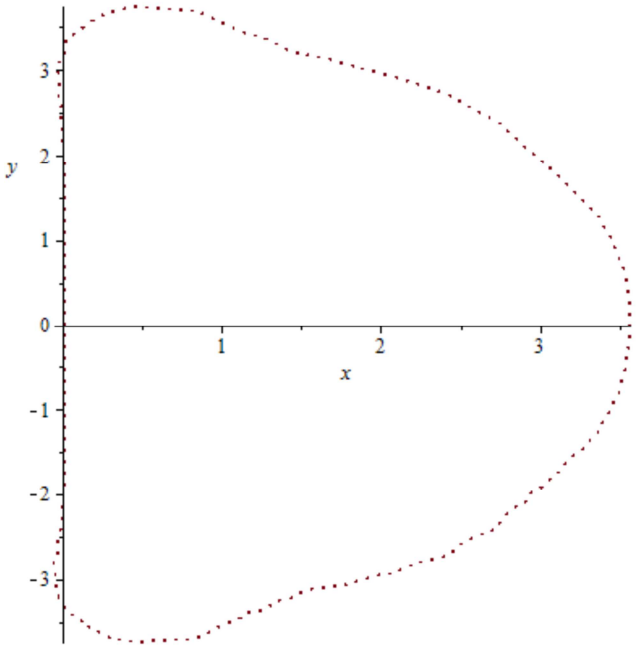

Figure 2. *Family stability region of the method of order 7.*

Note that the stability region does not depend on b_8.

4. Numerical Analysis

4.1. Example

Let us use the numerical example of the publication [12] illustrated in Figure 2.

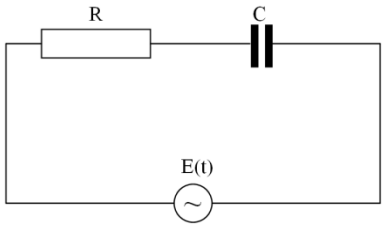

Figure 3. *RC_Circuit.*

The governing first order ODE is given by

$$\mathrm{R}\frac{di}{dt} + \frac{\mathrm{i}}{C} = \frac{\mathrm{d(E(t))}}{dt} \tag{26}$$

Where R is the resistance (ohms), C is the capacitance (farads), i is the current (ampere), and E(t) is the voltage (volts). Given E(t) = sin{(100t) volts, R = 5 ohms, C = 0.1 farads and at the initial time t = 0 the initial current is i = 0. We want to solve the differential equation (2), which is the RC-circuit ODE, for the time interval $0 \leq t \leq 0.05$ seconds with the time step size t = 0.01 seconds using the RK7 family method. If the exact solution is given by

$$\mathrm{i} = \left(\frac{10}{2501}\right)\cos(100t) + \left(\frac{500}{2501}\right)\sin(100t) - \left(\frac{10}{2501}\right)e^{-2t}. \quad (27)$$

Then, the absolute errors could be calculated at each iteration. The computation procedure of the RK7 family method is summarized as follows [12]:

Rewrite the ODE in (13) by substituting the given values to yield

$$\frac{di}{dt} = 20\cos(100\mathrm{t}) - 2\mathrm{i} = f(t,i). \quad (28)$$

By using the RK7 family method, formulate the solution of (15) into the form of (3), we get

$$i_{k+1} = i_k + \mathrm{h}\left(\left(\frac{77}{1440}\right) - b_8\right)k_1 + \mathrm{h}\left(\frac{32}{105}\right)k_4 + \mathrm{h}\left(\frac{1771561}{6289920}\right)k_5 + \mathrm{h}\left(\frac{243}{2560}\right)k_6 + \mathrm{h}\left(\frac{16807}{74880}\right)k_7 + \mathrm{h}(b_8)k_8 + \mathrm{h}\left(\frac{11}{270}\right)k_9 \quad (29)$$

With

$$k_1 = -2i + 20\,cos\,(100t), \quad (30)$$

$$k_2 = -\left(\frac{299}{150}\right)i - \left(\frac{1}{15}\right)cos(100t) + 20\,cos\left(100t + \frac{1}{6}\right), \quad (31)$$

$$k_3 = -\left(\frac{44701}{22500}\right)\mathrm{i} + \left(\frac{1}{2250}\right)\cos(100\mathrm{t}) - \left(\frac{2}{15}\right)\cos\left(100\mathrm{t} + \frac{1}{6}\right) + 20\cos\left(100\mathrm{t} + \frac{1}{3}\right), \quad (32)$$

$$k_4 = -\left(\frac{59601}{30100}\right)i - \left(\frac{15001}{300000}\right)cos(100t) + \left(\frac{1}{1000}\right)cos\left(100t + \frac{1}{6}\right) - \left(\frac{3}{20}\right)cos\left(100t + \frac{1}{3}\right) + 20\,cos\left(100t + \frac{1}{2}\right), \quad (33)$$

$$k_5 = -\left(\frac{443867}{222742}\right)i - \left(\frac{7905}{177557}\right)cos(100t) + \left(\frac{244}{809643}\right)cos\left(100t + \frac{1}{6}\right) - \left(\frac{4679}{103506}\right)cos\left(100t + \frac{1}{3}\right) + \left(\frac{112}{6655}\right)cos\left(100t + \frac{1}{2}\right) + 20\,cos\left(100t + \frac{2}{11}\right), \quad (34)$$

$$k_6 = -\left(\frac{32259}{16346}\right)i + \left(\frac{11306}{16799}\right)cos(100t) - \left(\frac{3661}{216946}\right)cos\left(100t + \frac{1}{6}\right) + \left(\frac{6313}{2494}\right)cos\left(100t + \frac{1}{3}\right) - \left(\frac{31523}{33239}\right)cos\left(100t + \frac{1}{2}\right) - \left(\frac{21296}{8505}\right)cos\left(100t + \frac{2}{11}\right) + 20\,cos\left(100t + \frac{2}{3}\right), \quad (35)$$

$$k_7 = -\left(\frac{27298}{13885}\right)i - \left(\frac{12962}{8777}\right)cos\left(100t + \frac{1}{3}\right) + \left(\frac{25653}{20402}\right)cos\left(100t + \frac{2}{11}\right) + \left(\frac{34454}{73977}\right)cos\left(100t + \frac{1}{2}\right) - \left(\frac{2106}{12005}\right)cos\left(100t + \frac{2}{3}\right) + \left(\frac{3437}{349096}\right)cos\left(100t + \frac{1}{6}\right) - \left(\frac{11258}{26763}\right)cos(100t) + 20\,cos\left(100t + \frac{6}{7}\right), \quad (36)$$

$$k_8 = -\left(\frac{1}{320512820512b_8}\right)i - \left(\frac{343}{374400b_8}\right)cos\left(100t + \frac{6}{7}\right) + \left(\frac{1248}{2324605b_8}\right)cos\left(100t + \frac{1}{3}\right) + \left(\frac{1093}{776977b_8}\right)cos\left(100t + \frac{2}{11}\right) - \left(\frac{2727}{682439b_8}\right)cos\left(100t + \frac{1}{2}\right) + \left(\frac{1087}{308486b_8}\right)cos\left(100t + \frac{2}{3}\right) - \left(\frac{36}{10058387b_8}\right)cos\left(100t + \frac{1}{6}\right) - \left(\frac{749}{1357900b_8}\right)cos(100t) - 2i + 20\,cos(100t), \quad (37)$$

$$k_9 = -\left(\frac{14603}{7449}\right)i + \left(\frac{68160}{18623}\right)cos\left(100t + \frac{1}{3}\right) - \left(\frac{56659}{16803}\right)cos\left(100t + \frac{2}{11}\right) - \left(\frac{33964}{18229}\right)cos\left(100t + \frac{1}{2}\right) + \left(\frac{170091}{298214}\right)cos\left(100t + \frac{2}{3}\right) + \left(\frac{19683}{21278}\right)cos(100t) - \left(\frac{1}{5000000000}\right)ib_8 + \left(\frac{1}{500000000}\right)cos(100t)\,b_8 - \left(\frac{7123}{291927}\right)cos\left(100t + \frac{1}{6}\right) - \left(\frac{7703}{26391}\right)cos\left(100t + \frac{6}{7}\right) + 20\,cos(100t + 1). \quad (38)$$

With h = Δt = 0:01 and i = 0 when t = 0.

4.2. Determination of Some Values b_8

In this part, we determine some values of b_8 for which methods are better compared to that of Butcher [8]. The method is as follows:

The error is calculated at each step of the Butcher method. That is, b_8 =77/1440. We write Ei, the error with each step.

Calculate the error at each step of the method for b_8 ≠77/1440.

We count all the number of error at each step that are less than or equal to the number of error at each step of the method of Butcher. We write compt, this number.

If compt = 100%, then all local errors in the new method are less than or equal to all the local errors of the Butcher method.

For example, we assume that $b_8 \in$ [-1; 1]. Then increment

b_8 by step of 0:0001 and 0:00001. compt = 100% are obtained (see Table 1). The code of the java programming is given on the annex page (see Appendix):

Table 1. *Determination of some values of* b_8.

Incrémentation de b_8 de 0.0001	Incrémentation de b_8 de 0.00001
-0.0040000006556510925	-0.0010700002312660217
-0.004799999296665192	-0.002209998667240143
-0.005199998617172241	-0.003010004758834839
-0.005599997937679291	-0.007650040090084076
-0.00639999657869339	-0.007890038192272186
-0.0067999958992004395	-0.008370019495487213
-0.007199995219707489	-0.008610010147094727
-0.007999993860721588	-0.009009994566440582
-0.008399993181228638	0.0010700002312660217
-0.008799992501735687	0.002209998667240143
0.0040000006556510925	0.003010004758834839
0.004799999296665192	0.005570024251937866
0.005199998617172241	0.005730025470256805
0.005599997937679291	0.006210029125213623
0.00639999657869339	0.006370030343532562

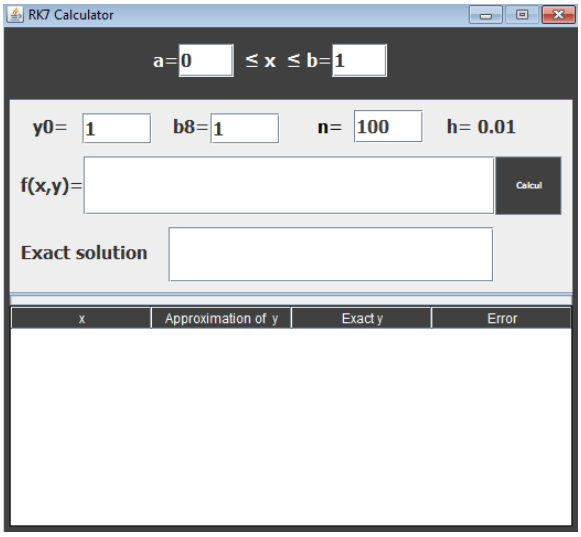

Figure 4. *The Initial layout of the RK7 family calculator.*

5. The RK7 Java Calculator

Let's design the RK7 Program Java.exe to compute digital solutions. Before installing the software, the user will have to install the JDK java package. Just copy it and paste it where you want it and the software is installed. By clicking on RK7 Program Java.exe, we get the following window (see Figure 3).

The values of a; b; $b8$ are given by default. Note that $b8 = b_8$. The value of h is automatically calculated according to the choices of a; b and n. Remember that x is between a and b.

The user will have to enter the range $f(x, y)$ which is defined by: $y' = f(x, y)$. He will also have to enter the exact solution in Exact solution range if he knows. Otherwise he can leave the field empty.

If the user clicks on calculation without filling in the field $f(x, y)$ there is an error and we obtain the following dialog box (see Figure 4).

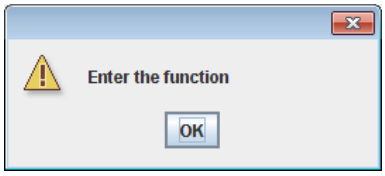

Figure 5. *Error message box if no function is entered.*

If after filling in the field $f(x, y)$ the user enters the field $b8 = 0$, then there is an error in the method and we obtain (see Figure 5).

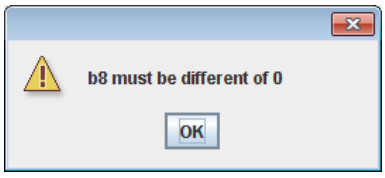

Figure 6. *Error message box if b8 = 0.*

Let us know the differential equation of the publication [2] and take the table of Butcher [1], ie $b8 = 77/1440$. We obtain (see Figure 6).

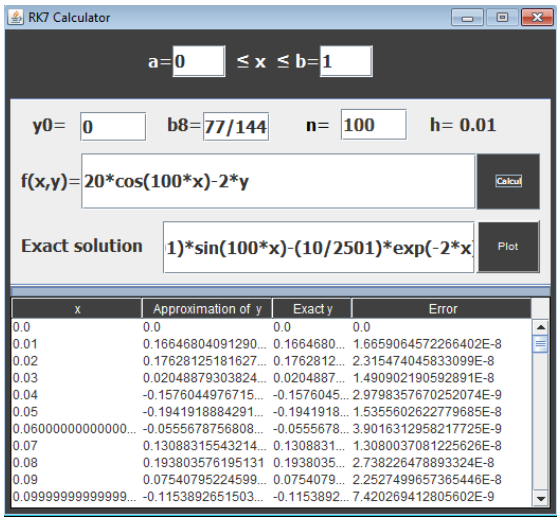

Figure 7. *The full solution of the ODE by the RK7 method with error estimate.*

6. Conclusion

In this article, a new seventh order Runge-Kutta family has been discovered. This method depends on the coefficient b_8. For b_8 = 77/1440, we find the Butcher method [8] on page 196. It has been shown that the stability region does not depend on the coefficient b_8. However, the code in java programming allows us to determine values of b_8 so that all the local errors of these methods are better than those of the Butcher method. This code can be used to find the best method between any two proposed methods. RK7 Java Calculator can be recommended in teaching and also to all those who have no basis in numerical analysis to find a numerical solution of an ordinary differential equation.

Acknowledgements

We would like to express our deepest appreciation and gratitude to Professor Sergey Khashin of Ivanovo State University who provided us the possibility to coordinate and complete this article.

Appendix

```
package Perfect;

import com.singularsys.jep.Jep;
import com.singularsys.jep.JepException;


public class Perfect {
 static double a=0;
 static double b=1;
 static int n=100;
 static double h=(b-a)*Math.pow(n, -1);;
 static double k[];
 static double x[];
 static double y[];
 static double g[];
 static int s=0;
 static double b8=0;
 static double y0=0;
    // function f=f_abs. For example: y'=f(x,y)
 static String f_abs="20*cos(100*x)-2*y";
    // exact solution
 static String f_ex="(10/2501)*cos(100*x)+(500/2501)*sin(100*x)-(10/2501)*exp(-2*x)";
 static int compt=0;
 public static void main(String[] args) {
  x=new double[n+1];
  y=new double[n+1];
  g=new double[n+1];

 // incrementation of b8
  for(float i=0;i<2;i=i+0.001f){

   if(i<1){
       Methode(7,f_abs,f_ex,i-1);
   }
   if(i>1){
   Methode(7,f_abs,f_ex,i-1);
   }
   }

   }


 // determination of percentage
 private static void Methode(int ordre,String funct,String excat,double b8){
  x[0]=0;
  y[0]=0;
  g[0]=0;
  s=butcher(ordre,b8).length-1;
  k=new double[s];
  compt=0;

  for(int i=0;i<n;i++){

   k[0]=f(x[i],y[i],funct);
   x[i+1]=x[i]+h;

   for(int j=1;j<s;j++){

k[j]=f(x[i]+butcher(ordre,b8)[j][0]*h,y[i]+h*somprod(butcher(ordre,b8),k,j),funct);
   }

   y[i+1]=y[i]+h*somprod(butcher(ordre,b8), k, s,ordre);
                // method of Butcher[1]
   g[i+1]=y[i]+h*somprod(butcher(ordre,(77/1440)), k, s,ordre);

   if((y[i+1]-f(x[i+1],f_ex)<=(g[i+1]-f(x[i+1],f_ex)))){
    compt++;
   }

  }
  if(compt==n){
  System.out.println(b8+":"+((compt*100)/n));
  }


 }

  private static double f(double x,double y,String funct){
   float f = 0;
   Jep jep = new Jep();
    try {
      jep.addVariable("x", x);
      jep.addVariable("y", y);
      jep.parse(funct);
       f= Float.parseFloat(jep.evaluate().toString());

    } catch (JepException e) {



    }

  return f;

 }
   private static double f(double x,String funct){
    float f = 0;
    Jep jep = new Jep();
     try {
```

```
            jep.addVariable("x", x);
            jep.parse(funct);
            f= Float.parseFloat(jep.evaluate().toString());

          } catch (JepException e) {



        }

      return f;

    }
      private static double somprod(double a[][],double k[],int
j){
       double som=0;
       for(int i=1;i<=j;i++){
        som=som+a[j][i]*k[i-1];

       }


      return som;

       }
      private static double somprod(double a[][],double k[],int
s,int ordre){
       double som=0;
       for(int i=1;i<=s;i++){
        som=som+a[s][i]*k[i-1];
       }
      return som;

       }
      private static double[][] butcher(int ordre,double b8){
       double[][] tab = null;

       double[][] matrice_ordre7={
        {0,0,0,0,0,0,0,0,0,0}
        ,{(1d/6d),(1d/6d),0,0,0,0,0,0,0,0}
        ,{(1d/3d),0,(1d/3d),0,0,0,0,0,0,0}
        ,{(1d/2d),(1d/8d),0,(3d/8d),0,0,0,0,0,0}
        ,{(2d/11d),(148d/1331d),0,(150d/1331d),(-
56d/1331d),0,0,0,0,0}
        ,{(2d/3d),(-404d/243d),0,(-
170d/27d),(4024d/1701d),(10648d/1701d),0,0,0,0}
        ,{(6d/7d),(2466d/2401d),0,(1242d/343d),(-
19176d/16807d),(-51909d/16807d),(1053d/2401d),0,0,0}
        ,{0,(1d/576d*b8),0,0,(1d/105d*b8),(-
1331d/279552d*b8),(-
9d/1024d*b8),(343d/149760d*b8),0,0}
        ,{1d,((-71d/32d)-(270d/11d)*b8),0,(-
195d/22d),(32d/7d),(29403d/3584d),(-
729d/512d),(1029d/1408d),(270d/11d)*b8,0}
        ,{0,((77d/1440d)-
b8),0,0,(32d/105d),(1771561d/6289920d),(243d/2560d),(168
07d/74880d),b8,(11d/270d)}
       };

       if(ordre==7){
        tab=matrice_ordre7;
       }
      return tab;
       }


   }
```

References

[1] Najmuddin Ahamad, Shiv Sharan "Study of Numerical Solution of Fourth Order Ordinary Differential Equations by Fifth Order Runge-Kutta Method". IJSRSET. 6 (1). 2019.

[2] Robert I. McLachlan et al. "Butcher Series. A story of rooted trees and numerical methods for evolution equations". Asia Pacific Mathematics Newsletter. 2017.

[3] Gashu Gadisa Kiltu, Habtamu Garoma. "Comparison of Higher Order Taylor's Method and Runge-Kutta Methods for Solving First Order Differential Equations". Journal of Computer and Mathematical Sciences. 8 (1). 2017.

[4] Trésor Kanyiki. "Contribution of a Runge-Kutta Order 4 Method in a Dynamic of Mechanical System". ISTE Open Science. 2018.

[5] F. A. Fawzi et al. "An Embedded 6 (5) pair of Explicit Runge-Kutta Method for Periodic IVPs". Far East Journal of Mathematical Sciences. Vol 100. Issue 11. P 1841-1857 (2016).

[6] Séka Hippolyte, Kouassi Assui Richard, "A New Eighth Order Runge-Kutta Family Method", Journal of Mathematics Research. 11 (2), 2019.

[7] Abbas Fadhil Abbas Al-Shimmary, "Solving initial value using Runge-Kutta 6th order method", ARPN Journal of Engineering and Applied Sciences. 12 (12), 2017.

[8] J. C. Butcher "Numericals Methods for Ordinary Differential Equations". Second Edition, 2008.

[9] Mohammed S. Mechee et al. "On the rehabilitee stability of direct explicit Runge-Kutta integrators". Global Journal of Pure and Applied Mathematics 12 (4). 2016, pp. 3959-3975.

[10] Z. W. Sun et al. "Stability of the fourth order Runge-Kutta method for time-dependent partial differential equations". Annals of Mathematical sciences and Application. 2 (2). 2017.

[11] Rachid Ait-Haddou "New Stability Results for Explicit Runge-Kutta Methods". arXiv: 1804.09896. (2018).

[12] Kim Gaik Taya, et al., "The fourth Order Runge-Kutta Spreadsheet Calculator Using VBA Programing for ordinary differential equations", 4th World Congress on Technical and Vocational Education and Training (WoCTVET), 5th–6th November, 2014, Malaysia.

[13] Sergey Khashin, "List of some known Runge-Kutta methods." http://math.ivanovo.ac.ru/dalgebra/Khashin/rk/sh_rk.html

[14] Kasim Hussain, et al., "Runge-Kutta Type Methods for Directly Solving Special Fourth-, Order Ordinary Differential Equations." Problems in Engineering Volume 2015.

[15] Md. Babul Hossain, Md. Jahangir Hossain Md. Musa. Miah and Md. Shah Alam. A Comparative Study on Fourth Order and Butchers Fifth Order Runge-Kutta Methods with Third Order Initial Value Problem (IVP). Applied and Computational Mathematics. December 2015 Volume 2015.

[16] Md. Babul Hossain, et al., "A Comparative Study on Fourth Order and Butchers Fifth Order Runge-Kutta Methods with Third Order Initial Value Problem (IVP)."

[17] ROSETTACODE. ORG. https://rosettacode.org/wiki/Runge-Kutta method.

[18] Stephan Houben, Stability regions of Runge-Kutta methods. Eindhoven University of Technology. February 19, 2002.

[19] Jackiewicz, Zdzislaw, General Linear Methods for Ordinary Differential Equations. 482 p., 2009. isbn 978-0-470-40855-1. doi 10.1002/9780470522165. publisher John Wiley & Sons, Inc.

[20] M. Calvo and J. I. Montijano and L. Randez, A new embedded pair of Runge-Kutta formulas of orders 5 and 6. Comput. Math. Appl. Vol. 20, No. 1, 1990. Issn 0898-1221. 15-24 p. doi http://dx.doi.org/10.1016/0898-1221(90)90064-Q, Pergamon Press, Inc., Tarrytown, NY, USA.

[21] C. R. Cassity, The complete solution of the fifth order Runge-Kutta equations, SIAM J. Numer. Anal., vol. 6, 1969, 432-436 p.

[22] T. Feagin, A tenth-order Runge-Kutta method with error estimate, Proceedings of the IAENG Conf. on Scientific Computing, 2007, Hong Kong.

[23] T. Feagin, High-Order Explicit Runge-Kutta Methods Using m-Symmetry. Neural, Parallel & Scientific Computations. Vol 20. Issue 3/4, p 437 (2012).

[24] E. Hairer, S. P. Norsett, G. Wanner, isbn 3-540-56670-8. 528 p., 2Ed. Springer-Verlag, Solving ordinary differential equations I. Nonstiff Problems, 2000.

[25] Hairer, Ernst; Norsett, Syvert Paul; Wanner, Gerhard, Solving ordinary differential equations I. Nonstiff Problems, Springer-Verlag, isbn 978-3-540-56670-0, 2008.

[26] Sindhuja Reddy Velgala. Stability analysis of the 4^{th} order Runge-Kutta method in application to colloidal particle interactions. Thesis University of Illinois at Urbana-Champaign, 2014.

[27] M. Z. Liu, M. H. Song, Z. W. Yang, Stability of RungeKutta methods in the numerical solution of equation $u'(t) = au(t) + a_0u([t])$. Volume 166, Issue 2, 15 April 2004, Pages 361-370.