International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)
Web Site: www.ijettcs.org Email: editor@ijettcs.org
Volume 6, Issue 3, May- June 2017 ISSN 2278-6856

ICT Tool: - 'C' Language Program for Runge-Kutta Method

Sanjay C. Gawande[1], Pradip P. Kolhe[2] and Dr. Prakash R. Kolhe[3]

[1]College of Agriculture, Dr. P.D.K.V., Akola,
P.O. Krishi Nagar, Akola – 444104 (M.S.), India

[2]ARIS Cell, Dr. P.D.K..V., Akola
P.O. Krishi Nagar, Akola -444104 (M.S.), India

[3]College of Agricultural Engineering & Technology, Dr. BSKKV, Dapoli
Dapoli (M.S.), India

Abstract: ***In mathematics, Runge-Kutta method here after called as RK method is the generalization of the concept used in Modified Euler's method. A number of ordinary differential equations come in engineering and all of them may not be solved by analytical method. In order to solve or get numerical solution of such ordinary differential equations, Runge Kutta method is one of the widely used methods.***
In the era of Information Communication Technology (ICT) .The ICT programming technique, it is easier task. One of the very popular programs in C programming is Runge-Kutta method. This paper discuss Runge-Kutta method in C language, source code and methods with outputs. The source codes of program for Runge-Kutta method in C programming are to be compiled. Running them on Turbo C or available version and other platforms might require a few modifications to the code. You probably know how to multiply two matrices.
Keywords: Runge-Kutta Method, ICT, C Lang., Turbo C, Ordinary Differential Equations.

1. INTRODUCTION

One of the very popular programs in C programming is Runge-Kutta Method. Runge-Kutta method is a popular iteration method of approximating solution of ordinary differential equations. Developed around 1900 by German mathematicians C.Runge and M. W. Kutta, this method is applicable to both families of explicit and implicit functions.

Also known as RK method, the Runge-Kutta method is based on solution procedure of initial value problem in which the initial conditions are known. Based on the order of differential equation, there are different Runge-Kutta methods which are commonly referred to as: RK2, RK3, and RK4 methods.

2. RUNGE-KUTTA METHOD IS DESCRIBED AS

The most widely known member of the Runge–Kutta family is generally referred to as "RK4", "classical Runge–Kutta method" or simply as "the Runge–Kutta method".

Let an initial value problem be specified as follows:

$$\dot{y} = f(t, y), \quad y(t_0) = y_0.$$

Here *y* is an unknown function (scalar or vector) of time *t*, which we would like to approximate; we are told that $\dot{y}$, the rate at which y changes, is a function of t and of y itself. At the initial time t_0 the corresponding *y* value is y_0. The function *f* and the data t_0, y_0 are given.

Now pick a step-size $h > 0$ and define

$$y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4),$$
$$t_{n+1} = t_n + h$$

for $n = 0, 1, 2, 3, \ldots$, using

$$k_1 = f(t_n, y_n),$$
$$k_2 = f(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_1),$$
$$k_3 = f(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_2),$$
$$k_4 = f(t_n + h, y_n + hk_3).$$

3. RUNGE KUTTA METHOD IS

Let's consider an initial value problem given as:

$y' = f(t, y)$, $\quad y(t_0) = y_0$

where,

- y is an unknown function of time t, which may be scalar or vector quantity
- y' is the rate of change of y with respect to t
- t_0 is the initial value of t
- y_0 is the value of y at time t_0

Let 'h' be the step size such that h > 0. Now, the generation term of the series can be defined as:

$y_{n+1} = y_n + h/6 * (k_1 + 2k_2 + 2k_3 + k_4)$
$t_{n+1} = t_n + h$

for n = 1, 2 , 3, 4, …. such that:

- $k_1 = f(t_n, y_n)$
- $k_2 = f(tn + h/2 , yn + h/2\ k_1)$
- $k_3 = f(tn + h/2, y_n + h/2k_2)$
- $k_4 = f(t_n + h, y_n + hk_3)$
- yn+1 is the Runge-Kutta method approximation of y(tn+1)
- k_1 is the increment which depends on the gradient of starting interval as in Euler's method
- k_2 is the increment which relies on the slope at the midpoint of the interval, $k_2 = y+ h/2 * k_1$
- k_3 is also an increment based on the gradient at midpoint, k3=y+h/2*k2
- k_4 is again an increment whose value depends on end of the end of interval $k_4 = y + hk_3$

4. ALGORITHM FOR RUNGE-KUTTA METHOD

Runge-Kutta Method for 4th order
Step 1: Read x1,y1 initial values.
Step 2: Read a, value at which function value is to be found.
Step 3: Read n, the number of steps.
Step 4: count=0
Step 5: h=(a-x1)/n

Step 6: write x1,y1
Step 7: s1=f(x1,y1)
Step 8: s2=f(x1+h/2,y1+s1*h/2)
Step 9: s3=f(x1+h/2,y1+s2*h/2)
Step 10: s4=f(x1+h,y1+s3*h)
Step 11: y2=y1+(s1+2*s2+2*s3+s4)*h/6
Step 12: x2=x1+h
Step 13: write x2,y2
Step 14: count=count+1
Step 15: If count<n . then
x1=x2
y1=y2
go to step step 7
endif
Step 16: write x2,y2
Step 17: stop

Runge-Kutta Method for 2nd order
Step 1: Read x1,y1, initial values
Step 2: Read a,value at which function value is to be found
Step 3: Read n,the number of subintervals
Step 4: count=0
Step 5: h=(a-x1)/n
Step 6: write x1,y1
Step 7: k1=h*f(x1,y1)
Step 8: k2=h*f(x1+h,y1+k1)
Step 9: y2=y1+(k1+k2)/2
Step 10: x2=x1+h
Step 11: write x2,y2
Step 12: count=count+1
Step 13: If count<n Then
x1=x2
y1=y2
Go to Step 7
Endif
Step 14: write x2,y2
Step 15: Stop

5. SOURCE CODE FOR RUNGE KUTTA METHOD IN C:

```
#include<stdio.h>
#include<math.h>
float f(float x,float y);
int main()
{
   float x0,y0,m1,m2,m3,m4,m,y,x,h,xn;
   printf("Enter x0,y0,xn,h:");
   scanf("%f %f %f %f",&x0,&y0,&xn,&h);
   x=x0;
   y=y0;
   printf("\n\nX\t\tY\n");
   while(x<xn)
   {
      m1=f(x0,y0);
      m2=f((x0+h/2.0),(y0+m1*h/2.0));
      m3=f((x0+h/2.0),(y0+m2*h/2.0));
      m4=f((x0+h),(y0+m3*h));
      m=((m1+2*m2+2*m3+m4)/6);
      y=y+m*h;
      x=x+h;
      printf("%f\t%f\n",x,y);
   }
}
float f(float x,float y)
{
   float m;
   m=(x-y)/(x+y);
   return m;
}
```

Input/Output:

```
Enter x0,y0,xn,h:0 2 2 0.5


X               Y
0.500000        1.621356
1.000000        1.242713
1.500000        0.864069
2.000000        0.485426

Process returned 16384 (0x4000)   execution time : 5.825 s
Press any key to continue.
```

References

[1] Ascher, Uri M.; Petzold, Linda R. (1998), Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations, Philadelphia: Society for Industrial and Applied Mathematics, ISBN 978-0-89871-412-8 .
[2] Atkinson, Kendall A. (1989), An Introduction to Numerical Analysis (2nd ed.), New York: John Wiley & Sons, ISBN 978-0-471-50023-0 .
[3] Butcher, John C. (May 1963), Coefficients for the study of Runge-Kutta integration processes, 3 (2), pp. 185–201, doi:10.1017/S1446788700027932 .
[4] Butcher, John C. (1975), "A stability property of implicit Runge-Kutta methods", BIT, 15: 358–361, doi:10.1007/bf01931672 .
[5] Butcher, John C. (2008), Numerical Methods for Ordinary Differential Equations, New York: John Wiley & Sons, ISBN 978-0-470-72335-7 .
[6] Cellier, F.; Kofman, E. (2006), Continuous System Simulation, Springer Verlag, ISBN 0-387-26102-8 .
[7] Dahlquist, Germund (1963), "A special stability problem for linear multistep methods", BIT, 3: 27–43, doi:10.1007/BF01963532, ISSN 0006-3835 .
[8] Forsythe, George E.; Malcolm, Michael A.; Moler, Cleve B. (1977), Computer Methods for Mathematical Computations, Prentice-Hall (see Chapter 6).
[9] Hairer, Ernst; Nørsett, Syvert Paul; Wanner, Gerhard (1993), Solving ordinary differential equations I: Nonstiff problems, Berlin, New York: Springer-Verlag, ISBN 978-3-540-56670-0 .
[10] Hairer, Ernst; Wanner, Gerhard (1996), Solving ordinary differential equations II: Stiff and differential-algebraic problems (2nd ed.), Berlin, New York: Springer-Verlag, ISBN 978-3-540-60452-5 .
[11] Iserles, Arieh (1996), A First Course in the Numerical Analysis of Differential Equations, Cambridge University Press, ISBN 978-0-521-55655-2 .
[12] Lambert, J.D (1991), Numerical Methods for Ordinary Differential Systems. The Initial Value Problem, John Wiley & Sons, ISBN 0-471-92990-5
[13] Kaw, Autar; Kalu, Egwu (2008), Numerical Methods with Applications (1st ed.), autarkaw.com .
[14] Kutta, Martin Wilhelm (1901), "Beitrag zur näherungsweisen Integration totaler Differentialgleichungen", Zeitschrift für Mathematik und Physik, 46: 435–453 .
[15] Press, William H.; Flannery, Brian P.; Teukolsky, Saul A.; Vetterling, William T. (2007), "Section 17.1 Runge-Kutta Method", Numerical Recipes: The Art of Scientific Computing (3rd ed.), Cambridge University Press, ISBN 978-0-521-88068-8 . Also, Section 17.2. Adaptive Stepsize Control for Runge-Kutta.
[16] Stoer, Josef; Bulirsch, Roland (2002), Introduction to Numerical Analysis (3rd ed.), Berlin, New York: Springer-Verlag, ISBN 978-0-387-95452-3 .
[17] Süli, Endre; Mayers, David (2003), An Introduction to Numerical Analysis, Cambridge University Press, ISBN 0-521-00794-1 .
[18] Tan, Delin; Chen, Zheng (2012), "On A General Formula of Fourth Order Runge-Kutta Method" (PDF), Journal of Mathematical Science & Mathematics Education, 7.2: 1–10 .