A Case Study on Runge Kutta 4 th Order Differential Equations and Its Application

Article *in* "Imperial Journal of Interdisciplinary Research (IJIR) · January 2017

1 author:

Gowri Palanisamy
Sri Krishna Arts and Science College, Coimatore

A Case Study on Runge Kutta 4th Order Differential Equations and Its Application

Gowri.P[1] , Priyadharsini.S[2], Maheswari.T[3]
[1]Asst.Prof Sri Krishna Arts and Science College
[2,3]Sri Krishna Arts and Science College

Abstract: *In this paper it is discussed about Runge Kutta 4th order with differential equations and its applications. The diferential equation problems has sloved by Runge Kutta 4th order method and application problems are discussed with Runge Kutta 4th order and codeded in C programming*.

1.Introduction

First Order Runge-Kutta Method

Consider the following case: we wish to use a computer to approximate the solution of the differential equation

dy(t)dt+2y(t)=0dy(t)dt+2y(t)=0

or dy(t)dt=−2y(t)

Second Order Runge-Kutta Method

The first order Runge-Kutta method used the derivative at time t_0 (t_0=0 in the graph below) to estimate the value of the function at one time step in the future. If you are not familiar with it, you should read the section entitled: A First Order Linear Differential Equation with No Input. We repeat the central concept of generating a step forward in time in the following text.

dy(t)dt+2y(t)=0ordy(t)dt=−2y(t)

Third Order Runge-Kutta Method

This method is a third order Runge-Kutta method for approximating the solution of the initial value problem $y'(x) = f(x,y)$; $y(x_0) = y_0$ which evaluates the integrand, *f(x,y)*, three times per step. For step *i+1*,

$$y_{i+1} = y_i + 1/6 (k_1 + 4 k_2 + k_3),$$

where

$$k_1 = h f(x_i, y_i),$$
$$k_2 = h f(x_i + h / 2, y_i + k_1 / 2),$$
$$k_3 = h f(x_i + h, y_i - k_1 + 2 k_2),$$

and $x_i = x_0 + i h$.

Fourth Order Runge-Kutta Method

Runge-Kutta 4th order method is a numerical technique used to solve ordinary differential equation of the form

dy/dx = f(x,y),y(0)=y0

So only first order ordinary differential equations can be solved by using the Runge-Kutta 4th order method. Euler and Runge-Kutta methods are used to solve higher order ordinary differential equations or coupled (simultaneous) differential equations

2. Derivation

The runge –kutta 4th order method is based on the following

$$y_{i+1} = y_i + a_1k_1 + a_2k_2 + a_3k_3 + a_4k_4)h$$
(1)

Where knowing the value of $y=y_i$ at x_i, we can find the value of $y=y_{i+1}$ at x_{i+1} , and

$$h = x_{i+1} - x_i$$

Equation is equated to the first five terms of Taylor series

$$y_{i+1} = y_i + \frac{dy}{dx}\Big|_{x_i,y_i}(x_{i+1}-x_i) + \frac{1}{2!}\frac{d^2y}{dx^2}\Big|_{x_i,y_i}(x_{i+1}-x_i)^2 + \frac{1}{3!}\frac{d^3y}{dx^3}\Big|_{x_i,y_i}(x_{i+1}-x_i)^3 + \frac{1}{4!}\frac{d^4y}{dx^4}\Big|_{x_i,y_i}(x_{i+1}-x_i)^4$$

(2)

Knowing that dy/dx=f(x,y) and $x_{i+1} - x_i = h$

$$y_{i+1} = y_i + f(x_i, y_i)h + 1/2! f'(x_i, y_i)h^2 + 1/3! f'(x_i, y_i)h^3 + 1/4! f'''(x_i, y_i)h^4 \qquad (3)$$

Based on equating equation (2)and equation(3),one of the popular solutions used is

Y_{i+1}=yi+1/6(k_1+2k_2+2k_3+k_4)h

k_1=f (x_i,y_i)

k_2=f(x_i+h/2,y_i+k_1/2 h)

$k_3=f(x_i+h/2,y_i+k_2/2)$

$k_4=f(x_i+h,y_i+k_3\,h$

3. Problems

1. Using fourth order runge-kutta method. Evaluate the value of y when x=1.1 given that dy/dx+y/x=1/x^2 y(1) =1

Solution :
The formula for the fourth order Runge-kutta method of the differential equation dy/dx=f(x,y) is given by
$k_1=h *f (x_{0,}y_0)$
$k_2=h*f(x_0+h/2,y_0+k_1/2)$
$k_3=h*f(x_0+h/2,y_0+k_2/2)$
$k_4=h*f(x_0+h,y_0+k_3)$
$\Delta_y=1/6(k_1+2k_2+2k_3+k_4)$
Where h is the interval of differencing and $(x_{0,}y_0)$ is the initial value
Hence f (x,y)=1/x^2-y/x; x_0 =1 y_0=1 h=0.1
$K_1=(0.1)*(1/1^2-1/1)$
=0
$K_2=(0.1)\ [(1/(x_0+h/2)^2-(y_0+k_1/2/x_0+h/2))]$
$=(0.1)[(1/(1+0.1/2)^2)-(1/1+0.1/2)]$
=-0.00454
K_3=(0.1)[(0.9070)-(1+(-0.00454/2)/1.05)]
=0.1(0.9070-0.9502)
=-0.00432
$K_4=(0.1)[(1/(1.1)^2)-(1-0.00432/1.1)$
=(0.1)(0.8264-0.9052)
=-0.00788
Δ_y=1/6(0-0.00908-0.00864-0.00788)
=-0.0042667
Y_1=y(1.1)
$=y_0+ \Delta_y$
=1+(-0.0042667)
=0.9957

2. Compute y(0.1) and y(0.2) by runge kutta method of fourth order differential equation dy/dx=xy+y^2 ,y(0) =1
Solution:
The formula for the fourth order runge kutta method are
$k_1=h *f (x_{0,}y_0)$
$k_2=h*f(x_0+h/2,y_0+k_1/2)$
$k_3=h*f(x_0+h/2,y_0+k_2/2)$
$k_4=h*f(x_0+h,y_0+k_3)$
$\Delta_y=1/6(k_1+2k_2+2k_3+k_4)$
Where h is the interval of differencing and $(x_{0,}y_0)$ is the initial value
Hence f (x,y)=xy/y^2; x_0 =1 y_0=1 h=0.1
K_1=(0.1)(0+1)=0.1
$K_2=(0.1)[(0.05(1.05)+(1.05)^2)]$
=0.1155
$K_3=(0.1)[0.05(1.05775)+(1.05775)^2$
=0.1172
$K_4=(0.1)[(0.1)(1.1172)+(1.1172)^2]$
=0.1360
Δ_y=1/6[0.1+0.2310+0.2344+0.1366]
=0.1169
Y_1=y0+ Δ_y
=1+0.1169
=1.1169
y(0.1)=1.1169
For the second approximation we have x1=0.1
$K_1=0.1[0.1*(1.1169)+(1.1169)^2\ ]$
=0.1359
$K_2=0.1[0.15(1.1849)+(1.1849)^2]$
=0.1582
$K_3=0.1[0.15(1.196)+(1.196)^2]$
=0.1610
$K_4=0.1[0.2(1.2779)+(1.2779)^2]$
=0.1889
Δ_y=1/6[0.1359+0.3164+0.3220+0.1889]
=0.1605
Y_2=y1+ Δ_y
=1.1169+0.605
=1.2774
Y(0.2)=1.2774

3. Use the classical RK method to estimate y(0.4) when $y'(x) = x^2 + y^2$ with y(0)=0 assume h=0.2
SOLUTION :
$$f(x,y) = x^2 + y^2$$
$$m_1 = f(x_0, y_0) = 0$$
$$m_2 = f(x_0 + \frac{h}{2}, y_0 + \frac{m_1 h}{2})$$
$$= f(0.1,0) = 0.01$$
$$m_3 = f(x_0 + \frac{h}{2}, y_0 + \frac{m_2 h}{2})$$
$$== f(\frac{0.2}{2}, \frac{0.01\times 0.2}{2})$$
=0.01
$$m_4 = f(x_0 + h, y_0 + m_3 h)$$
$$= f(0.2, 0.01\times 0.2)$$
=0.04
$$y(0.2) = 0 + \frac{0 + 2\times 0.01 + 2\times 0.01 + 0.04}{6} 0.2$$

=0.002667
ITERATION 2
X_1 =0.2
Y_1 =0.002667

4. Application Problem

1. In this program for Runge Kutta method in C, a function f(x,y) is defined to calculate slope whenever it is called. f(x,y) = (x-y)/(x+y)
SOLUTION:
#include<studio.h>

```
#include<math.h>
float f(float x,float y);
void 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);
   }
getch();
}
float f(float x,float y)
{
   float m;
   m=(x-y)/(x+y);
   return m;
}
```

OUTPUT:

Enter x0,y0,xn,h :
X=
Y=

2. Using fourth order runge-kutta method. Evaluate the value of y when x=1.1 given that dy/dx+y/x=1/x^2 y(1) =1

Solution:

Function[]=runge (f,x_0,y_0,x_n,n)
x=x_0, y_1=y_0;
h=x_n-x_0/n;
disp('x rk4')
for i=1:n
k_1=h*f(x,y_1);
k_2=h*f(x+0.5*h,y_1+0.5*k_1);
k_3=h*f(x+0.5*h,y_1+0.5*k_2);
k_4=h*f(x+h,y_1+k_3);
Y_1=y_1+1/6(k_1+2*k_2+2*k_3+k_4);
X=x+h;
disp ([x y_1])
end
disp ('RK 4th order method:')
disp ([y_1])

OUTPUT:

f=@(x,y)1+y*y;
Runge(f,1,1, , 5)

3. A polluted lake has an initial concentration of a bacteria of 10^{7}parts/m^3 .The concentration of the bacteria will reduce as fresh water enters the lake .Find the concentration of the pollutant after 7 weeks.

$$\frac{dC}{dt}+0.06c=0$$

The differential equation that governs the concentration c of the pollution as a function of time (in week) is given by

$$\frac{dc}{dt}+0.06c=0, c(0)=10^6$$

$$\frac{dc}{dt}=-0.06c$$

$$f(t,c)=-0.06c$$

$$c_{i+1}=c_1+1/6(k_1+2k_2+2k_3+k_4)h$$

For i=0,t_0 =0,c_0 =10^7

$$k_1=f(t_0,c_0)$$

$$=f(0,10^7)$$

$$=-0.06(10^7)$$

$$=-600000$$

$$k_2=f(t_0+1/2\times h,c_0+1/2k_1h)$$

$$=f(0+1/2\times h,c_0+1/2k_1h)$$

$$=f(0+1/2\times 3.5,10^7+1/2(-600000)3.5)$$

$$=f(1.75,8950000)$$

$$=-0.06(8950000)$$

$$=-537000$$

$$k_3=f(t_0+1/2\times h,c_0+1/2k_2h)$$

$$=f(0+1/2\times 3.5,10^7+1/2(-537000)3.5)$$

$$=f(1.75,9060300)$$

$$=-0.06(9060300)$$

$= -543620$

$k_4 = f(t_0 + h, c_0 + k_3 h)$

$= f(0 + 3.5, 10^7 + (-543620)3.5)$

$= f(3.5, 8097300)$

$= -0.06(8097300)$

$= -485840$

$c_1 = c_0 + 1/6(k_1 + 2k_2 + 2k_3 + k_4)h$

$= 10^7 + 1/6(-600000 + 2(-537000) + 2(-543620) + (-485840))3.5$

$= 10^7 + 1/6(-3247100)3.5$

$= 8.1059 \times 10^6\ parts/m^3$

C_1 is the approximate concentration of bacteria at

$t=t_1=t_0+h=0+3.5=3.5$ parts/m^3

$C(3.5)=C_1 = 8.1059 \times 10^6\ parts/m^3$

For i=1, $t_1=3.5$, $C_1 = 8.1059 \times 10^6$

$k_1 = f(t_1, c_1)$

$= f(3.5, 8.1059 \times 10^6)$

$= -0.06(\ 8.1059 \times 10^6)$

$= -486350$

$k_2 = f(t_1 + 1/2 \times h, c_1 + 1/2 k_1 h)$

$= f(3.5 + 1/2 \times 3.5, 8105900 + 1/2(-486350)3.5)$

$= f(5.25, 7254800)$

$= -0.06(7254800)$

-435290

$k_3 = f(t_1 + 1/2 \times h, c_1 + 1/2 k_2 h)$

$= f(3.5 + 1/2 \times 3.5, 8105900 + 1/2(-440648)3.5)$

$= f(5.25, 7344100)$

$= -0.06(7444100)$

$= -440648$

$k_4 = f(t_1 + h, c_1 + k_3 h)$

$= f(3.5 + 3.5, 8105900 + (-440648)3.5)$

$= f(7, 6563600)$

$= 0.06(6563600)$

$= -393820$

$c_2 = c_1 + 1/6(k_1 + 2k_2 + 2k_3 + k_4)h$

$= 8105900 + 1/6(-486350 + 2 \times (-435290) + 2 \times (-440648) + (-393820)) \times 3.5$

$= 8105900 + 1/6(-2632000) \times 3.5$

$= 6.5705 \times 10^6\ parts/m^3$

C_2 is the approximate concentration of bacteria at

$t_2 = t_1 + h = 3.5 + 3.5 = 7 weeks$ C

C(7)= $C_2 = 6.5705 \times 10^6\ parts/m^3$

PROGRAM:

```
#include<conio.h>
#include<iostream.h>
Void main()
{
float c[10], f ,t[10],h , n;
cout<<"Enter the initial values of concentration of
bacteria:";
int i;
cin>>c[0];
cout<<"Enter the initial value of time:";cin>>t[0];
cout<<"Enter the value of time in weeks at which
we want to see the concentration:";cin>>f;
cout<<"Enter the difference: ";cin>>h;
n=(f-c[0])/h;
for(i=1;i<=n;i++)
```

```
{
a=0
k1=-(.06*c[a]);
k2=-h*(.06*(c[a]+k1/2));
k3=-h*(.06*(c[a]+k2/2));
k4=-h*(.06*(c[a]+k3));
k=(k1+2*k2+2*k3+k4)/6;
y[i]=y[a]+1;
a++;
}
Cout<<"/n table ";
For (i-0;i<n;i++)
{
Cout<<"\t x=%f\ty=%f",val[i][0],val[i][1];
Cout<<"\n";
Getch();
}
```

Output:

Enter the initial values of concentration of bacteria:
Enter the initial value of time:
Enter the value of time in weeks at which we want to see the concentration:
Enter the difference:

4. A ball at 1200 K is allowed to cool down in air at an ambient temperature of 300 K. Assuming heat is lost only due to radiation, the differential equation for the temperature of the ball is given by

$$\frac{d\theta}{dt} = -2.2067\times10^{-12}(\theta^4 - 81\times10^8), \theta(0) = 1200k$$

f(t, θ)= $-2.2067\times10^{-12}(\theta^4 - 81\times10^8)$

$$\theta_{i+1} = \theta_i + 1/6(k_1 + 2k_2 + 2k_3 + k_4)h$$

For i=0, t_0=0, θ_0 =1200k

$$k_1 = f(t_0, \theta_0)$$

=f(0,1200)

$$= -2.2067\times10^{-12}(1200^4 - 81\times10^8)$$

=-4.5579

$$k_2 = f(t_0 + 1/2h, \theta_0 + 1/2k_1h)$$

= $f(0 + 1/2(240), 1200 + 1/2(-4.5579)\times 240)$

=f(120,653.05)

$$= -2.2067\times10^{-12}(653.05^4 - 81\times10^8)$$

=-0.38347

$$k_3 = f(t_0 + 1/2h, \theta_0 + 1/2k_2h)$$

= $f(0 + 1/2(240), 1200 + 1/2(-0.38347)\times 240)$

= $f(120, 1154.0)$

= $-2.2067\times10^{-12}(1154.0^4 - 81\times10^8)$

= -3.8954

$$k_4 = f(t_0 + h, \theta_0 + k_3h)$$

= $f(0 + 240, 1200 + (-3.894)\times 240)$

$$= f(240, 265.10)$$

$$= -2.2067\times10^{-12}(265.10^4 - 81\times10^8)$$

$$= -0.0069750$$

$$\theta_1 = \theta_0 + 1/6(k_1 + 2k_2 + 2k_3 + k_4)h$$

=1200+1/6(-4.5579+2(-0.38954)+0.069750))240

=1200+(-2.1848)$\times 240$

=675.65K

θ_1 is the approximate temperature at

t = t_1

=t_0+h

=0+240

=240

θ_1=θ(240)

~675.65K

For i=1,t_1=240, θ_1=675.65K

$$k_1 = f(t_1, \theta_1)$$

= $f(240, 675.65)$

$= -2.2067 \times 10^{-12}(675.65^4 - 81 \times 10^8)$

=-0.44199

$k_2 = f(t_1 + 1/2h, \theta_1 + 1/2k_1h)$

=
$f(240 + 1/2(240), 675.65 + 1/2(-0.44199)240)$

$= f(360, 622.61)$

$= \quad -2.2067 \times 10^{-12}(622.61^4 - 81 \times 10^8)$

=-0.31372

$k_3 = f(t_1 + 1/2h, \theta_1 + 1/2k_2h)$

=
$f(240 + 1/2(240), 675.65 + 1/2(-0.31372) \times 240)$

$= f(360, 638.00)$

$= -2.2067 \times 10^{-12}(638.00^4 - 81 \times 10^8)$

$= -0.34775$

$k_4 = f(t_1 + h, \theta_1 + k_3h)$
$= f(240 + 240, 675.65 + (-0.34775) \times 240)$

$= f(480, 592.19)$

=
$-2.2067 \times 10^{-12}(592.19^4 - 81 \times 10^8)$

=-0.25351

$\theta_2 = \theta_1 + 1/6(k_1 + 2k_2 + 2k_3 + k_4)h$

=
$675.65 + 1/6(-0.31372) + 2(-0.34775) + (-0.25351)) \times 240$

=675.65+1/6(-2.0184)$\times 240$

=594.91K

θ_2 is the approximate temperature at

$t = t_2$

=t_1+h

=240+240

=480

$\theta_2 = \theta$ (480)

=594.91K

5. References

[1]Atkinson, Kendall A. (1989), An Introduction to Numerical Analysis (2nd ed.), New York: John Wiley & Sons, ISBN 978-0-471-50023-0.

[2]Butcher, John C. (May 1963), study of Runge-Kutta integration processes 3 (2), pp. 185–201, doi:10.1017/S1446788700027932.

[3]Butcher, John C. (1975), "A stability property of implicit Runge-Kutta methods", BIT 15: 358–361, doi:10.1007/bf01931672.

[4]Butcher, John C. (2008), Numerical Methods for Ordinary Differential Equations, New York: John Wiley & Sons, ISBN 978-0-470-72335-7.

[5] John.H.Mathews , KurtisD.Fink , Numerical Method Using MATLAB, ISBN 978-81-203-2765-8.

[6] Laurene V Fausett , Applied Numerical Analysis Using MATLAB ,2009 ,ISBN 978-81-317-2853-6

[7]Pallab Ghosh , Numerical Methods with computer programs in c++ , ISBN 81-203-2987-2

[8] Vedamurthy. V N Dr N Ch S Iyengar, Numerical Methods ,Vellore , ISBN 978-81259-0630-8.