Computer Approach to Runge-Kutta Method of First Order Differential Equation Using C++ Programming.

[1]Atajeromavwo Edafe John [2]Onavwie Ufuoma A.

[1]Department of Computer Science, Delta State Polytechnic, Ogwashi-Uku, Delta State
[2]Department of Physics, College of Education, Warri, Delta State, Nigeria.

Abstract

The paper focused on the Runge-Kutta Method with computer programming approach under the platform of Windows Vista Operating System and C++ programming language. The purpose is to compare exact solution with Runge-Kutta solution and then examine the error difference. The computer programming approach gave faster and accurate solution when compared to other methods of solving first order differential equations. When the exact solution was compared with the Runge-Kutta method solution, the generated errors were found to be negligible.

1.0 Introduction

The purpose of this paper was to use computer application in solving first order differential equation using the Runge-Kutta method algorithm which is transformed to C++ programming language to arrive at a solution by comparing the exact solution with Runge Kutta solution and examine the error difference. There are many numerical methods for solving first order initial value problems of ordinary differential equations, especially non linear, implicit and stiff problems [1]

Computer programme is the set of instruction that are executed by the computer to produce a desirable result. According to Atajeromavwo [2] when the computer is directed to do arithmetic, the program indicates exactly what is required and the data consists of numbers and the program cards tells the computer exactly what to do. The data consists of the number and writing which the computer will need to perform its task. Computer programming involves writing software that allows a machine to perform various tasks [3]. There are two parts to computer program. The sequence of steps which is necessary to complete the given task is refers to algorithm and the translation of these step into language which is fed into computer.

2.0 Methodology

A given First Order Differential Equation was subjected to integration, via the Runge-Kutta method using numerical analysis. The algorithm was then transformed into C++ programming language. A Toshiba Computer laptop made up of windows Vista Operating System platform of 2.0 gigabyte memory and speed 1.73 GHz of Intel processor was used for the programming. Finally, evaluation of the errors involved in the exact solution and Runge Kutta method solution was compared.

3.0 An algorithm for Integrating the Runge Kutta Method.

An algorithm is a sequence of precise instructions which leads to a solution [4]. Some approximately equivalent words are recipe, method, directions, procedure, and routine. The instructions may be expressed in a programming language or in a human language. Our algorithms will be expressed in English and in the programming language C++. A computer program is simply an algorithm expressed in a language that a computer can understand. But since this research was expressed in programming language we would use the more specific term program.

The properties that qualify a set of instructions as algorithm now are determined by the nature of the instruction rather than by the things manipulated by the instructions. To qualify an algorithm as a set of instructions must completely and unambiguously specify the steps to be taken and the order in which they are taken. The person or machine carrying out the algorithm does exactly what the algorithm says, neither more or less. According to Christopher [5], an algorithm is a precisely described routine procedure that can be applied and systematically followed through to a conclusion. Algorithm can also be viewed as a procedure for carrying out a desired computation in solving a problem, written sequentially in a series of steps(6). In the same vein, an algorithm is a logical and coincise series of steps required in solving a given problem(7).

Any first order differential equation can be put into the form

$$F^1=f(x,y) \qquad (3.1)$$

Or

$$M(x,y)dx+N(x,y)dy=0 \qquad (3.2)$$

4.0 Runge-Kutta Method

Suppose that we are to solve the following equation by applying **Runge-Kutta Method**

$$\frac{dx}{dt}= f(t, x, y),\ x(t_0) = x_0 \qquad (4.1)$$

and

$\frac{dy}{dt} = f(t, x, y)\ y(t_0) = y_0$ (4.2)

Note tha the Runge-Kutta Method formula involves a weighted average of value of f(x,y) taken at different points for the interval $Xn<X<X_{n+1}$.then the rule becomes

$x_1 = x_0 + \frac{1}{6}(m_1 + 2m_2 + 2m_3 + 2\ m_4)$ (4.3)

$y_1 = y_0 + \frac{1}{6}(n_1 + 2n_2 + 2n_3 + 2n_4)$ (4.4)

where $m_1, n_1, m_2, n_2, m_3, n_3, m_4, n_4$ are the slope, they are expressed below:

$m_1 = hf(t_0, x_0, y_0)$ $n_1 = hg(t_0, x_0, y_0)$

$m_2 = hf(t_0 + {}^{h}/_{2}, x_0 + \frac{m_1}{2}, y_0 + \frac{n_2}{2},)$, $n_2 = hg(t_0 + {}^{h}/_{2}, x_0 + \frac{m_1}{2}, y_0 + \frac{n_1}{2})$

$m_3 = hf(t_0 + {}^{h}/_{2}, x_0 + \frac{m_2}{2}, y_0 + \frac{n_2}{2},)$, $n_3 = hg(t_0 + {}^{h}/_{2}, x_0 + \frac{m_2}{2}, y_0 + \frac{n_2}{2},)$

$m_4 = hf(t_0 + h, x_0 + m_3, y_0 + n_3,)$, $n_4 = hg(t_0 + h, x_0 + m_3, y_0 + n_3,)$.

The sum $(m_1 + m_2 + m_3 + m_4)/6$ and sum $(n_1 + n_2 + n_3 + n_4)/6$ can be interpreted as slope

In general, $x_1 = x_0 + \frac{1}{6}(m_1 + 2m_2 + 2m_3 + 2m_4)$ (4.5)

and

$y_1 = y_0 + \frac{1}{6}(n_1 + 2n_2 + 2n_3 + 2n_4)$ (4.6)

The two expressed equation as stated in equation (4.2) which is the Runge-Kutta formula.

Error = [exact solution – Runge-Kutta solution]. (4.7)

5.0 First Order Ordinary Differential Equation by Runge-Kutta Method Using C++ Programming Language.

The equation (4.3) will be transformed into computer programme to solve a given first order ordinary differential equation in (4.7). First Order Ordinary Differential Equation by Runge-Kutta Method in equation (4.3) using C++ Programming Language approach.

```
/*****************************************************************----------
* (Solve Y^1 = -Y + X/((1+X)*(1+X))  with Y(0)=1 for X0=0 up to  X1=1.0,          (4.8)
 exact solution is Y = 1 / (1+X).
Input Data

*     X
* -------------------
*   0.000000
*   0.050000
*   0.100000
*   0.200000
*   0.250000
*   0.300000
*   0.350000
```

```
*   0.400000
*   0.450000
*   0.500000
*   0.550000
*   0.600000
*   0.650000
*   0.700000
*   0.750000
*   0.800000
*   0.850000
*   0.900000
*   0.950000
*   1.000000
* ------------------------------------------              *

*****************************************************************/
#include <stdio.h>
#include <math.h>

const double H=0.05;  // integration step

double B[4],X[4],Y[4];
int I,K;
double C1,C2,C3,C4,ER,X1,YEX;

   // User defined function Y'=F(X,Y)
double F(double X, double Y) {
return -Y + X/((1.0+X)*(1.0+X));
   }

   // Exact solution Y=FX(X)
double FX(double X) {
return 1.0/(1.0+X);
   }

   // Runge-Kutta method to calculate first points only
void RK4() {
     C1=F(X[K],Y[K]);
     C2=F(X[K]+H/2,Y[K]+H/2*C1);
     C3=F(X[K]+H/2,Y[K]+H/2*C2);
     C4=F(X[K]+H,Y[K]+H*C3);
X[K+1]=X[K]+H;
Y[K+1]=Y[K]+H*(C1+2*C2+2*C3+C4)/6;
   }


void main()  {
```

```
  // Initial conditions
X[0]=0.0;  //starting X
  X1=1.0;    //ending X
Y[0]=1.0;  //initial Y
  // write header
printf("      X         Y       Y exact      Error   \n");
printf("  ----------------------------------------------\n");
  // write initial line
printf("%12.6f%12.6f%12.6f   0.000000\n", X[0],Y[0],Y[0]);
  // use Runge-Kutta to start
for (K=0; K<2; K++) {
RK4(); YEX=FX(X[K+1]); ER=fabs(YEX-Y[K+1]);
printf("%12.6f%12.6f%12.6f   %f\n", X[K+1],Y[K+1],YEX,ER);
  }
  // main integration loop
while (X[2]<X1) {
for (I=1; I<4; I++) B[I]=F(X[3-I],Y[3-I]);
X[3]=X[2]+H;
Y[3]=Y[2]+H*(23*B[1]-16*B[2]+5*B[3])/12;
   YEX=FX(X[3]); ER=fabs(Y[3]-YEX);
printf("%12.6f%12.6f%12.6f   %f\n", X[3],Y[3],YEX,ER);
for (K=0; K<3; K++) {
X[K]=X[K+1]; Y[K]=Y[K+1];
   }
  }
printf("  ----------------------------------------------\n");

}

// end of file adambash.cpp
```

Results

X	Y	Y exact	Error
0.000000	1.000000	1.000000	0.000000
0.050000	0.952381	0.952381	0.000000
0.100000	0.909091	0.909091	0.000000
0.150000	0.869525	0.869565	0.000040
0.200000	0.833265	0.833333	0.000068
0.250000	0.799910	0.800000	0.000090
0.300000	0.769125	0.769231	0.000106
0.350000	0.740623	0.740741	0.000117
0.400000	0.714160	0.714286	0.000125

0.450000	0.689525	0.689655	0.000131
0.500000	0.666533	0.666667	0.000134
0.550000	0.645026	0.645161	0.000135
0.600000	0.624865	0.625000	0.000135
0.650000	0.605926	0.606061	0.000134
0.700000	0.588103	0.588235	0.000133
0.750000	0.571298	0.571429	0.000131
0.800000	0.555428	0.555556	0.000128
0.850000	0.540416	0.540541	0.000125
0.900000	0.526194	0.526316	0.000121
0.950000	0.512703	0.512821	0.000118
1.000000	0.499886	0.500000	0.000114

Table 1 shows the generated results obtained from C++ programme on the First Order Differential Equation applying Runge Kutta Method and Exact Equation Method including error difference.

6.0 Discussion of Results

The X column are the values of that are inputted into both exact solution and Runge-Kutta method, the Y Column consist of the output from the computer approach to the Runge-Kutta method, the Y exact column are the results for the exact solution while the error column is the difference between the Runge-Kutta solution and the exact solution and was found to be negligible.

7.0 Conclusion: The computer programming approach gives faster and accurate solution to first order differential equation. When the exact solution is compared with the Runge Kutta method solution, the generated errors were found to be negligible. Therefore the computer programming approach organise the lengthy iteration which will take the manual system to accomplish with a bust time.

REFERENCES:

[1]Agam S.A and Odion P.O.: "A Perturbed Fourth Order Runge-Kutta Method for First Order Ordinary Differencial Equations" Journal of the Nigerian Association of Mathematical Physics vol 19: 143-148 (2011)

[2] AtajeromavwoE.J(2007). Introduction to Numerical Analysis with Programming Approach, Bridgeo Press, Agbor, Delta State. Nigeria Pp5-6

[3] SarkarS.K(2007).A Text Book of Discrete Mathematics,S. Chand & Company Ltd,RamNagar,New Delhi P633

[4] Savitch .W. (1996). Problem Solving with C^{++} the object of programming. Addison-WesleyPublishing Company Inc.California, U.S.A Pp 14-15.

[5] Christopher(1996). Oxford Concise Dictionary of Mathematics. Oxford University Press, New York PP4-5

[6] Jaggi(2006). Academic's Dictionary Mathematics Academic(India) Publishers,New-Delhi Pp9-10

[7] MumbaiH.O(2000). Logic Building with Javascipt.Aptech limited, Elite Auto House, 54A, Sir M. Vansanji Road, Andheri, India Pp16-17