Feb-Apr.2012, Vol.2.No.2, 901-909 e- ISSN: 2249 –1929

Journal of Chemical, Biological and Physical Sciences

An International Peer Review E-3 Journal of Sciences

Available online at www.jcbsc.org

Section C: Physical Sciences

CODEN (USA): JCBPAT

Research Article

Numerical Accuracy of Runge-Kutta Fourth Order Method

R. B. Srivastava*and Vikas Kant Pandey

Mathematics Department, L. K. P. G. College, Balrampur, U. P., India

Received: 12 January 2012; Revised: 24 January 2012; Accepted: 4 February 2012

ABSTRACT

Set of ordinary differential equations with boundary conditions has been solved by using Runge-Kutta fourth order method by developing a computer program. Values of y at the points of the interval [0, 1] have been calculated by adopting the stepwise of 0.01 and have been compared with the exact values of y at these points. Minimum, maximum and percentage errors have been calculated for each differential equation. It has been observed that Runge-Kutta fourth order method is accurate up to seven digits after the decimal point in some cases. In the worst case, it is accurate up to two digits after the decimal point.

Keywords: Differential equation, Numerical accuracy, Euler's method, Runge-Kutta fourth order method.

INTRODUCTION

There are several reasons that Euler's method is not recommended for practical use solution of ordinary differential equation of first order dy/dx = $f(x, y)$ with boundary conditions, among them, (i) the method is not very accurate when compared to other methods run at the equivalent stepsize, and (ii) neither is it very stable. As indicated in the error term, this symmetrization cancels out the first-order error term, making the method *second order*. In fact, it is called the *second-order Runge-Kutta* or *midpoint* method.

There are many ways to evaluate the right-hand side $f(x, y)$ that all agree to first order, but that have different coefficients of higher-order error terms. Adding up the right combination of these, one can eliminate the error terms order by order. That is the basic idea of the Runge-Kutta method. Abramowitz and Stegun[1], and Gear[2] give various specific formulas that derive from this basic idea. By far the most often used is the classical *fourth-order Runge-Kutta formula*, which has a certain sleekness of organization about it:

$$k_1 = hf(x_n, y_n)$$
$$k_2 = hf(x_n + \frac{h}{2}, y_n + \frac{k_1}{2})$$

$$k_3 = hf(x_n + \frac{h}{2}, y_n + \frac{k_2}{2})$$
$$k_4 = hf(x_n + h, y_n + k_3)$$
$$y_{n+1} = y_n + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} + O(h^5)$$

For many scientific users, fourth-order Runge-Kutta is not just the first word on ODE integrators, but the last word as well. In fact, it yields good result if one combines it with an adaptive stepsize algorithm. Bulirsch-Stoer or predictor-corrector methods can be very much more efficient for problems where very high accuracy is a requirement. An excellent discussion of the pitfalls in constructing a good Runge-Kutta code have been discussed by Shampine et. al.[3] , Rice [4], Bulirsch et. al.[5], Blanes et. al.[6] and Murugesan et. al.[7]

The Runge-Kutta method treats every step in a sequence of steps in identical manner. Prior behavior of a solution is not used in its propagation. This is mathematically proper, since any point along the trajectory of an ordinary differential equation can serve as an initial point. The fact that all steps are treated identically also makes it easy to incorporate Runge-Kutta into relatively simple schemes [8-12].

MATERIAL AND METHOD

Let an initial value problem be specified as follows [13-16]

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

Then, the Runge-Kutta fourth order method for this problem is given by the following equation:

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

Where

$$k_1 = f(t_n, y_n)$$

$$k_2 = f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_1\right)$$

$$k_3 = f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_2\right)$$

$$k_4 = f(t_n + h, y_n + hk_3)$$

Thus, the next value (y_{n+1}) is determined by the present value (y_n) plus the product of the size of the interval (h) and an estimated slope. The slope is a weighted average of slopes:[17-23]

- k_1 is the slope at the beginning of the interval;
- k_2 is the slope at the midpoint of the interval, using slope k_1 to determine the value of y at the point $t_n + h/2$ using Euler's method;
- k_3 is again the slope at the midpoint, but now using the slope k_2 to determine the y-value;
- k_4 is the slope at the end of the interval, with its y-value determined using k_3.

The RK4 method is a fourth-order method, meaning that the error per step is on the order of h^5, while the total accumulated error has order h^4.

We have solved the differential equations given in Table-1 at the points

$$0=x_0,\ x_1,\ x_2\ldots\ x_n=1,\ n=100,\ x_i - x_{i-1} = 0.01,\ i=1,\ 2,\ \ldots,\ 100$$

With the help of Runge-Kutta fourth order method by using the following computer program developed by us.

```
#include<conio.h>
#include<stdio.h>
#include<math.h>
//Runge-Kutta 4
void main(void)
{
  FILE *fpt;
  int i;
  float a,b,x0,y0,h,h1,k1,k2,k3,k4,k5;
  double ev, y1;
  double f(float x, float y);
  clrscr();
  fpt=fopen("vikas5.txt", "w");
  printf("Boundary conditions\n");
  printf("x0= ");
  scanf("%f", &x0);
  printf("y0= ");
  scanf("%f", &y0);
  a=0; b=1;
  h=(b-a)/100;
  fprintf(fpt,"a= %6.2f\n",a);
  fprintf(fpt,"b= %5.2f\n",b);
  fprintf(fpt,"h= %7.2f\n",h);
  //to write DE
  fprintf(fpt,"f(x,y)=(4x^2-2xy)/(1+x^2), x=0, y=0\n");
  fprintf(fpt,"Value of x            Exact value          Value by RK4\n");
  for(i=1; i<=100; i++)
  {
    h1=i*h;
    k1=h1*f(x0,y0);
```

```
    k2=h1*f(x0+h1/4,y0+k1/4);
    k3=h1*f(x0+3*h1/8,y0+3*k1/32+9*k2/32);
    k4=h1*f(x0+12*h1/13,y0+1932*k1/2197-7200*k2/2197+7296*k3/2197);
    k5=h1*f(x0+h1,y0+439*k1/216-8*k2+3680*k3/513-845*k4/4104);
    y1=y0+25*k1/216+1408*k3/2565+2197*k4/4104-k5/5;
    //exact value write h1 for x
    ev=(4*h1*h1*h1)/(3*(1+h1*h1));
    fprintf(fpt,"%7.2f  %22.14f %22.14f %22.14f\n",h1,ev,y1);
   }
  fclose(fpt);
 }
double f(float x, float y)
{
  double r;
  r=(4*x*x-2*x*y)/(1+x*x);
  return(r);
}
```

With the help of above computer program, we have calculated the value of y, exact value of y, difference between calculated and exact values of y and percentage error in the value of y. We have calculated the values of y in the interval [0, 1] by assuming the step size of 0.01. Percentage error in the value of y is defined as

$$PE_y = \frac{\text{Difference in the calculated and exact values of y}}{\text{Exact value of y}} \times 100$$

Table-1: Differential equations with boundary conditions along with exact solutions

S. No.	Differential Equation	Boundary Condition	Exact Solution
1	$dy/dx = y - x$	y=2 when x=0	$y=e^x+x+1$
2	$dy/dx = 1 + x$	y=1 when x=0	$y=x^2/2+x+1$
3	$dy/dx = x + y + 1$	y=1 when x=0	$y=3e^x-x-2$
4	$dy/dx = -x + y + 1$	y=1 when x=0	$y=e^x+x$
5	$dy/dx = x / y$	y=1 when x=0	$y=(x^2+1)^{1/2}$
6	$dy/dx = x + y$	y=1 when x=0	$y=2e^x-x-1$
7	$dy/dx = -(1 +x) y^2 / 2$	y=1 when x=0	$y=4/(x^2+2x+4)$
8	$dy/dx = 1 + y^2$	y=0 when x=0	$y=\tan x$
9	$dy/dx = -xy^2$	y=1 when x=0	$y=2/(x^2+2)$
10	$dy/dx = x^2 - y$	y=1 when x=0	$y=-e^{-x}+x^2-2x+2$
11	$dy/dx = 3x + y / 2$	y=1 when x=0	$y=13e^{x/2}-6x-12$
12	$dy/dx = 1 - y$	y=0 when x=0	$y=1-e^{-x}$
13	$dy/dx = 1 + y$	y=0 when x=0	$y=e^x-1$
14	$dy/dx = 1 / (2y)$	y=1 when x=0	$y=(x+1)^{1/2}$
15	$dy/dx = (4x^2 – 2xy) / (1 + x^2)$	y=0 when x=0	$y=4x^3/\{3(x^2+1)\}$

RESULT AND DISCUSSION

Accuracy of Solution of Differential equation dy/dx = y – x, y=2 when x=0, by Runge-Kutta fourth order method.

Differential equation has been solved by Runge-Kutta fourth order method with the help of computer program at different points of x. Exact values of y at these points have also been calculated. Exact value of y, calculated value of y by Runge-Kutta fourth order method and difference between them at different points is included in **Table-2.** A close look to this Table indicates the following

Maximum Error	=	0.00033308425264
Minimum Error	=	0.00000000012960
Maximum Percentage Error	=	0.00705943953222
Minimum Percentage Error	=	0.00000000635231

It is clear that the accuracy of Runge-Kutta fourth order method is up to 3 places after the decimal point. Graph between errors in the value of y calculated by Runge-Kutta fourth order method at different points is shown in **Graph-1.**

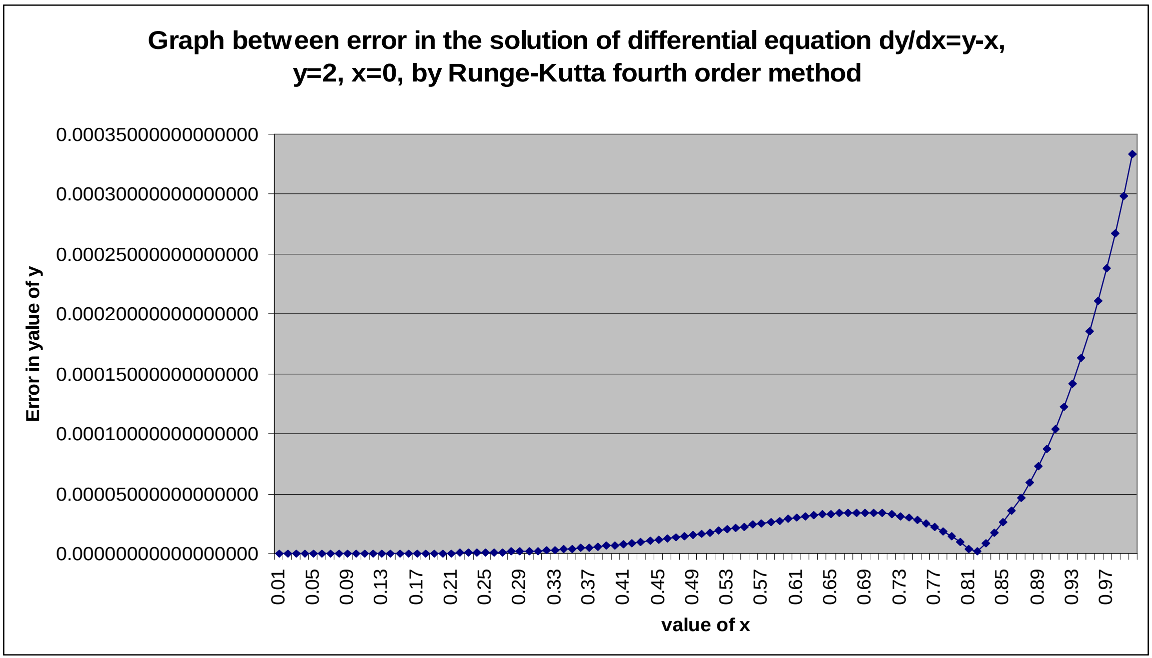
Table-2: Exact value of y, calculated value of y by Runge-Kutta fourth order method and difference between them at different points for differential equation dy/dx = y – x, y=2 when x=0

Value of x	Exact value of y for dy/dx = y – x, y=2 when x=0	Value of y calculated by Runge-Kutta fourth order method	Difference between exact value of y and value calculated by Runge-Kutta fourth order method
0.01	2.02005016663488	2.02005016685982	0.00000000022494
0.02	2.04020133912365	2.04020133925325	0.00000000012960
0.03	2.06045453259199	2.06045452780589	0.00000000478610
0.04	2.08081077236776	2.08081077342820	0.00000000106044
0.05	2.10127109026275	2.10127108927387	0.00000000098888
0.06	2.12183654378022	2.12183654261734	0.00000000116288
0.07	2.14250818187187	2.14250818201164	0.00000000013977
0.08	2.16328706394975	2.16328707694876	0.00000001299901
0.09	2.18417427559174	2.18417428742883	0.00000001183709
0.10	2.20517090552785	2.20517092544157	0.00000001991372
0.11	2.22627806919747	2.22627810534910	0.00000003615163
0.12	2.24749684587298	2.24749688033747	0.00000003446449
0.13	2.26882837312589	2.26882842200250	0.00000004887661
0.14	2.29027380013889	2.29027386482289	0.00000006468400
0.15	2.31183422339997	2.31183428739943	0.00000006399946
0.16	2.33351086321872	2.33351097284411	0.00000010962539
0.17	2.35530485522799	2.35530500954877	0.00000015432078
0.18	2.37721734609644	2.37721754868476	0.00000020258832
0.19	2.39924959238999	2.39924984033896	0.00000024794897
0.20	2.42140273167898	2.42140306028706	0.00000032860808
0.21	2.44367804531160	2.44367841415499	0.00000036884339

Value of x	Exact value of y for dy/dx = y – x, y=2 when x=0	Value of y calculated by Runge-Kutta fourth order method	Difference between exact value of y and value calculated by Runge-Kutta fourth order method
0.22	2.46607672790984	2.46607722292634	0.00000049501650
0.23	2.48859998569732	2.48860059120559	0.00000060550827
0.24	2.51124913813747	2.51124987421706	0.00000073607959
0.25	2.53402541668774	2.53402632862387	0.00000091193613
0.26	2.55693006476053	2.55693115368921	0.00000108892868
0.27	2.57996440667417	2.57996571955450	0.00000131288033
0.28	2.60312981510682	2.60313133016141	0.00000151505459
0.29	2.62642746852880	2.62642924608136	0.00000177755256
0.30	2.64985876555725	2.64986079205372	0.00000202649647
0.31	2.67342511976702	2.67342752353315	0.00000240376613
0.32	2.69712774733341	2.69713052952731	0.00000278219390
0.33	2.72096808856019	2.72097123238252	0.00000314382233
0.34	2.74494759916435	2.74495113494335	0.00000353577900
0.35	2.76906753417449	2.76907159062156	0.00000405644707
0.36	2.79332937685052	2.79333396808736	0.00000459123684
0.37	2.81773462633503	2.81773974954733	0.00000512321230
0.38	2.84228457769313	2.84229031245262	0.00000573475949
0.39	2.86698075844914	2.86698710639283	0.00000634794369
0.40	2.89182463823154	2.89183175251555	0.00000711428401
0.41	2.91681777614777	2.91682561894199	0.00000784279422
0.42	2.94196152254809	2.94197016297086	0.00000864042277
0.43	2.96725746607999	2.96726698379012	0.00000951771013
0.44	2.99270721242520	2.99271755142983	0.00001033900463
0.45	3.01831215487350	3.01832349572265	0.00001134084915
0.46	3.04407392954626	3.04408627173356	0.00001234218730
0.47	3.06999419011792	3.07000753394751	0.00001334382959
0.48	3.09607437412546	3.09608880885918	0.00001443473372
0.49	3.12231616660996	3.12233175866262	0.00001559205266
0.50	3.14872127070012	3.14873794946986	0.00001667876974
0.51	3.17529116952768	3.17530912494915	0.00001795542147
0.52	3.20202759854326	3.20204668269868	0.00001908415542
0.53	3.22893223140147	3.22895247717233	0.00002024577086
0.54	3.25600675857742	3.25602835440031	0.00002159582289
0.55	3.28325305045031	3.28327580887323	0.00002275842292
0.56	3.31067250685421	3.31069659501837	0.00002408816416
0.57	3.33826703163354	3.33829234570671	0.00002531407317
0.58	3.36603838425304	3.36606500001213	0.00002661575909
0.59	3.39398834186033	3.39401597622890	0.00002763436857
0.60	3.42211869946367	3.42214760475339	0.00002890528972
0.61	3.45043143941433	3.45046140015357	0.00002996073924
0.62	3.47892805547877	3.47895915055135	0.00003109507258
0.63	3.50761056554282	3.50764246426130	0.00003189871848
0.64	3.53648083787046	3.53651363241974	0.00003279454928

Value of x	Exact value of y for dy/dx = y – x, y=2 when x=0	Value of y calculated by Runge-Kutta fourth order method	Difference between exact value of y and value calculated by Runge-Kutta fourth order method
0.65	3.56554075950198	3.56557414646734	0.00003338696536
0.66	3.59479223644277	3.59482620826125	0.00003397181848
0.67	3.62423719385382	3.62427160610348	0.00003441224966
0.68	3.65387775350127	3.65391222490088	0.00003447139961
0.69	3.68371552610550	3.68375003527828	0.00003450917278
0.70	3.71375267154374	3.71378684756700	0.00003417602326
0.71	3.74399119354436	3.74402481613219	0.00003362258783
0.72	3.77443311597351	3.77446579898890	0.00003268301539
0.73	3.80508048303754	3.80511193741948	0.00003145438194
0.74	3.83593554401950	3.83596542160413	0.00002987758463
0.75	3.86700001661267	3.86702789680534	0.00002788019267
0.76	3.89827619056788	3.89830162331142	0.00002543274354
0.77	3.92976619351715	3.92978849283203	0.00002229931488
0.78	3.96147217447555	3.96149109122116	0.00001891674561
0.79	3.99339630405606	3.99341102589873	0.00001472184267
0.80	4.02554077468669	4.02555064856425	0.00000987387756
0.81	4.05790799442008	4.05791232240246	0.00000432798238
0.82	4.09049981413996	4.09049789739979	0.00000191674017
0.83	4.12331868530099	4.12330945347368	0.00000923182731
0.84	4.15636688980590	4.15634970411106	0.00001718569484
0.85	4.18964673249091	4.18962077102814	0.00002596146277
0.86	4.22316054135619	4.22312433082341	0.00003621053278
0.87	4.25691086967432	4.25686382100596	0.00004704866836
0.88	4.29089969015277	4.29084057921786	0.00005911093491
0.89	4.32512960214995	4.32505687468233	0.00007272746762
0.90	4.35960302867358	4.35951543682732	0.00008759184626
0.91	4.39432241708300	4.39421840421637	0.00010401286663
0.92	4.42929023933389	4.42916812659704	0.00012211273685
0.93	4.46450920289863	4.46436762826600	0.00014157463263
0.94	4.49998140984161	4.49981847737151	0.00016293247010
0.95	4.53570961657085	4.53552385239224	0.00018576417861
0.96	4.57169639592452	4.57148554123516	0.00021085468936
0.97	4.60794434659819	4.60770638089547	0.00023796570272
0.98	4.64445609340476	4.64418876232111	0.00026733108365
0.99	4.68123428753690	4.68093534070729	0.00029894682961
1.00	4.71828182845904	4.71794874420640	0.00033308425264

Graph-1: Graph between errors in the value of y calculated by Runge-Kutta fourth order method at different points for differential equation DE1



In above manner, the remaining differential equations have been solved with the help of developed computer program.

CONCLUSION

Maximum error in the value of y and maximum percentage error in it for each differential equation have been calculated and included in **Table-3.** Average of the maximum error is 0.00116602888541 which indicates that the Runge-Kutta fourth order method is accurate at least up to two places after the decimal point in worst cases. In some cases, the accuracy of Runge-Kutta fourth order method has been found to be accurate up to seven places after the decimal point. Value of average percentage error has been found to be 0.13989860729377.

Table-3: Maximum Error and maximum percentage error in the value of y calculated by Runge-Kutta fourth order method

Differential Equation	Maximum Error in the value of y calculated by Runge-Kutta fourth order method	Maximum Percentage Error in the value of y calculated by Runge-Kutta fourth order method
DE1	0.00033308425264	0.00705943953222
DE2	0.00000005796213	0.00000243769418
DE3	0.00099928502345	0.01938535357238
DE4	0.00033303656893	0.00895673282162
DE5	0.00005527723547	0.00390869080461
DE6	0.00066616850529	0.01938472764644
DE7	0.00023466931125	0.04106712946875
DE8	0.00273229267543	0.19758704060591
DE9	0.00429651245736	0.64447686860400
DE10	0.00231286876084	0.36589045056946
DE11	0.00021728685466	0.00632866373528
DE12	0.00249483133102	0.39467650532414

Differential Equation	Maximum Error in the value of y calculated by Runge-Kutta fourth order method	Maximum Percentage Error in the value of y calculated by Runge-Kutta fourth order method
DE13	0.00033321188559	0.01939215558654
DE14	0.00002414770471	0.00170750057505
DE15	0.00245770275244	0.36865541286600
Average	**0.00116602888541**	**0.13989860729377**

REFERENCES

1. M.Abramowitz and I.A.Stegun, 1964, *Handbook of Mathematical Functions*, Applied Mathematics.
2. Series, Volume 55 (Washington: National Bureau of Standards; reprinted 1968 by Dover Publications, New York), Gear, C.W. 1971, *Numerical Initial Value Problems in Ordinary Differential Equations* (Englewood Cliffs, NJ: Prentice-Hall), Chapter 2.
3. L.F.Shampine and H.A.Watts, 1977, in *Mathematical Software III*, J.R. Rice, ed. (New York: Academic Press), pp. 257–275; 1979, *Applied Mathematics and Computation*, vol. 5, pp. 93–121.
4. J.R.Rice, 1983, *Numerical Methods, Software, and Analysis* (New York: McGraw-Hill).
5. J. Stoer and R. Bulirsch, 1980, *Introduction to Numerical Analysis* (New York: Springer-Verlag)
6. S.Blanes,F.; Casas, J. Ros, *Special issue: Themes in geometric integration. Appl. Numer. Math.*,2001, **39,** 3-4, 245.
7. K.Murugesan, Dhayabaran, D. Paul E.C. Amirtharaj, Henry Evans, J. David , *Int. J. Comput. Math.*, 2002, **79,** 2, 247.
8. G.Papageorgiou, I Famelis, *Comput. Math. Appl.,2001*, **42,** 8-9, 1165.
9. Ling Xiang, Tu, Shan-Tung; Gong, Jian-Ming; *International Journal of Pressure Vessels and Piping*, 2000, **77,** 5, 243.
10. Chou, Lin-Yi; Sharp, P. W.; *J. Appl. Math. Decis. Sci., 2000*, **4,** 2, 143.
11. Tsitouras, Ch.; *Celestial Mech. Dynam. Astronom.*, 1999,**74,** 4, 223.
12. Burnton, Christopher; Scherer, Rudolf; *BIT, 1998,* **38,** 1, 12.
13. Peter Albrecht; *SIAM Journal on Numerical Analysis*, 1996, **33,** 5, 1712.
14. Dumitras, Daria Elena; *Automat. Comput. Appl. Math., 1997*, **6,** 2, 42.
15. J.H.Verner, *Applied Numerical Mathematics*,1996, **22,** 1-3, 345.
16. P.Bogacki, L.F. Shampine, Source; *Computers & Mathematics with Applications*,1996, **32,** 6, 15.
17. C.Fredebeul, D. Kornmaier, M.W. Muller, *Journal of Computational and Applied Mathematics*, 2002,**144,** 1-2, 187.
18. A.B.Sideridis, T.E. Simos, *J. Comput. Appl. Math.*,1992 **44,** 2, 235.
19. Daniel Okunbor, D. Robert Skeel; *SIAM Journal on Numerical Analysis*, 1992, **29,** 2, 521.
20. Dumitras, Daria Elena; *Automat. Comput. Appl. Math., 1993,* **2**, 2, 139.
21. J.H.Verner, SIAM *J. Numer. Anal.*, 1991, **28,** 2, 496.
22. J.H.Verner, SIAM *Journal on Numerical Analysis*, 1990, **27,** 5, 1332.
23. P.J.Van der Houwen,B.P. Sommeijer, *SIAM Journal on Numerical Analysis*,1989, **26,** 2, 414.