

Numerical Solution of Ordinary Differential Equation using Scilab

¹P.Yaswanth, ²L.Naveen

¹PG Student, ²Assistant Professor

Department of Mathematics,

Dr. SNS Rajalakshmi College of Arts and Science, Coimbatore, Tamil Nadu, India.

Abstract: Numerical methods are developed to solve certain types of linear and non-linear ordinary differential equations by and its application. $dy/dx = f(x,y)$ with the initial condition $y(x_0) = y_0$ and we launch differing numerical method of Ordinary Differential Equation using Scilab Programming. We compare the differential equations with the exact solution and the approximation solution. We rectified percentage of error calculation with the approximate solution form the exact solution using initial value problems. In this paper we discuss about the general first order differential equation (ODE).

Index Terms: Ordinary Differential Equation, Initial Condition, Step – by – Step Method, Initial value Problem, Scilab.

I. INTRODUCTION

In Mathematics, Ordinary differential equations are used for mathematical modelling in science and engineering.

Numerical methods are used for solving mathematical problems that are formulated in science and engineering however it is used to solve the first order differential equation but the initial value differential equation where the solution is obtained. Numerical method for ordinary differential equation is method used to find numerical approximations to the solution of order differential equation (ODEs) where it is difficult or complicated to obtain exact solution. There are several numerical methods to compute the approximate solution and the exact solution. In this paper we compare Taylor Series, Euler Method, Modified Euler Method, Improved Euler Method and Runge – Kutta Method with the exact solution using Scilab Programming.

Scilab is a high level numerically oriented programming language to built a function for all of the most numerical method.

Here we consider the general form of first order differential equation as

$$y' = f(x, y)$$

Euler method uses step – by – step process whereas Runge – Kutta method uses Multi – steps process to solve ordinary differential equation with the initial conditions

$$\frac{dy}{dx} = f(x, y) \text{ and } y(x_0) = y_0$$

In all of the Euler and Runge – kutta method we consider the interval length 'h' and the solution of these methods can be tabulated by the variable x & y.

We shall describe the various numerical methods for solving the initial value problem and compute the solution using Scilab.

II. NUMERICAL METHOD

Taylor Series Method

The numerical solution of the equation $y' = f(x, y)$ give the initial condition $y(x_0) = y_0$. F is a function of two variable x, y and $f(x_0, y_0)$ is known point on the solution curve.

If the existence of all higher order partial derivatives is assumed for y at $x = x_0$, then by Taylor series the value of y at any neighbouring point $x + h$ can be written as

$$y(x_0 + h) = y(x_0) + h * y'(x_0) + \frac{h^2}{2!} y''(x_0) + \frac{h^3}{3!} y'''(x_0) + \dots$$

Similarly higher derivatives of y at x_0 also can be computed by making use of relation.

$$y' = f(x, y)$$

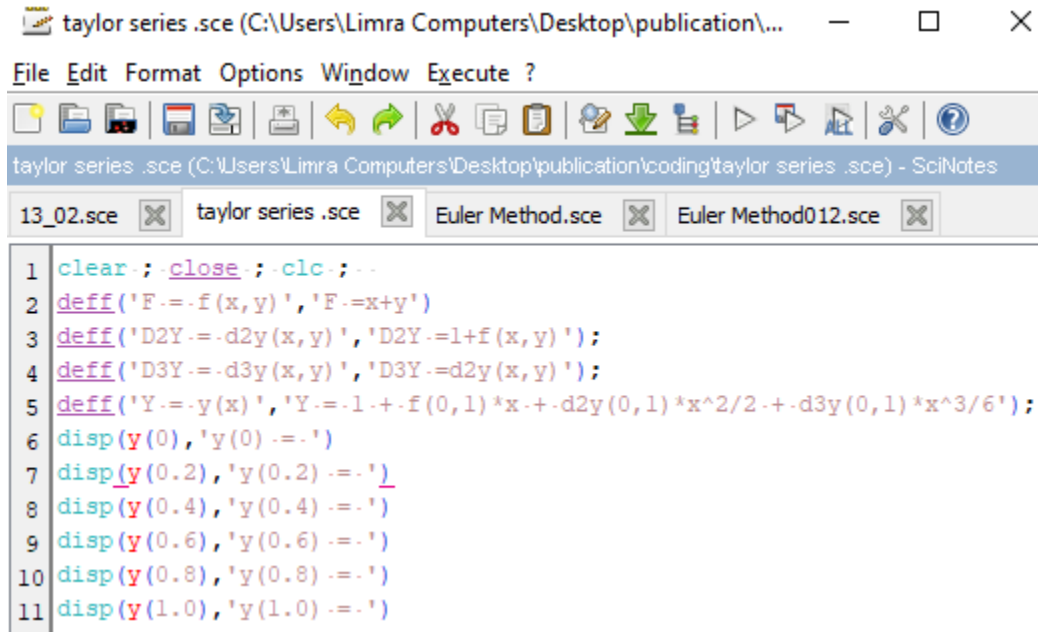
$$y'' = f_x + f_y y'$$

$y''' = f_{xx} + 2f_{xy}y' + f_{yy}y'^2 + f_y y''$ and so on.

Example

Solve $\frac{dy}{dx} = x + y$, given $y(1) = 0$, and get $y(1)$ by Taylor series method. Compare your result with the explicit solution.

TAYLOR SERIES METHOD SOLUTION USING SCILAB CODING



```

1 clear; close; clc; --
2 deff('F=-f(x,y)', 'F=-x+y')
3 deff('D2Y=-d2y(x,y)', 'D2Y=-1+f(x,y)');
4 deff('D3Y=-d3y(x,y)', 'D3Y=-d2y(x,y)');
5 deff('Y=-y(x)', 'Y=-1+.f(0,1)*x-.d2y(0,1)*x^2/2-.d3y(0,1)*x^3/6');
6 disp(y(0), 'y(0) .-.')
7 disp(y(0.2), 'y(0.2) .-.')
8 disp(y(0.4), 'y(0.4) .-.')
9 disp(y(0.6), 'y(0.6) .-.')
10 disp(y(0.8), 'y(0.8) .-.')
11 disp(y(1.0), 'y(1.0) .-.')

```

Fig. 1

Euler Method**1. Euler Method**

The numerical solution of the equation $y' = f(x, y)$ gives the initial condition $y(x_0) = y_0$. f is a function of two variable x, y and $f(x_0, y_0)$ is known point on the solution curve.

Let us take the point $= x_0, x_1, x_2, \dots$, where $x_i - x_{i-1} = h$,
i.e., $x_i = x_0 + ih$, $i = 0, 1, 2, 3, \dots$

We require the value of y of the curve at $x = x_1$.

The equation of tangent at $p_0(x_0, y_0)$ to the curve is

$$\begin{aligned}
 y - y_0 &= y'_{(x_0, y_0)}(x - x_0) \\
 &= f(x_0, y_0) \cdot (x - x_0) \\
 y &= y_0 + f(x_0, y_0) \cdot (x - x_0) \\
 y_1 &= y_0 + f(x_0, y_0) \cdot (x_1 - x_0) \\
 y_1 &= y_0 + h y_0' \\
 y_{n+1} &= y_n + h f(x_n, y_n); n = 0, 1, 2 \dots
 \end{aligned}$$

Example:-

Solve $\frac{dy}{dx} = x + y$, given $y(1) = 0$, and get $y(1)$ by Euler Method. Compare your result with the explicit solution.

EULER METHOD SOLUTION USING SCILAB CODING

```

clc;clear;close;
deff('y=f(x,y)', 'y=x+y')
y=1;
x=0;
h=0.2;
disp('x.....y')
for i=1:6
printf('\n%g=%g\n', (i-1)*h, y(i))
y(i+1)=y(i)+h*f(i-1)*h, y(i))
end

```

2. Improved Euler method

In the interval (x_0, x_1) , by previous Euler's method, we approximate the curve by the tangent p_0A .

$$\therefore y_0^{(1)} = y_0 + h f(x_0, y_0)$$

$Q_1(x_1, y_0^{(1)})$ Let Q_1C be the line at Q_1 whose slope is $f(x_1, y_1^{(1)})$. Now take the average of the slopes at P_0 and Q_1 i.e.,,
 $\frac{1}{2}[f(x_0, y_0) + f(x_1, y_1^{(1)})]$

Now draw a line P_0D through $P_0(x_0, y_0)$ with this as the slope.

That is $y - y_0 = \frac{1}{2}[f(x_0, y_0) + f(x_1, y_1^{(1)})](x - x_0)$

This line intersect $x=x_1$ at

$$y_1 = y_0 + \frac{1}{2}h[f(x_0, y_0) + f(x_1, y_1^{(1)})]$$

$$y_1 = y_0 + \frac{1}{2}h[f(x_0, y_0) + f(x_1, y_0 + hf(x_0, y_0))]$$

$$y_{n+1} = y_n + \frac{1}{2}h[f(x_n, y_n) + f(x_n + h, y_n + hf(x_n, y_n))]$$

Example:-

Solve $\frac{dy}{dx} = x + y$, given $y(1) = 0$, and get $y(1)$ by Improved Euler Method. Compare your result with the explicit solution.

IMPROVED EULER METHOD SOLUTION USING SCILAB CODING

```
clc;clear;close;
deff('y=f(x,y)', 'y=x+y')
y=1;
x=0;
h=0.2;
printf('\n\n.By-Improved-Euler-Method\n')
for i=1:6
y(i+1)=y(i)+h*f((i-1)*h+h/2,y(i)+h*f((i-1)*h,y(i))/2)
y(i+1)=y(i)+h/2*f((i-1)*h,y(i)+f((i-1)*h+h,y(i)+h*f((i-1)*h,y(i))))
end
```

3. Modified Euler method

Recall the Euler governing equation $y_{n+1} = y_n + hf(x_n, y_n)$

A better estimation of the slope from (x_n, y_n) to (x_{n+1}, y_{n+1}) would be

$$y_{n+1} = y_n + \frac{h}{2}\{f(x_n, y_n) + f(x_{n+1}, y_{n+1})\},$$

But we don't know y_{n+1} .

However, we can estimate it by using the Euler's method, to give a two stage Predictor – corrector algorithm that is called the Modified Euler's Method.

Example:-

Solve $\frac{dy}{dx} = x + y$, given $y(1) = 0$, and get $y(1)$ by Modified Euler Method. Compare your result with the explicit solution.

MODIFIED EULER METHOD SOLUTION USING SCILAB CODING

```
clc;clear;close;
deff('y=f(x,y)', 'y=x+y')
y=1;
x=0;
h=0.2;
printf('\n\n.By-Modified-Euler-Method\n')
for i=1:6
y(i+1)=y(i)+h*f((i-1)*h+h/2,y(i)+h*f((i-1)*h,y(i))/2)
end
```

Runge – Kutta Method

Runge Kutta Methods, the derivative of higher order are not require only the given function value at different point. Since the derivation of fourth order Runge – Kutta method is tedious, we will derive Runge – Kutta method of second order.

Second Order R.K algorithm

$$K_1 = hf(x, y)$$

$$K_2 = hf(x + \frac{1}{2}h, y + \frac{1}{2}k_1)$$

$$\Delta y = k_2 \text{ where } h = \Delta x.$$

Third Order R.K algorithm

$$K_1 = hf(x, y)$$

$$K_2 = hf(x+\frac{1}{2}h, y+\frac{1}{2}k_1)$$

$$K_3 = hf(x+h, y+2k_2 - k_1)$$

$$\Delta y = \frac{1}{6}(k_1+4k_2+k_3)$$

Fourth Order R.K algorithm

$$K_1 = hf(x,y)$$

$$K_2 = hf(x+\frac{1}{2}h, y+\frac{1}{2}k_1)$$

$$K_3 = hf(x+\frac{1}{2}h, y+\frac{1}{2}k_2)$$

$$K_4 = hf(x+h,y+k_3)$$

$$\Delta y = \frac{1}{6}(k_1+2k_2+2k_3+k_4)$$

Example:-

Solve $\frac{dy}{dx} = x + y$, given $y(1) = 0$, and get $y(1)$ by **Runge – Kutta** Method. Compare your result with the explicit solution.

RUNGE – KUTTA METHOD SOLUTION USING SCILAB CODING

```

clc;clear;close;
deff('y=f(x,y)', 'y=x+y')
printf('the value of y1')
y=1; x=0; h=0.2;
k1=h*f(x,y);
k2=h*f(x+h/2,y+k1/2);
k3=h*f(x+h/2,y+k2/2);
k4=h*f(x+h,y+k3);
disp(k4,'k4=',k3,'k3=',k2,'k2=',k1,'k1=')
y1=y+[(k1+2*k2 +2*k3+k4)/6]
printf('\ny(0.2)=%.8f\n',y1)
end
    
```

FIRST ORDER ORDINARY DIFFERENTIAL EQUATION SOLUTION FOR TAYLOR, EULER, MODIFIED, IMPROVED METHOD, RUNGE – KUTTA AND EXACT SOLUTION

Table 1

X	Taylor Series Method	Euler Method	Modified Euler Method	Improved Euler Method	Runge – Kutta Method	Exact Solution
0	1	1	1	1	1	1
0.2	1.2425	1.2	1.24	1.24	1.2428	1.2428
0.4	1.5813	1.48	1.5768	1.5768	1.5836	1.5836
0.6	2.032	1.856	2.0317	2.0317	2.0442	2.0442
0.8	2.6105	2.3472	2.6307	2.6307	2.6510	2.6510
1.0	3.3333	2.9766	3.4054	3.4054	3.4365	3.4366

Error value

Table 2

X	Percentage of error				
	Taylor Series Method	Euler Method	Modified Euler Method	Improved Euler Method	Runge – Kutta Method
0	0	0	0	0	0
0.2	0.02	3.44	0.23	0.23	0
0.4	0.14	6.54	0.43	0.43	0
0.6	0.60	9.21	0.61	0.61	0
0.8	1.53	11.46	0.77	0.77	0
1.0	3	13.39	0.91	0.91	0.006

III CONCLUSION

In this paper we obtain the approximate solution & Exact solution for the given ordinary differential equations with Initial value condition using Scilab for Taylor series, Euler's Method, Improved Euler's Method, Modified Euler's Method and Runge – Kutta Method. Finally, the results are Calculated and tabulated with approximation solution, Exact Solution and Percentage of error using Scilab Coding. The Runge – Kutta Method is more effective while comparing to other numerical methods since we get the minimum percentage error.

REFERENCES

- [1] E. Balaguruswamy , "Scilab Textbook Companion for Numerical Methods" S. Pal, "Scilab Textbook Companion for Numerical Methods: Principles, Analysis, And Algorithms", ISBN: 9780195693751.
- [2] B. Jayapriya, M. Muthuselvi, "Exact and Numerical Solution of ordinary differential equations using Highly Improved Euler's method and MATLAB", Volume 4, Issue 10, 2018, ISSN: 2394-4099.
- [3] N. M. M. Yusop, M. K. Hasan, and M. Rahmat, "Comparison New Algorithm Modified Euler in Ordinary Differential Equation Using Scilab Programming", Vol. 3, No. 3, August 2015, DOI: 10.7763/LNSE.2015.V3.190.
- [4] Gadamsetty Revathi, Assistant Professor in Mathematics, "Numerical Solution Of Ordinary Differential Equations And Applications", Volume-3, Issue-2, Feb.-2017, ISSN: 2394-7926.
- [5] Zulzamri sallah, "Ordinary Differential Equations (ode) using euler's technique and Scilab programming", ISBN: 978-1-61804-106-7.

