Indian Institute of Technology Hyderabad

Teaching Learning Center IIT Hyderabad

Numerical Solution of Ordinary Differential Equations

G V V Sharma*

***Abstract*—Through examples, this manual discusses the numerical solution of ordinary differential equations (ODE) by Taylor series method, Euler's Method, Euler's modified method and Runge-Kutta Methods. Python codes are provided for all these methods.**

1 Taylor Series Method

Definition 1. The Taylor series of $f(x)$ that is infinitely differentiable at a is the power series

$$f(x) = f(a) + \frac{f^1(a)}{1!}(x-a) + \frac{f^2(a)}{2!}(x-a)^2 + \frac{f^3(a)}{3!}(x-a)^3 + \dots \quad (0.1)$$

where $f^n(a)$ is the nth derivative of f at a.

Problem 1. Find the 2nd and 3rd derivative of y using the following differential equation.

$$y^{(1)} = 1 - xy, \quad y(0) = 1 \quad (1.1)$$

where $y^{(1)} = \frac{dy}{dx}$.

Solution: From (1.1), through successive differentiation,

$$y^{(2)} = -xy^{(1)} - y \quad (1.2)$$
$$y^{(3)} = -xy^{(2)} - 2y^{(1)} \quad (1.3)$$

Problem 2. Express (1.1) as a difference equation using the Taylor series method. Assume a step size h.

Solution: Substituting $x = a + h$ in (0.1) [1],

$$f(a+h) = f(a) + \frac{f^1(a)}{1!}h + \frac{f^2(a)}{2!}h^2 + \frac{f^3(a)}{3!}h^3 + \dots \quad (2.1)$$

*The author is with the Department of Electrical Engineering, IIT, Hyderabad 502285 India e-mail: gadepall@iith.ac.in.

From (0.1), Let $f(a) = y_n, f(a+h) = y_{n+1}$. From (1.1) - (2.1), the desired difference equation is

$$y_{n+1} = y_n + \frac{y_n^{(1)}}{1!}h + \frac{y_n^{(2)}}{2!}h^2 + \frac{y_n^{(3)}}{3!}h^3 + \dots \quad (2.2)$$

where

$$x_0 = 0, y_0 = 1 \quad (2.3)$$
$$x_{n+1} = x_n + h \quad (2.4)$$
$$y_n^{(1)} = 1 - x_n y_n \quad (2.5)$$
$$y_n^{(2)} = -x_n y_n^{(1)} - y_n \quad (2.6)$$
$$y_n^{(3)} = -x_n y_n^{(2)} - 2y_n^{(1)} \quad (2.7)$$

Problem 3. Compute and plot y for $x \in (0,5)$ with 25 subintervals using (2.2).

Solution: The following script plots the output in Fig. 3

```
import numpy as np
import matplotlib.pyplot as plt

a = 0
b = 5
n = 25
x = np.linspace(a,b,n)
h = (b-a)/(n+1) #interval
y=[]
tempy = 1
for i in range(n):
        y.append(tempy)
        yn1 = 1 - x[i]*tempy
        yn2 = -x[i]*yn1 -tempy
        yn3 = -x[i]*yn2 - 2*yn1
        tempy = tempy + yn1*h+yn2*
           h**2/2 + yn3*h**3/6


#Plotting
plt.plot(x,y)
plt.grid()
plt.xlabel('$x$')
```

```
plt.ylabel('$y$')

#Comment the following line
#plt.savefig('../figs/taylor.eps')
plt.show()
```

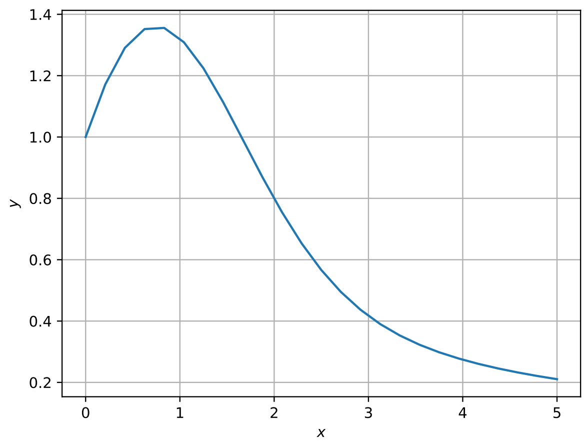

Fig. 3: Taylor series method.

2 Euler's Method

Problem 4. Formulate a difference equation for (1.1) using the Euler method.

Solution: (1.1) can be expressed as [2]

$$\frac{y_{n+1} - y_n}{h} \approx 1 - x_n y_n, \tag{4.1}$$

$$\implies y_{n+1} = y_n + h\left(1 - x_n y_n\right), \quad y_0 = 1 \tag{4.2}$$

using the definition of the derivative.

Problem 5. Compute and plot y using (4.1).

Solution: The following script plots the output in Fig. 5

```
import numpy as np
import matplotlib.pyplot as plt

a = 0
b = 5
n = 25
x = np.linspace(a,b,n)
h = (b-a)/(n+1) #interval
y=[]
tempy = 1
for i in range(n):
        y.append(tempy)
        tempy = tempy*(1-h*x[i]) +
            h

#Plotting
plt.plot(x,y)
plt.grid()
plt.xlabel('$x$')
plt.ylabel('$y$')

#Comment the following line
#plt.savefig('../figs/euler.eps')
plt.show()
```

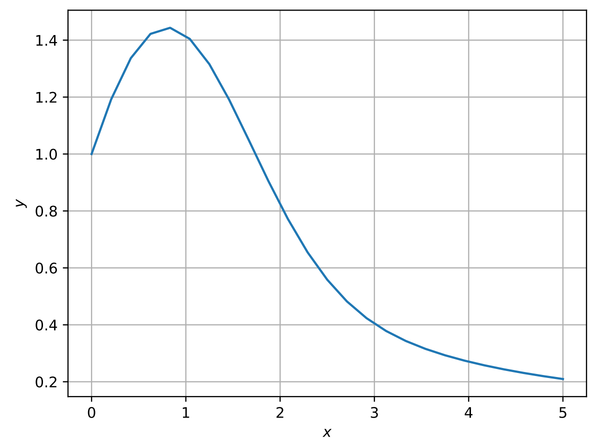

Fig. 5: Euler's method

3 Euler's Modified Method

Problem 6. Show that the differential equation

$$y^{(1)}(t) = f\left(t, y(t)\right) \tag{6.1}$$

results in the approximation

$$y(t+h) \approx y(t) + hf\left(t + \frac{h}{2}, y(t) + \frac{h}{2} f(t, y(t))\right) \tag{6.2}$$

for small values of h.

Proof. Using the definition of the deritave,

$$y\left(t+h\right) \approx y(t) + hy^{(1)}\left(t\right) \tag{6.3}$$

$$y^{(1)}\left(t + \frac{h}{2}\right) \approx y(t) + \frac{h}{2} y^{(1)}\left(t\right) \tag{6.4}$$

$$= y(t) + \frac{h}{2} f\left(t + \frac{h}{2}, y(t) + \frac{h}{2} f(t, y(t))\right) \tag{6.5}$$

using (6.1). From Fig. 6 [3],

$$y^{(1)}(t) \approx y^{(1)}\left(t + \frac{h}{2}\right) \tag{6.6}$$

resulting in (6.2) by substituting (6.5) in (6.3). □

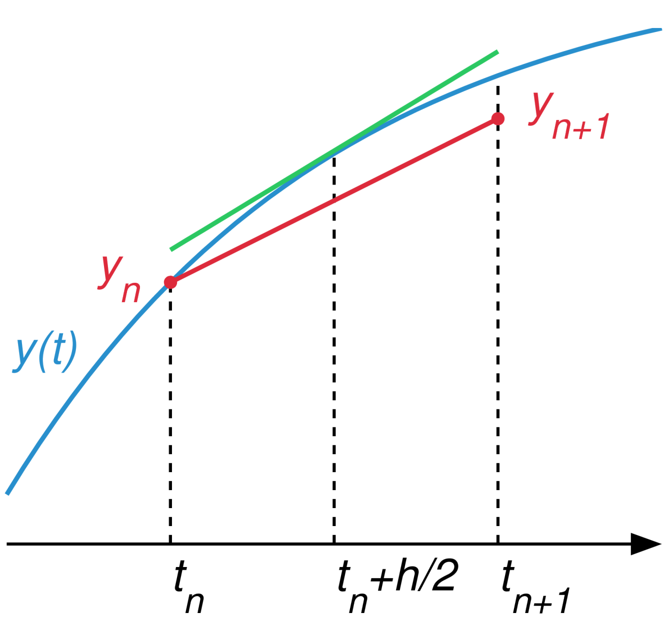

Fig. 6

Problem 7. Formulate a difference equation for the modified Euler method.

Solution: From (6.2) [3],

$$y_{n+1} = y_n + hf\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}f(x_n, y_n)\right) \tag{7.1}$$

$$x_{n+1} = x_n + h \tag{7.2}$$

Problem 8. Compute and plot y using (7.1).

Solution: The following script plots the output in Fig. 8

```
import numpy as np
import matplotlib.pyplot as plt

a = 0
b = 5
n = 25
x = np.linspace(a,b,n)
h = (b-a)/(n+1) #interval
y=[]
tempy = 1
for i in range(n):
        y.append(tempy)
        tempy = tempy + h*(1-x[i]*
           tempy)

#Plotting
plt.plot(x,y)
plt.grid()
plt.xlabel('$x$')
plt.ylabel('$y$')

#Comment the following line
plt.savefig('../figs/
   euler_modified.eps')
plt.show()
```

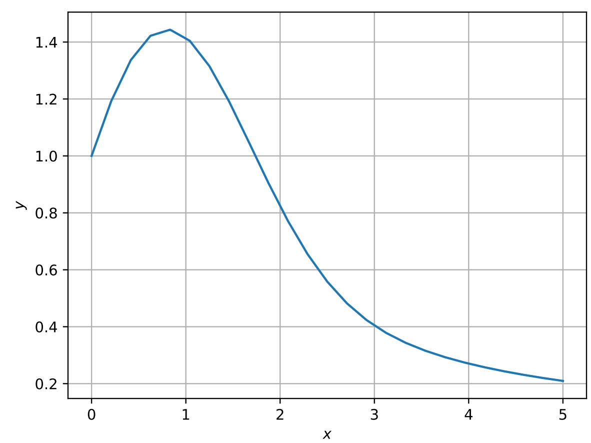

Fig. 8: Euler's modified method.

4 The Runge-Kutta method

Problem 9. Obtain the difference equation for (6.1) using the Runge-Kutta method.

Solution: The desired equation is given by [4]

$$y_{n+1} = y_n + \tfrac{h}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right), \tag{9.1}$$

$$x_{n+1} = x_n + h, \tag{9.2}$$

where

$$k_1 = f(x_n, y_n), \tag{9.3}$$

$$k_2 = f\left(x_n + \frac{h}{2}, y_n + h\frac{k_1}{2}\right), \tag{9.4}$$

$$k_3 = f\left(x_n + \frac{h}{2}, y_n + h\frac{k_2}{2}\right), \tag{9.5}$$

$$k_4 = f\left(x_n + h, y_n + hk_3\right). \tag{9.6}$$

Problem 10. Compute and plot y using (9.1).

Solution: The following script plots the output in Fig. 10

```
import numpy as np
import matplotlib.pyplot as plt


def f(x,y):
        return 1-x*y


a = 0
b = 5
n = 25
x = np.linspace(a,b,n)
h = (b-a)/(n+1) #interval
y=[]
tempy = 1
for i in range(n):
        y.append(tempy)
        k1 = f(x[i],tempy)
        k2 = f(x[i]+h/2,tempy+ h*
           k1/2)
        k3 = f(x[i]+h/2,tempy+h*k2
           /2)
        k4 = f(x[i]+h,tempy+h*k3)
        tempy = tempy + h/6*(k1+2*
           k2+2*k3+k4)

#Plotting
plt.plot(x,y)
plt.grid()
plt.xlabel('$x$')
plt.ylabel('$y$')

#Comment the following line
plt.savefig('../figs/runge.eps')
plt.show()
```

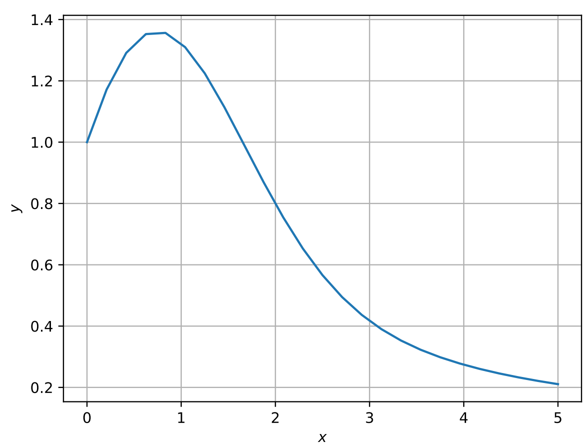

Fig. 10: Runge-Kutta method.

References

[1] [Online]. Available: http://mathfaculty.fullerton.edu/mathews/n2003/taylorde/TaylorDEMod/Links/TaylorDEMod_lnk_2.html

[2] Wikipedia. [Online]. Available: https://en.wikipedia.org/wiki/Euler_method

[3] ——. [Online]. Available: https://en.wikipedia.org/wiki/Midpoint_method

[4] ——. [Online]. Available: https://en.wikipedia.org/wiki/Runge%E2%80%93Kutta_methods