Presentation

Mohit
EE24BTECH11041

9 JAN 2025

Problem Statement

Solve the differential equation given below with initial conditions $x = 2$ and $y = 0$ and plot a graph.

$$x(x^2 - 1)\frac{dy}{dx} = 1 \tag{2.1}$$

Solution

Rearranging the Equation,

$$dy = \frac{dx}{x(x^2-1)} \tag{3.1}$$

Integration: Integrating on both sides.

$$\int dy = \int \frac{dx}{x(x^2-1)} \tag{3.2}$$

$$\int dy = \int \frac{dx}{x^3(1-\frac{1}{x^2})} \tag{3.3}$$

Subsituting ,

$$1 - \frac{1}{x^2} = t \tag{3.4}$$

Solution

Differentiating on both side ,

$$\frac{dx}{x^3} = \frac{dt}{2} \tag{4.1}$$

Now integrating,

$$\int dy = \int \frac{dt}{2t} \tag{4.2}$$

$$y = \frac{1}{2} \ln t + c \tag{4.3}$$

substituting t,

$$y = \frac{1}{2} \left(\ln \left(1 - \frac{1}{x^2} \right) \right) + c \tag{4.4}$$

finding constant by putting x=2 and y=0

$$c = \frac{1}{2} ln \frac{4}{3} \tag{4.5}$$

Solution

This leads to:

$$y = \frac{1}{2}\left(\ln\frac{4}{3}\left(1 - \frac{1}{x^2}\right)\right) \tag{5.1}$$

Finite difference Method

Initial point of curve $(1.0001, -4.10)$

$$h = 0.001 \tag{6.1}$$

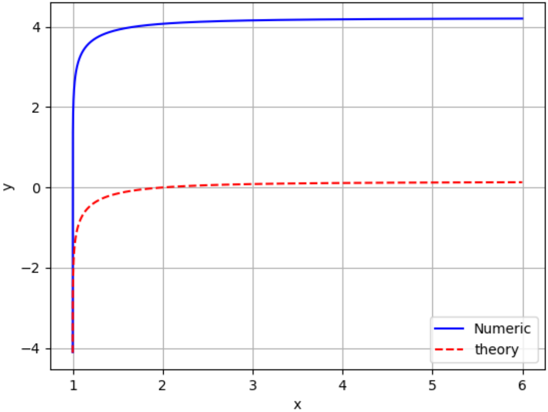
$$y_{n+1} = y_n + h \cdot \left(\frac{dy}{dx}\right) \tag{6.2}$$

$$x_{n+1} = x_n + h \tag{6.3}$$

Substituting $\frac{dy}{dx}$,

$$y_{n+1} = y_n + h \cdot \left(\frac{1}{x(x^2 - 1)}\right) \tag{6.4}$$

Plot by Finite Difference Method



Note:- Finite difference method fails here because $\frac{dy}{dx}$ is too large near x=1.Which creates a significant error in calculating y_{n+1}

C-Code For Finite difference method

```
#include <stdio.h>
#include <math.h>

float h = 0.001;
// Derivative function
float derivative(float y, float x) {
    return 1/(x*(x*x-1));
}
// Solution function
void solution(float *x, float *y, int n) {
    for (int i = 1; i <= n; i++) {
        // Update y using the derivative
        *y += derivative(*y, *x) * h;
        *x += h; // Increment x by h
    }
}
```

Runga-kutta Method

Let ,

$$h = 0.001 \tag{8.1}$$

$$\frac{dy}{dx_n} = f(x_n) \tag{8.2}$$

$$\tag{8.3}$$

Then ,

$$k_1 = hf(x_n) \tag{8.4}$$

$$k_2 = hf(x_n + h/2) \tag{8.5}$$

$$k_3 = hf(x_n + h/2) \tag{8.6}$$

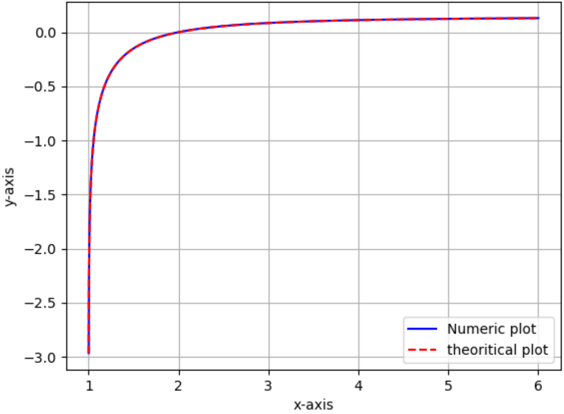
$$k_4 = hf(x_n + h) \tag{8.7}$$

$$k = \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{8.8}$$

$$y_{n+1} = y_n + k \tag{8.9}$$

$$x_{n+1} = x_n + h \tag{8.10}$$

Plot by Runga-Kutta Method



Note:- Runga-Kutta Method is computationally expensive because of its 4th order

C-Code For Runga-Kutta method

```
#include<stdio.h>
#include<math.h>
float h = 0.001; // global variable
float func(float x){
      return 1.00/(x*(x*(x)-1.00));
}
void solution(float *x,float *y,int n){
        float k1,k2,k3,k4,k;
        for(int i=1;i<=n;i++){
        k1 = h*func(*x);
        k2 = h*func(*x+h/2);
        k3 = h*func(*x+h/2);
        k4 = h*func(*x+h);
        k = (1.0/6.0)*(k1 + 2*k2 + 2*k3 + k4);
        *y += k; // finding the next value of y
        *x += h; // increment in x
        }
```