Question-12.9.7.1.1

EE24BTECH11030 - J.KEDARANANDA

Question

Solve the differential equation:

$$\frac{d^2y}{dx^2} + 5x\left(\frac{dy}{dx}\right)^2 - 6y = \ln(x), \tag{1}$$

Theoritical Solution :

The given differential equation is a second-order nonlinear ordinary differential equation and cannot be theoritcally solved using known methods.

Computational Solution :

Euler's Method **By the first principle of derivative,**

$$y'(x) = \lim_{h \to 0} \frac{y(x+h) - y(x)}{h} \tag{2}$$

$$y(x+h) = y(x) + h \cdot y'(x), \quad h \to 0 \tag{3}$$

For a m^{th} order differential equation:

Let:

$$y_1 = y, \quad y_2 = y', \quad y_3 = y'', \quad \ldots, \quad y_m = y^{m-1} \tag{4}$$

Euler's Method (System of Equations)

Then, we obtain the system:

$$\begin{bmatrix} y_1' \\ y_2' \\ \vdots \\ y_{m-1}' \\ y_m' \end{bmatrix} = \begin{bmatrix} y_2 \\ y_3 \\ \vdots \\ y_m \\ f(x, y_1, y_2, \ldots, y_m) \end{bmatrix} \tag{5}$$

Here, f is described by the given differential equation.

The initial conditions are:

$$y_1(x_0) = K_1, \quad y_2(x_0) = K_2, \quad \ldots, \quad y_m(x_0) = K_m$$

Representing the system in Euler's form (using the first principle of derivative):

$$\begin{bmatrix} y_1(x+h) \\ y_2(x+h) \\ \vdots \\ y_m(x+h) \end{bmatrix} = \begin{bmatrix} y_1(x) + hy_2(x) \\ y_2(x) + hy_3(x) \\ \vdots \\ y_m(x) + hf(x, y_1, y_2, \ldots, y_m) \end{bmatrix} \tag{6}$$

$$\begin{bmatrix} y_1(x+h) \\ \vdots \\ y_{m-1}(x+h) \\ y_m(x+h) \end{bmatrix} = \begin{bmatrix} y_1(x) \\ \vdots \\ y_{m-1}(x) \\ y_m(x) \end{bmatrix} + h \begin{bmatrix} y_2(x) \\ \vdots \\ y_m(x) \\ f(x, y_1, y_2, \ldots, y_m) \end{bmatrix} \tag{7}$$

Euler's Method (System Representation)

$$\vec{y}(x+h) = \vec{y}(x) + h \begin{bmatrix} 0 & 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 0 & 1 \\ 0 & 0 & 0 & 0 & \dots & 0 & \frac{f(x,y_1,y_2,\dots,y_m)}{y_m} \end{bmatrix} \vec{y}(x) \tag{8}$$

$$\vec{y}(x+h) = \begin{bmatrix} 1 & h & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & h & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & h & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 & h \\ 0 & 0 & 0 & 0 & \dots & 0 & 1 + \frac{f(x,y_1,y_2,\dots,y_m)}{y_m} \end{bmatrix} \vec{y}(x) \tag{9}$$

System Representation for a Specific Differential Equation (Part 1)

Note: The vector $\vec{y_n}$ is not to be confused with y_k, which represents the $(k-1)^{\text{th}}$ derivative of $y(x)$.

The given differential equation is:

$$y'' + 5x(y')^2 - 6y = \ln(x) \tag{10}$$

$$y'' = \ln(x) - 5x(y')^2 + 6y \tag{11}$$

We see that $m = 2$, so:

$$y_3 = y'' = \ln(x) - 5x(y')^2 + 6y \tag{12}$$

$$\begin{bmatrix} y_1' \\ y_2' \end{bmatrix} = \begin{bmatrix} y_2 \\ \ln(x) - 5x(y')^2 + 6y \end{bmatrix} \tag{13}$$

System Representation for a Specific Differential Equation (Part 2)

Iterative System Representation:

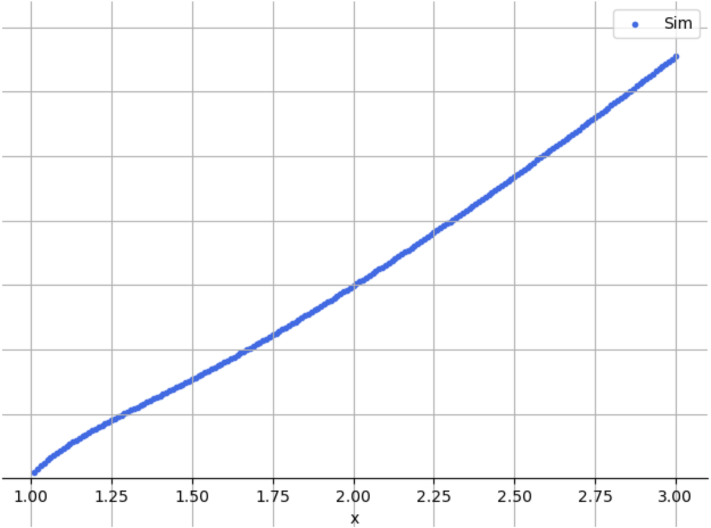
$$\begin{bmatrix} y_{1,n+1} \\ y_{2,n+1} \end{bmatrix} = \begin{bmatrix} y_{1,n} \\ y_{2,n} \end{bmatrix} + h \begin{bmatrix} y_{2,n} \\ \ln(x_n) - 5x_n(y_{2,n})^2 + 6y_{1,n} \end{bmatrix} \tag{14}$$

$$\vec{y}_{n+1} = \begin{bmatrix} 1 & h \\ 0 & 1 + \ln(x_n) - 5x_n(y_{2,n})^2 + 6y_{1,n} \end{bmatrix} \vec{y}_n \tag{15}$$

Initial Conditions:

$$x_0 = 0, \quad y_{1,0} = 0.01, \quad y_{2,0} = 1 \tag{16}$$

Diagram



C-Code

```c
#include <stdlib.h>
#include <math.h>

// get 'n' points on the plot of the differential equation, 'h' being the step size, 'x', 'y1', 'y2' being the inital conditions
// y1 = y, y2 = y'
float **diffEqPoints(int n, float h, float x, float y1, float y2){
  float pts = (float ) malloc(sizeof(float *) * n);

  // iteratively use euler's method with given parameters to return 'n' points in the plot of the differential equation
  for (int i = 0; i < n; i++){
    pts[i] = (float *) malloc(sizeof(float) * 2);
```

```c
            float y2_new = y2 + h*(-5 * x * pow(y2, 2)
               + 6 * y1 + log(x));
      y1 = y1 + h*y2;
      y2 = y2_new;
      x = x + h;
      pts[i][0] = x;
      pts[i][1] = y1;
   }
   return pts;
}
void freeMultiMem(float **points, int n){
   for(int i = 0; i < n; i++){
      free(points[i]);
   }
   free(points);
}
```

Python-Code

```python
import numpy as np
import matplotlib.pyplot as plt
import ctypes
import math

# dll linking
dll = ctypes.CDLL('./points.so')

# describing the argument and return types of the function 'diffEqPoints' and 'freeMultiMem' in the dll
dll.diffEqPoints.argtypes = [ctypes.c_int] + [ctypes.c_float]*4
dll.diffEqPoints.restype = ctypes.POINTER(ctypes.POINTER(ctypes.c_float))

dll.freeMultiMem.argtypes = [ctypes.POINTER(ctypes.POINTER(ctypes.c_float)), ctypes.c_int]
```

```
dll.freeMultiMem.restype = None

n = 200  # number of points to plot for given
    differential equation plot
h = 0.01 # step size

# initial conditions, y1 = y, y2 = dy/dx
x = 1.0     # x must be greater than 0 for log(x)
y1 = 0.01   # initial y1
y2 = 1.0    # initial dy/dx

# setting up the plot
plt.figure(figsize=(8, 6))

# getting an array of all the points in the plot
pts = dll.diffEqPoints(n, h, x, y1, y2)

# plotting the differential equation using plt.scatter
coords = []
```

```python
for i in range(n):
    pt = pts[i]  # access individual point from the
        pointer
    print(pt[0], pt[1])  # print coordinates to the
        console for verification
    coords.append(np.array([pt[0], pt[1]]))

coords_plot = np.array(coords)
plt.scatter(coords_plot[:, 0], coords_plot[:, 1],
    marker=".", label="Sim", color="royalblue")

# freeing the memory of the array 'pts'
dll.freeMultiMem(pts, n)

# customize the plot (axes, labels, grid)
ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['left'].set_position('zero')
```

Python-Code

```
ax.spines['right'].set_color('none')
ax.spines['bottom'].set_position('zero')

plt.legend(loc='best')
plt.grid()
plt.axis('equal')

plt.xlabel('x')
plt.ylabel('y')

# Display the plot
plt.show()
```