9.3.12B

EE24BTECH11027 - G.V.Satwika

Problem statement

Plot the solution of the differential equation:

$$y'' + xy' + xy = x. \tag{1}$$

Solution I

To plot the curve of the given differential equation (1) we can do it using the method of finite differences which is a numerical technique for solving complex differential equations by approximating derivatives with differences.
The approximated forward derivative of $y(x)$ is given as:

$$y_n' \approx \frac{y_{n+1} - y_n}{h} \tag{2}$$

On rearranging we get,

$$y_{n+1} = y_n + y_n'(h) \tag{3}$$

And also

$$x_{n+1} = x_n + h \tag{4}$$

Solution II

The approximated forward derivative of second order of $y(x)$ is given as:

$$y_n'' \approx \frac{y_{n+1}' - y_n'}{h} \tag{5}$$

Substitute eq (2) in eq (5) we get,

$$y_n'' \approx \frac{y_{n+2} - 2y_{n+1} - y_n}{h^2} \tag{6}$$

Substitute eq (2) and eq (6) in eq (1) and on reaaranging we get,

$$y_{n+2} = y_{n+1}\left(2 - hx_n\right) + y_n\left(1 + hx_n - h^2x_n\right) + h^2x_n \tag{7}$$

We need to assume two initial conditions as it is a second order differential equation.

Solution III

So here we assume the initial conditions as

$$x_0 = 0 \tag{8}$$

$$y_0 = 0 \tag{9}$$

$$y_0' = 1 \tag{10}$$

$$h = 0.1 \tag{11}$$

substitute eq (8), eq (9) and eq (10) in eq (1)
we get

$$y''(0) = 0 \tag{12}$$

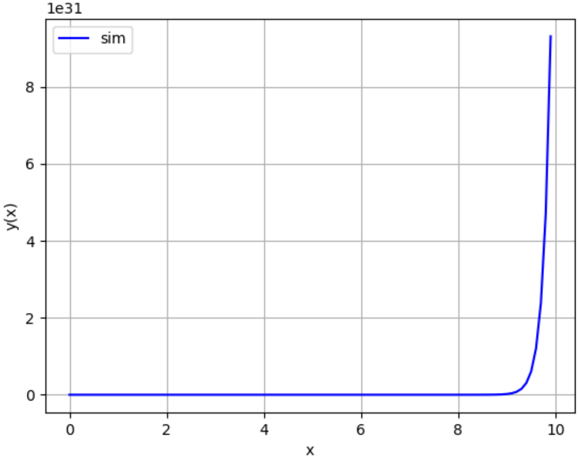
Substitute eq (10) in eq (3)

$$y_1 = y_0 + y_0'(0.1) \tag{13}$$

$$y_1 = 0.1 \tag{14}$$

For the rest of the points use eq (7) we get the other points.

Plot



```
#include <stdio.h>
#include <math.h>
// increasing the value of x by h where x_n means the new values of the
    x
float x_n(float x,float h){
  return x+h;
}
// we have already assumed the y_0 to find y_1 we are using finite
    differences method for first derivative
float y_1(float y_0,float dy,float h){
  return h*dy+y_0; // here dy means first derivative of y at x=0 i.e y
    '(0)
}
// to find the values of y from y2
float y_n(float y_1,float y_0,float h,float x_0){//here y_1 and y_0
    represents y_n-1 and y_n-2
  return y_1*(2-h*x_0) + y_0*(1+h*x_0-h*h*x_0) + h*h*x_0;//after
    substituting the y' and y" values in the given equation i got y_n+2=
    y_n-1(2-hx_n)+ y_n(1+hx_n-h^2x_n) + h^2x_n
```

C Code II

```
}
```

Python Code I

```python
import matplotlib.pyplot as plt
from ctypes import CDLL, c_float

# Load the shared library
newvalues = CDLL('./b.so')

# Declare the functions from the shared library
x_n = newvalues.x_n
y_1 = newvalues.y_1
y_n = newvalues.y_n

# Set return types for the C functions
x_n.restype = c_float
y_1.restype = c_float
y_n.restype = c_float

# Initial conditions
```

Python Code II

```python
x_0 = 0.0
y_0 = 0.0
dy_0 = 1.0  # Assumed first derivative
h = 0.1  # Increment in x

# Initialize variables for storing results
x_values = [x_0]
y_values = [y_0]
#computing x1
x1 = round(x_n(c_float(x_0), c_float(h)),3)
x_values.append(x1)
# Compute y1 using the first derivative
y1 = round(y_1(c_float(y_0), c_float(dy_0), c_float(h)),3)
y_values.append(y1)
# Compute subsequent y values using the finite difference method
for i in range(2, 100):  # Calculate up to y100
```

Python Code III

```
    x_i = round(x_n(c_float(x_values[-1]), c_float(h)),3)  # Update x
    y_i = round(y_n(c_float(y_values[-1]), c_float(y_values[-2]), c_float(h), c_float(x_values[-1])),3)
    x_values.append(x_i)
    y_values.append(y_i)
# Plot the results
plt.plot(x_values, y_values,color='blue', label='sim')
plt.xlabel("x")
plt.ylabel("y(x)")
plt.legend()
plt.grid(True)
plt.savefig("../figs/fig.png")
```