Bahria University,

Karachi Campus

A logo with text on it

Description automatically generated

LAB EXPERIMENT NO.

10

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| **01** | Write a Python program that utilize Euler’s method for IVP: 𝑦′ = 2 − 𝑒−4𝑡 − 2𝑦 with y (0) = 1, step size of h=0.1 to find approximate values of the solution at t = 0.1, 0.2, 0.3, 0.4, and 0.5 |
| **02** | Write a Python program that utilizes improved Euler’s method for IVPA black text on a white background  Description automatically generated  with y (0) = 0, to find approximate values of the solution at t = 1, t = 2, t = 3, t = 4, and t = 5. Use h = 0.1, h = 0.05, h = 0.01, h = 0.005 and h = 0.001 for the approximations. |
|  |  |

Submitted On:

10-12-2024

(Date: DD/MM/YYYY)

**Task No. 01**: Write a Python program that utilize Euler’s method for IVP: 𝑦′ = 2 − 𝑒−4𝑡 − 2𝑦 with y (0) = 1, step size of h=0.1 to find approximate values of the solution at t = 0.1, 0.2, 0.3, 0.4, and 0.5

**Solution:**

import numpy as np

def f(t, y):

return 2 - np.exp(-4 \* t) - 2 \* y

t0 = 0

y0 = 1

h = 0.1 #

steps = 5

t\_values = [t0]

y\_values = [y0]

for i in range(steps):

t\_next = t\_values[-1] + h

y\_next = y\_values[-1] + h \* f(t\_values[-1], y\_values[-1])

t\_values.append(t\_next)

y\_values.append(y\_next)

for t, y in zip(t\_values, y\_values):

print(f"t = {t:.1f}, y ≈ {y:.4f}")

**A number and equation on a white background

Description automatically generatedOutput:**

**Task No. 02**: . Write a Python program that utilize improved Euler’s method for IVP:

A black text on a white background

Description automatically generated

with y (0) = 0, to find approximate values of the solution at t = 1, t = 2, t = 3, t = 4, and t = 5. Use h = 0.1, h = 0.05, h = 0.01, h = 0.005 and h = 0.001 for the approximations.

**Solution:**

import numpy as np

def f(t, y):

return -0.5 \* np.exp(t / 2) \* np.sin(5 \* t) + 5 \* np.exp(t / 2) \* np.cos(5 \* t) + y

def improved\_euler(f, y0, t0, t\_end, h):

n = int((t\_end - t0) / h) # Number of steps

t\_values = np.linspace(t0, t\_end, n + 1) # Generate time points

y\_values = np.zeros(n + 1) # Array to store the solution

y\_values[0] = y0

for i in range(n):

t = t\_values[i]

y = y\_values[i]

k1 = f(t, y)

y\_predict = y + h \* k1

k2 = f(t + h, y\_predict)

y\_values[i + 1] = y + (h / 2) \* (k1 + k2)

return t\_values, y\_values

y0 = 0 # Initial condition y(0) = 0

t0 = 0 # Start time

t\_end = 5 # End time

step\_sizes = [0.1, 0.05, 0.01, 0.005, 0.001] # Different step sizes

# Solve the IVP for each step size and print all values

for h in step\_sizes:

print(f"\nStep size h = {h}:")

t\_values, y\_values = improved\_euler(f, y0, t0, t\_end, h)

for t, y in zip(t\_values, y\_values):

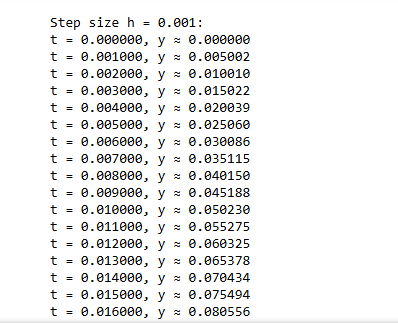
print(f"t = {t:.6f}, y ≈ {y:.6f}")

**Solution:**

**A white board with green lines

Description automatically generated**

**A screenshot of a computer screen

Description automatically generated**A screenshot of a computer

Description automatically generated**A screenshot of a computer

Description automatically generated**