

## 12.9.7.2.2 Presentation

G. Abhimanyu Koushik  
EE24BTECH11024

January 8, 2025

## 1 Problem

## 2 Solution

- Input Parameters
- Laplace Transform properties
- Equation solving
- Computational Solution

## 3 Plot of the function

## 4 C Code

## 5 Python Code

## Problem Statement

Solve the differential equation  $\frac{d^2y}{dx^2} - 2\frac{dy}{dx} + 2y = 0$  with initial conditions  $y(0) = 1$  and  $y'(0) = 0$

# Input Parameters

Variable	Description
$n$	Order of given differential equation
$a_i$	Coefficient of $i$ th derivative of the function in the equation
$c$	constant in the equation
$y^i$	$i$ th derivative of given function
$\mathbf{y}(t)$	$\begin{pmatrix} c \\ y(t) \\ y'(t) \\ \vdots \\ y^{n-1}(t) \end{pmatrix}$
$h$	stepsize, taken to be 0.001
$u(x)$	Unit step function

# Laplace Transform properties

## Properties of Laplace tranform

$$\mathcal{L}(y'') = s^2 \mathcal{L}(y) - sy(0) - y'(0) \quad (3.1)$$

$$\mathcal{L}(y') = s\mathcal{L}(y) - y(0) \quad (3.2)$$

$$\mathcal{L}(1) = \frac{1}{s} \quad (3.3)$$

$$\mathcal{L}^{-1}\left(\frac{s}{s^2 + 1}\right) = (\cos x) u(x) \quad (3.4)$$

$$\mathcal{L}^{-1}\left(\frac{1}{s^2 + 1}\right) = (\sin x) u(x) \quad (3.5)$$

$$\mathcal{L}(cf(t)) = c\mathcal{L}(f(t)) \quad (3.6)$$

$$\mathcal{L}(f(t)) = F(s) \implies \mathcal{L}(e^{at}f(t)) = F(s - a) \quad (3.7)$$

## Equation solving

Applying the properties to the given equation

$$y'' - 2y' + 2y = 0 \quad (3.8)$$

$$\mathcal{L}(y'') + \mathcal{L}(-2y') + \mathcal{L}(2y) = 0 \quad (3.9)$$

$$s^2 \mathcal{L}(y) - sy(0) - y'(0) - 2s\mathcal{L}(y) + 2y(0) + 2\mathcal{L}(y) = 0 \quad (3.10)$$

Substituting the initial conditions gives

$$s^2 \mathcal{L}(y) - s - 2s\mathcal{L}(y) + 2 + 2\mathcal{L}(y) = 0 \quad (3.11)$$

$$\mathcal{L}(y) = \frac{s-2}{s^2-2s+2} \quad (3.12)$$

$$\mathcal{L}(y) = \frac{s-2}{(s-1)^2+1} \quad (3.13)$$

$$\mathcal{L}(y) = \frac{s-1}{(s-1)^2+1} + \frac{-1}{(s-1)^2+1} \quad (3.14)$$

Let

$$F(s) = \mathcal{L}(y) \quad (3.15)$$

## Computational Solution

Consider the given linear differential equation

$$a_n y^n + a_{n-1} y^{n-1} + \cdots + a_1 y' + a_0 y + c = 0 \quad (3.23)$$

Then

$$y'(t) = \lim_{h \rightarrow 0} \frac{y(t+h) - y(t)}{h} \quad (3.24)$$

$$y(t+h) = y(t) + hy'(t) \quad (3.25)$$

Similarly

$$y^i(t+h) = y^i(t) + hy^{i+1}(t) \quad (3.26)$$

$$y^{n-1}(t+h) = y^{n-1}(t) + hy^n(t) \quad (3.27)$$

$$y^{n-1}(t+h) = y^{n-1}(t) + h \left( -\frac{a_{n-1}}{a_n} y^{n-1} - \frac{a_{n-2}}{a_n} y^{n-2} - \cdots - \frac{a_0}{a_n} y - \frac{c}{a_n} \right) \quad (3.28)$$

Where  $i$  ranges from 0 to  $n - 1$

$$\mathbf{y}(t+h) = \mathbf{y}(t) + \begin{pmatrix} 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 0 & 1 \\ -\frac{1}{a_n} & -\frac{a_0}{a_n} & -\frac{a_1}{a_n} & -\frac{a_2}{a_n} & \dots & -\frac{a_{n-2}}{a_n} & -\frac{a_{n-1}}{a_n} \end{pmatrix} (h\mathbf{y}(t)) \quad (3.29)$$

$$\mathbf{y}(t+h) = \begin{pmatrix} 1 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & h & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & h & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 & h \\ -\frac{h}{a_n} & -\frac{a_0 h}{a_n} & -\frac{a_1 h}{a_n} & -\frac{a_2 h}{a_n} & \dots & -\frac{a_{n-2} h}{a_n} & 1 - \frac{a_{n-1} h}{a_n} \end{pmatrix} (\mathbf{y}(t)) \quad (3.30)$$



Discretizing the steps gives us

$$\mathbf{y}_{k+1} = \begin{pmatrix} 1 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & h & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & h & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 & h \\ -\frac{h}{a_n} & -\frac{a_0 h}{a_n} & -\frac{a_1 h}{a_n} & -\frac{a_2 h}{a_n} & \dots & -\frac{a_{n-2} h}{a_n} & 1 - \frac{a_{n-1} h}{a_n} \end{pmatrix} (\mathbf{y}_k) \quad (3.31)$$

where  $k$  ranges from 0 to number of data points with  $y_0^i$  being the given

initial condition and vector  $\mathbf{y}_0 = \begin{pmatrix} c \\ y(0) \\ y'(0) \\ \vdots \\ y^{n-1}(0) \end{pmatrix}$

For the given question

$$\mathbf{y}_{k+1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & h \\ -h & 0 & 1 \end{pmatrix} \mathbf{y}_k \quad (3.32)$$

Record the  $y_k$  for

$$x_k = \textit{lowerbound} + kh \quad (3.33)$$

and then plot the graph. The result will be as given below.  
The codes below verifies the obtained solution.

## Plot of the function

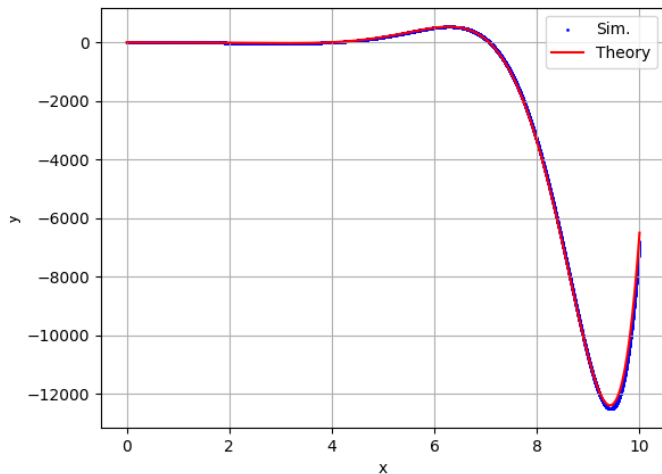


Figure: Function satisfying given differential equation

## C Code

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "functions.h"

double** matrixgen(int order, double coefficients[order+2], double
    stepsize){
    double** outputmatrix = identity(order+1);
    for(int i=1; i<order; i++){
        outputmatrix[i][i+1] = stepsize;
    }
    outputmatrix[order][0] = -1/coefficients[0]*stepsize;
    for(int i=1; i<order+1; i++){
        outputmatrix[order][i] = (-coefficients[order+1-i]/
            coefficients[0])*stepsize;
    }
    outputmatrix[order][order] += 1;
    return outputmatrix;}
```

```

double* recorddata(double lowerbound, double upperbound, int order,
    double coefficients[order+2], double initialconditions[order], double
    stepsize){
    double** vector_y = createMat(order+1,1);
    vector_y[0][0] = coefficients[order+1];
    for(int i=0;i<order;i++){
        vector_y[i+1][0] = initialconditions[i];
    }
    double** matrix = matrixgen(order, coefficients, stepsize);
    int no_datapoints = ((upperbound-lowerbound)/stepsize);
    double* yvalues = malloc(no_datapoints*sizeof(double));
    for(int i = 0; i<no_datapoints; i++){
        vector_y = Matmul(matrix,vector_y,order+1,order+1,1);
        yvalues[i] = vector_y[1][0];
    }
    return yvalues;
}

```

## Python Code for Plotting

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

# Load the shared library
solver = ctypes.CDLL('./solver.so')

# Define the function signatures
solver.recorddata.restype = ctypes.POINTER(ctypes.c_double)
solver.recorddata.argtypes = [
    ctypes.c_double, # lowerbound
    ctypes.c_double, # upperbound
    ctypes.c_int, # order
    ctypes.POINTER(ctypes.c_double), # coefficients
    ctypes.POINTER(ctypes.c_double), # initialconditions
    ctypes.c_double # stepsize
]
```

*# Define parameters*

order = 2

lowerbound = 0.0

upperbound = 10.0

stepsize = 0.001

coefficients = np.array([1.0, 0.0, 0.0, 1.0], dtype=np.double)

initialconditions = np.array([0.0, 0.0], dtype=np.double)

*# Calculate the number of data points*

no\_datapoints = **int**((upperbound - lowerbound) / stepsize)

*# Call the C function*

```
results_ptr = solver.recorddata(  
    ctypes.c_double(lowerbound),  
    ctypes.c_double(upperbound),  
    ctypes.c_int(order),  
    coefficients.ctypes.data_as(ctypes.POINTER(ctypes.c_double)),  
    initialconditions.ctypes.data_as(ctypes.POINTER(ctypes.c_double)),
```

```
    ctypes.c_double(stepsize)
)
# Convert results back to a NumPy array
results = np.ctypeslib.as_array(results_ptr, shape=(no_datapoints,))
# Generate x-values for plotting
x_values = np.arange(lowerbound + stepsize, upperbound + stepsize,
    stepsize)
# Calculate the y-values for the function  $y = -1/2 * x^2$ 
y_function = -0.5 * x_values**2
# Plot the data
plt.scatter(x_values, results, color='blue', s=1, label='Sim.')
plt.plot(x_values, y_function, color='red', label='Theory')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.grid(True)
plt.savefig('../figs/fig.png')
plt.show()
```