MA-221(Numerical Analysis)
Course Instructor: Prof. Rajendra K. Ray
TA: Kajal Mittal, Niladri Bose
Lab Assignment-12
Date: 29/04/2025

Instructions

- Solve each problem using Python, C++, and MATLAB.
- Plot every model.

Question 1: Spacecraft Descent Analysis (Richardson Extrapolation)

Background

When a spacecraft comes back to Earth, it is very difficult to measure how fast it is falling because the air is very turbulent. Scientists record the spacecraft's height (altitude) at different times. They want to find out the true falling speed using a smart method called Richardson Extrapolation.

Data Table

Time (s)	Altitude (m)
0.0	12000
0.2	11980
0.4	11959
0.6	11936
0.8	11910
1.0	11885
1.2	11857
1.4	11828
1.6	11800
1.8	11769
2.0	11740
2.2	11710
2.4	11679

Tasks

1. Estimate the speed at $t = 1.2$ s using forward difference with $h = 0.4$ s.
2. Estimate again using $h = 0.2$ s.
3. Apply Richardson extrapolation to improve the result.
4. Compare the extrapolated result with basic finite differences: how much is the error reduced?
5. List three possible real-world sources of error during measurement.
6. Predict the spacecraft altitude at $t = 1.4$ s assuming constant velocity. Compare with actual data.

Question 2: Chemical Reactor Monitoring (Forward Derivative)

Background:

In a chemical plant, engineers measure the temperature of a reactor every minute. They want to know how fast the temperature is rising at the beginning of the experiment.

Data Table

Time (min)	Temperature (°C)
0	25.0
1	27.2
2	30.1
3	33.8
4	38.2
5	43.4
6	49.3
7	56.1
8	63.7
9	72.1
10	81.4
11	91.5
12	102.5

Tasks

1. Estimate the rate of temperature rise at $t = 0$ using 2-point forward difference.
2. Re-estimate using 3-point forward difference (higher-order).
3. Which method is more reliable? Explain.
4. Based on the data, is the temperature rise linear, quadratic, or exponential?
5. What risks could arise if the temperature increases faster than expected?
6. If only the first four readings are available, suggest another method to estimate the rate.

Question 3. Delivery Truck Stoppage (Backward Derivative)

Background:

A delivery truck is being tracked using GPS every 5 minutes. At 50 minutes, the driver suddenly stops. The company wants to check whether the truck was slowing down before stopping.

Data Table

Time (min)	Distance (km)
0	0.0
5	5.4
10	10.6
15	15.7
20	20.7
25	25.4
30	29.8
35	33.9
40	37.5
45	40.6
50	43.0

Tasks

1. Estimate the truck's speed at $t = 50$ min using 2-point backward difference.
2. Estimate again using 3-point backward difference.
3. Was the truck slowing down? Use your calculations to answer.
4. Conceptually estimate the stopping distance if the deceleration continues steadily.
5. Which backward method is more accurate? Why?
6. Discuss how GPS signal noise might affect the derivative calculation.

Question 4. Atmospheric Pressure Sensing (Central Derivative)

Background:

Scientists send a weather balloon into the sky and measure air pressure at different heights. They want to know how fast pressure is dropping at a certain height.

Data Table

Altitude (m)	Pressure (hPa)
0	1013
100	1001
200	990
300	979
400	969
500	959
600	950
700	941
800	933
900	925
1000	917
1100	910
1200	902

Tasks

1. Estimate the rate of pressure drop at 600 meters using 2-point central difference.
2. Re-estimate using a 4-point central difference formula.
3. Based on your results, is the pressure drop steady, increasing, or decreasing?
4. Does the data show an exponential decrease of pressure with height?
5. Why is the central difference method generally more accurate than forward/backward methods?
6. What problems could occur if the pressure drop rate is wrongly estimated during balloon flights?

Pseudocode for Plotting

Python

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

x = np.array([...])  # Replace with actual values
y = np.array([...])
```

```
plt.plot(x, y, label='Data')
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title("Data_Visualization")
plt.legend()
plt.show()
```

C++

```
#include <iostream>
#include "matplotlibcpp.h"

namespace plt = matplotlibcpp;

int main() {
    std::vector<double> x = {...};  // Replace with actual values
    std::vector<double> y = {...};
    plt::plot(x, y);
    plt::xlabel("X-axis");
    plt::ylabel("Y-axis");
    plt::title("Data_Visualization");
    plt::show();
    return 0;
}
```

MATLAB

```
x = [...]; % Replace with actual values
y = [...];
plot(x, y);
xlabel('X-axis');
ylabel('Y-axis');
title('Data_Visualization');
grid on;
```

Appendix: Pseudocode and Complexity Analysis

- The loop structure or recursive calls to estimate **Time Complexity** in terms of n (number of data points).
- The size of data structures used (e.g., arrays, tables) to estimate **Memory Complexity**.

A typical pseudocode might look like:

```
For i = 0 to n-1:
    Compute term involving n-i operations
    Accumulate result

=> Time Complexity: O(n^2)
=> Memory Complexity: O(n^2) if 2D table is used
```

Submission

- Submit all source codes with inline comments.
- Include a brief report for each problem (PDF).
- Attach plots for visual comparison.
- Code submission must be modular and readable.