Ezenwere Kelvin

21CG029910

CSC 431 ASSIGNMENT

Question 1

A biologist is studying the growth rate of a bacterial culture at time intervals. The observed data points for the population size $P(t)$ (in million) at specific times t (in hours) are as follows:

t **(hours)**	$P(t)$ **(millions)**
1	1.5
3	3.2
6	5.7
9	8.4

The task is to estimate the population size at $t = 4$ hours using interpolation techniques.
a) Derive the Lagrange polynomial $P(t)$ and use it to estimate $P(4)$.
b) Construct the divided difference table, derive the Newton polynomial $P(t)$, and use it to estimate $P(4)$.
c) Compare the results obtained from both methods and discuss any differences.

Question 2

A chemical manufacturing company uses a cylindrical tank to mix a solution. Due to quality concerns, it is critical to ensure that the volume of the solution is at a specific level. The volume V (in liters) as a function of the height h (in meters) of the liquid in the tank is given by:

$$V(h) = \pi r^2 h$$

Where r is the radius of the tank (assumed to be 2 meters). The target volume is 40 liters, but due to sensor calibration, an error term e needs to be considered, resulting in the equation:

$$f(h) = \pi r^2 h - 40 = 0$$

The following are the measured heights of the liquid for different time periods:

Time (min)	Measured Height h (m)
5	1.5
10	1.8
15	2.0
20	2.2
25	2.5

Using the initial guess $h_0 = 1.9$ and $h\backslash = 2.1$, from the dataset, perform three iterations of the Newton Raphson, Modified Secant and Fixed-Point Iteration methods. Show all calculations clearly, and round the results to four decimal places. Write a Python code to implement any of the methods.

Question 3

A civil engineering team is tasked with designing a drainage system to handle water flow during heavy rainfall. The flow rate Q (in cubic meters per second) through a rectangular channel is modeled by the Manning's equation:

$$f(x) = \frac{1}{n}AR^{\frac{2}{3}}S^{\frac{1}{2}} - Q = 0$$

Where:

$A = b \times h$ is cross-sectional area of the channel ($b = 2$ meters, width of the channel; h, height of water in meters).
$R = \frac{A}{P}$ is hydraulic radius ($P = b + 2h$, wetted perimeter).
$S = 0.01$ is channel slope.
$Q = 1.5$ is the required flow rate.
$n = 0.015$ is Manning's roughness coefficient.

The task is to determine the height h, of water that satisfies the flow condition. Given the interval $[0.2, 0.5]$, perform three iterations and show all calculations using Bisection, Secant and Regula Falsi methods. Write a Python function to implement any of the methods.

SOLUTION

Question 1

a) Lagrange Polynomial Estimate

The Lagrange polynomial passes through all given data points. Using the points $(1, 1.5), (3, 3.2), (6, 5.7), (9, 8.4)$, the polynomial is:

$$P(t) = 1.5 \cdot L_1(t) + 3.2 \cdot L_2(t) + 5.7 \cdot L_3(t) + 8.4 \cdot L_4(t)$$

The Lagrange basis polynomials are:

$$L_1(t) = \frac{(t-3)(t-6)(t-9)}{(1-3)(1-6)(1-9)} = \frac{(t-3)(t-6)(t-9)}{(-2)(-5)(-8)} = \frac{(t-3)(t-6)(t-9)}{-80}$$

$$L_2(t) = \frac{(t-1)(t-6)(t-9)}{(3-1)(3-6)(3-9)} = \frac{(t-1)(t-6)(t-9)}{(2)(-3)(-6)} = \frac{(t-1)(t-6)(t-9)}{36}$$

$$L_3(t) = \frac{(t-1)(t-3)(t-9)}{(6-1)(6-3)(6-9)} = \frac{(t-1)(t-3)(t-9)}{(5)(3)(-3)} = \frac{(t-1)(t-3)(t-9)}{-45}$$

$$L_4(t) = \frac{(t-1)(t-3)(t-6)}{(9-1)(9-3)(9-6)} = \frac{(t-1)(t-3)(t-6)}{(8)(6)(3)} = \frac{(t-1)(t-3)(t-6)}{144}$$

Now, evaluate each basis polynomial at $t = 4$:

$$L_1(4) = \frac{(4-3)(4-6)(4-9)}{-80} = \frac{(1)(-2)(-5)}{-80} = \frac{10}{-80} = -0.125$$

$$L_2(4) = \frac{(4-1)(4-6)(4-9)}{36} = \frac{(3)(-2)(-5)}{36} = \frac{30}{36} = 0.8333$$

$$L_3(4) = \frac{(4-1)(4-3)(4-9)}{-45} = \frac{(3)(1)(-5)}{-45} = \frac{-15}{-45} = 0.3333$$

$$L_4(4) = \frac{(4-1)(4-3)(4-6)}{144} = \frac{(3)(1)(-2)}{144} = \frac{-6}{144} = -0.0417$$

Now, multiply each basis polynomial by its corresponding $P(t)$ value:

$$1.5 \cdot L_1(4) = 1.5 \cdot (-0.125) = -0.1875$$

$$3.2 \cdot L_2(4) = 3.2 \cdot 0.8333 = 2.6667$$

$$5.7 \cdot L_3(4) = 5.7 \cdot 0.3333 = 1.9$$

$$8.4 \cdot L_4(4) = 8.4 \cdot (-0.0417) = -0.35$$

Summing these contributions:

$$P(4) \approx -0.1875 + 2.6667 + 1.9 - 0.35 = 4.0292 \text{ million}$$

b) Newton Polynomial Estimate

The divided difference table is constructed as follows:

t	$P(t)$	1^{st}	2^{nd}	3^{rd}
1	1.5	0.85	-0.00334	0.001807
3	3.2	0.8333	0.01111	
6	5.7	0.9		
9	8.4			

The Newton polynomial is:

$$P(t) = 1.5 + 0.85(t-1) - 0.00334(t-1)(t-3) + 0.001807(t-1)(t-3)(t-6)$$

Evaluating at $t = 4$:

$$P(4) = 1.5 + 0.85(4-1) - 0.00334(4-1)(4-3) + 0.001807(4-1)(4-3)(4-6)$$

$$P(4) = 1.5 + 0.85(3) - 0.00334(3)(1) + 0.001807(3)(1)(-2)$$

$$P(4) = 1.5 + 2.55 - 0.01002 - 0.01084 = 4.0291 \text{ million}$$

c) Comparison

Both methods yield $P(4) \approx 4.029$ million. The minor difference arises from rounding errors during manual calculations.

Question 2

Equation:

$$f(h) = 4\pi h - 40 = 0$$

The root is $h = \frac{10}{\pi} \approx 3.1831.$

1. Newton-Raphson Method

Formula:

$$h_{n+1} = h_n - \frac{f(h_n)}{f'(h_n)}$$

Here, $f(h) = 4\pi h - 40 and f'(h) = 4\pi.$

Initial guess: $h_0 = 1.9.$

Iteration 1:

$$h_1 = 1.9 - \frac{4\pi(1.9) - 40}{4\pi} = 1.9 - \frac{23.876 - 40}{12.566} = 1.9 + \frac{16.124}{12.566} = 3.1831$$

The method converges immediately to $h = 3.1831.$

2. Modified Secant Method

Formula:

$$h_{n+1} = h_n - \frac{f(h_n) \cdot \delta}{f(h_n + \delta) - f(h_n)}$$

Here, $\delta = 0.2$.

Initial guess: $h_0 = 1.9$.

Iteration 1:

$$f(h_0) = 4\pi(1.9) - 40 = 23.876 - 40 = -16.124$$

$$f(h_0 + \delta) = 4\pi(2.1) - 40 = 26.389 - 40 = -13.611$$

$$h_1 = 1.9 - \frac{(-16.124)(0.2)}{-13.611 - (-16.124)} = 1.9 - \frac{-3.2248}{2.513} = 1.9 + 1.283 = 3.1831$$

The method converges immediately to $h = 3.1831$.

3. Fixed-Point Iteration

Rearranged equation:

$$h = \frac{h + 10/\pi}{2}$$

Initial guess: $h_0 = 1.9$.

Iteration 1:

$$h_1 = \frac{1.9 + 10/\pi}{2} = \frac{1.9 + 3.1831}{2} = 2.5416$$

Iteration 2:

$$h_2 = \frac{2.5416 + 3.1831}{2} = 2.8624$$

Iteration 3:

$$h_3 = \frac{2.8624 + 3.1831}{2} = 3.0228$$

Python Code for Fixed-Point Iteration:

```python
import math

def fixed_point(h0, iterations):
    h = h0
    for i in range(iterations):
        h = (h + 10/math.pi) / 2
        print(f"Iteration {i+1}: h = {h:.4f}")
    return h
```

TEST 2

Question 1: Methods of Curve Fitting

A software engineering team wants to understand the relationship between the size of a codebase (x, in thousand lines of code) and the corresponding number of reported bugs (y). Due to issues from using peer programming approach of the software development process, a project timeline constraint term A needs to be considered, resulting in the model: $y = Ax^k$. The data collected from recent projects is as follows:

Codebase Size (x)	Reported Bugs (y)
1.0	5
2.0	7
3.0	10
4.0	12
5.0	15

a) From the power-law relationship given,
i. Derive the equation of the line of best fit using linearization method
ii. Predict the number of bugs for a codebase size of 6000 lines given that A = 4.8

Using the given data,

b) Derive the equation of the line of best fit using the
i. Least squares method
ii. Predict the number of bugs for a codebase of 6000 lines.

c) Derive the equation of the line of best fit using the
i. Grouped averages method
ii. Predict the number of bugs for a codebase size of 6000 lines

Question 2: Methods of Linear Systems of Equation

In a multi-core processor, a bottleneck core will cause delays when it consumes system resources in double units than another core. Suppose that resource utilization across the cores X, Y, Z in an IoT device is modelled by the following system of equations:
$x + y - z = 20$ .... ①
$2x + 3y + z = 44$ ...②
$3x + y + 2z = 35$ ...③
Where:

- equation 1 tracks how computational workloads (CPU cycles) are shared among the cores
- equation 2 monitors how memory (RAM) is used by the cores
- equation 3observes power consumption by the servers associated with core

a) Solve the above system of equations using:
i. Gaussian Elimination
ii. Gauss-Jordan Elimination
iii. LU Decomposition

b) Based on results from Gaussian Elimination
i. Which of the cores will cause system delays?

ii. Why?
iii. How can this bottleneck be resolved?

SOLUTION

Question 1: Methods of Curve Fitting

a) Power-Law Model: $y = Ax^k$

i. Linearization Method

1. **Transform the model:** Take natural logs:

$$\ln y = \ln A + k \ln x$$

2. **Compute sums for linear regression:**

$$\sum \ln x = 4.7874, \quad \sum \ln y = 11.0508,$$
$$\sum (\ln x \ln y) = 11.6825, \quad \sum (\ln x)^2 = 6.1994$$

3. **Calculate k and $\ln A$:**

$$k = \frac{5 \cdot 11.6825 - 4.7874 \cdot 11.0508}{5 \cdot 6.1994 - (4.7874)^2} \approx 0.68$$

$$\ln A = \frac{11.0508 - 0.68 \cdot 4.7874}{5} \approx 1.5614 \implies A \approx 4.76$$

Equation:

$$y \approx 4.76x^{0.68}$$

ii. Prediction with $A = 4.8$

1. **Recalculate** k using fixed $A = 4.8$:
$k \approx 0.673$
2. **Predict for** $x = 6$ ($6,000$ lines):

$$y = 4.8 \cdot 6^{0.673} \approx 16$$

Predicted bugs: $\boxed{16}$

b) Least Squares Method (Linear Model: $y = a + bx$)

i. Derive Equation

1. **Compute sums:**

$$\sum x = 15, \quad \sum y = 49, \quad \sum x^2 = 55, \quad \sum xy = 172$$

2. **Calculate** b **and** a:

$$b = \frac{5 \cdot 172 - 15 \cdot 49}{5 \cdot 55 - 15^2} = 2.5, \quad a = \frac{49 - 2.5 \cdot 15}{5} = 2.3$$

Equation:

$$y = 2.3 + 2.5x$$

ii. Prediction for $x = 6$

$$y = 2.3 + 2.5 \cdot 6 = 17.3 \approx 17$$

Predicted bugs: $\boxed{17}$

c) Grouped Averages Method

i. Derive Equation

1. **Group data:**
   - Group 1 $(x = 1, 2) : \bar{x} = 1.5, \bar{y} = 6$
   - Group 2 $(x = 4, 5) : \bar{x} = 4.5, \bar{y} = 13.5$
2. **Calculate slope b:**

$$b = \frac{13.5 - 6}{4.5 - 1.5} = 2.5$$

3. **Calculate intercept a:**

$$a = 6 - 2.5 \cdot 1.5 = 2.25$$

   Equation:

$$y = 2.25 + 2.5x$$

ii. Prediction for $x = 6$

$$y = 2.25 + 2.5 \cdot 6 = 17.25 \approx 17$$

Predicted bugs: $\boxed{17}$

Question 2: Methods of Linear Systems of Equation

a) Solving the System

i. Gaussian Elimination

1. **Augmented Matrix:**

$$\begin{bmatrix} 1 & 1 & -1 & | & 20 \\ 2 & 3 & 1 & | & 44 \\ 3 & 1 & 2 & | & 35 \end{bmatrix}$$

2. **Row Operations:**
   - $R2 \leftarrow R2 - 2R1$:

$$\begin{bmatrix} 0 & 1 & 3 & | & 4 \end{bmatrix}$$

   - $R3 \leftarrow R3 - 3R1$:

$$\begin{bmatrix} 0 & -2 & 5 & | & -25 \end{bmatrix}$$

   - $R3 \leftarrow R3 + 2R2$:

$$\begin{bmatrix} 0 & 0 & 11 & | & -17 \end{bmatrix}$$

3. **Upper Triangular Matrix:**

$$\begin{bmatrix} 1 & 1 & -1 & | & 20 \\ 0 & 1 & 3 & | & 4 \\ 0 & 0 & 11 & | & -17 \end{bmatrix}$$

4. **Back Substitution:**
   - $z = -\frac{17}{11}$
   - $y = \frac{95}{11}$

- $x = \frac{108}{11}$

Solution:

$$x = \frac{108}{11}, \quad y = \frac{95}{11}, \quad z = -\frac{17}{11}$$

ii. Gauss-Jordan Elimination

1. **Start with Augmented Matrix:**

$$\begin{bmatrix} 1 & 1 & -1 & | & 20 \\ 2 & 3 & 1 & | & 44 \\ 3 & 1 & 2 & | & 35 \end{bmatrix}$$

2. **Row Operations:**
   - $R2 \leftarrow R2 - 2R1$:

$$\begin{bmatrix} 0 & 1 & 3 & | & 4 \end{bmatrix}$$

   - $R3 \leftarrow R3 - 3R1$:

$$\begin{bmatrix} 0 & -2 & 5 & | & -25 \end{bmatrix}$$

   - $R3 \leftarrow R3 + 2R2$:

$$\begin{bmatrix} 0 & 0 & 11 & | & -17 \end{bmatrix}$$

3. **Normalize $R3$:**

$$\begin{bmatrix} 0 & 0 & 1 & | & -\frac{17}{11} \end{bmatrix}$$

4. **Eliminate z from $R1$ and $R2$:**
   - $R1 \leftarrow R1 + R3$:

$$\begin{bmatrix} 1 & 1 & 0 & | & \frac{203}{11} \end{bmatrix}$$

   - $R2 \leftarrow R2 - 3R3$:

$$\begin{bmatrix} 0 & 1 & 0 & | & \frac{95}{11} \end{bmatrix}$$

5. **Eliminate y from $R1$:**
   - $R1 \leftarrow R1 - R2$:

$$\begin{bmatrix} 1 & 0 & 0 & | & \frac{108}{11} \end{bmatrix}$$

6. **Solution:**

$$x = \frac{108}{11}, \quad y = \frac{95}{11}, \quad z = -\frac{17}{11}$$

iii. LU Decomposition

1. **Decompose Matrix A:**

$$A = \begin{bmatrix} 1 & 1 & -1 \\ 2 & 3 & 1 \\ 3 & 1 & 2 \end{bmatrix}$$

$$L = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & -2 & 1 \end{bmatrix}$$

$$U = \begin{bmatrix} 1 & 1 & -1 \\ 0 & 1 & 3 \\ 0 & 0 & 11 \end{bmatrix}$$

2. **Solve** $Ly = b$:

$$\begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & -2 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 20 \\ 44 \\ 35 \end{bmatrix}$$

- $y_1 = 20$
- $y_2 = 44 - 2 \cdot 20 = 4$
- $y_3 = 35 - 3 \cdot 20 + 2 \cdot 4 = -17$

3. **Solve** $Ux = y$:

$$\begin{bmatrix} 1 & 1 & -1 \\ 0 & 1 & 3 \\ 0 & 0 & 11 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 20 \\ 4 \\ -17 \end{bmatrix}$$

- $z = -\frac{17}{11}$
- $y = \frac{95}{11}$
- $x = \frac{108}{11}$

Solution:

$$x = \frac{108}{11}, \quad y = \frac{95}{11}, \quad z = -\frac{17}{11}$$

b) Bottleneck Analysis

i. Bottleneck Core: $\boxed{\text{Core } X}$
ii. Reason: Core X has the highest resource utilization ($x \approx 9.82$).
iii. Resolution: Redistribute workloads or enhance Core X's capacity.

ASSIGNMENT

In a 100 meters race, the speed $v(t)$ of the runners at time t is approximated by $v(t) = a_1t^2 + a_2t + a_3, 0 <= t <= 100$ where a_1, a_2 and a_3 are constants. It has been found that the speed at times $t = 3, t = 6$ and $t = 9$ seconds are $64, 133$ and 208 miles per second respectively, Find the speed at time $t = 15$ seconds using
a) Gaussian Elimination Method
b) Gauss-Jordan Elimination Method
c) LU Decomposition
d) Write a C++ function for any of the 3 methods

SOLUTION

At $t = 3$,
equation ① : $9a_1 + 3a_2 + a_3 = 64$
At $t = 6$,
equation ② : $36a_1 + 6a_2 + a_3 = 133$
At $t = 9$,
equation ③ : $81a_1 + 9a_2 + a_3 = 208$
Augmented Matrix:

$$\begin{bmatrix} 9 & 3 & 1 & | & 64 \\ 36 & 6 & 1 & | & 133 \\ 81 & 9 & 1 & | & 208 \end{bmatrix}$$

a) **Gaussian Elimination Method**

- $R2 : R2 - 4R1$
  Augmented Matrix:

$$\begin{bmatrix} 9 & 3 & 1 & | & 64 \\ 0 & -6 & -3 & | & -123 \\ 81 & 9 & 1 & | & 208 \end{bmatrix}$$

- $R3 : R3 - 9R1$
  Augmented Matrix:

$$\begin{bmatrix} 9 & 3 & 1 & | & 64 \\ 0 & -6 & -3 & | & -123 \\ 0 & -18 & -8 & | & -368 \end{bmatrix}$$

- $R3 : R3 - 3R2$
  Augmented Matrix:

$$\begin{bmatrix} 9 & 3 & 1 & | & 64 \\ 0 & -6 & -3 & | & -123 \\ 0 & 0 & 1 & | & 1 \end{bmatrix}$$

- **Back Substitution**
  - $a_3 = 1$
  - $-6a_2 - 3(1) = -123 \implies a_2 = 20$
  - $9a_1 + 3(20) + 1 = 64 \implies a_1 = \frac{1}{3}$

$\therefore a_1 = \frac{1}{3}, a_2 = 20, a_3 = 1$

b) **Gauss-Jordan Elimination Method**

- **Upper Triangular Matrix from Gaussian Elimination:**

$$\begin{bmatrix} 9 & 3 & 1 & | & 64 \\ 0 & -6 & -3 & | & -123 \\ 0 & 0 & 1 & | & 1 \end{bmatrix}$$

- **Normalize Leading Entries:**
  - $R1 : \frac{1}{9}R1, R2 : -\frac{1}{6}R2$

$$\begin{bmatrix} 1 & \frac{1}{3} & \frac{1}{9} & | & \frac{64}{9} \\ 0 & 1 & 0.5 & | & 20.5 \\ 0 & 0 & 1 & | & 1 \end{bmatrix}$$

- **Eliminate Above Entries:**
- $R1 : R1 - \frac{1}{3}R2$

$$\begin{bmatrix} 1 & 0 & -\frac{1}{18} & | & \frac{5}{18} \\ 0 & 1 & 0.5 & | & 20.5 \\ 0 & 0 & 1 & | & 1 \end{bmatrix}$$

- $R1 : R1 + \frac{1}{18}R3$, $R2 : R2 - 0.5R3$

$$\begin{bmatrix} 1 & 0 & 0 & | & \frac{1}{3} \\ 0 & 1 & 0 & | & 20 \\ 0 & 0 & 1 & | & 1 \end{bmatrix}$$

$\therefore a_1 = \frac{1}{3}, a_2 = 20, a_3 = 1$

c) **LU Decomposition**

- **Matrix Decomposition:**
  - $L = \begin{bmatrix} 1 & 0 & 0 \\ 4 & 1 & 0 \\ 9 & 3 & 1 \end{bmatrix}$
  - $U = \begin{bmatrix} 9 & 3 & 1 \\ 0 & -6 & -3 \\ 0 & 0 & 1 \end{bmatrix}$
- **Forward Substitution** $(Ly = b)$**:**
  - $y_1 = 64$
  - $y_2 = 133 - 4(64) = -123$
  - $y_3 = 208 - 9(64) - 3(-123) = 1$
- **Backward Substitution** $(Ux = y)$**:**
  - $a_3 = 1$
  - $a_2 = \frac{-123+3(1)}{-6} = 20$
  - $a_1 = \frac{64-3(20)-1}{9} = \frac{1}{3}$

$\therefore a_1 = \frac{1}{3}, a_2 = 20, a_3 = 1$

d) C++ Code for Gaussian Elimination

```
#include <iostream>
#include <vector>

using namespace std;

vector<double> gaussianElimination(vector<vector<double>> A, vector<double> b) {
    int n = A.size();
    vector<vector<double>> augmented(n, vector<double>(n + 1));

    // Form augmented matrix
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
```

```cpp
            augmented[i][j] = A[i][j];
        }
        augmented[i][n] = b[i];
    }

    // Forward elimination
    for (int col = 0; col < n; ++col) {
        for (int row = col + 1; row < n; ++row) {
            double factor = augmented[row][col] / augmented[col][col];
            for (int j = col; j <= n; ++j) {
                augmented[row][j] -= factor * augmented[col][j];
            }
        }
    }

    // Back substitution
    vector<double> x(n);
    for (int i = n - 1; i >= 0; --i) {
        x[i] = augmented[i][n];
        for (int j = i + 1; j < n; ++j) {
            x[i] -= augmented[i][j] * x[j];
        }
        x[i] /= augmented[i][i];
    }

    return x;
}

int main() {
    vector<vector<double>> A = {{9, 3, 1}, {36, 6, 1}, {81, 9, 1}};
    vector<double> b = {64, 133, 208};

    vector<double> coeffs = gaussianElimination(A, b);

    double a1 = coeffs[0];
    double a2 = coeffs[1];
    double a3 = coeffs[2];

    double t = 15;
    double speed = a1 * t * t + a2 * t + a3;

    cout << "The speed at t=15 is: " << speed << endl;

    return 0;
}
```

The speed at time $t = 15$ seconds is $\boxed{376}$ m/s.