Matgeo Presentation - Problem 5.9.10

EE25BTECH11048 - Revanth Siva Kumar.D

October 3, 2025

Problem Statement

The present age of a father is three years more than three times the age of his son. Three years hence the father's age will be 10 years more than twice the age of the son. Determine their present ages.

Data

Name	**Equation**
Equation 1	$y = 3x + 3 \Rightarrow -3x + y = 3$
Equation 2	$y + 3 = 2(x + 3) + 10 \Rightarrow -2x + y = 13$

Table : Equations

Solution

Let

$$x = \text{son's present age}, \quad y = \text{father's present age}.$$

From the problem statement:

1. **Father's present age is three years more than three times son's age**:

$$y = 3x + 3 \quad \Rightarrow \quad -3x + y = 3$$

2. **Three years hence, father's age will be 10 years more than twice son's age**:

$$y + 3 = 2(x + 3) + 10 \quad \Rightarrow \quad -2x + y = 13$$

Writing in matrix form:

$$\begin{pmatrix} -3 & 1 \\ -2 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 3 \\ 13 \end{pmatrix} \tag{0.1}$$

Solution

Forming the augmented matrix:

$$\left(\begin{array}{cc|c} -3 & 1 & 3 \\ -2 & 1 & 13 \end{array}\right) \tag{0.2}$$

Using Gaussian elimination:

$$\left(\begin{array}{cc|c} -3 & 1 & 3 \\ -2 & 1 & 13 \end{array}\right) \xleftrightarrow{R_2 \to R_2 - \frac{2}{3} R_1} \left(\begin{array}{cc|c} -3 & 1 & 3 \\ 0 & \frac{1}{3} & 11 \end{array}\right) \tag{0.3}$$

Back substitution gives:

$$\frac{y}{3} = 11 \tag{0.4}$$

$$y = 33 \tag{0.5}$$

$$-3x + y = 3 \;\Rightarrow\; -3x + 33 = 3 \;\Rightarrow\; x = 10 \tag{0.6}$$

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 10 \\ 33 \end{pmatrix} \tag{0.7}$$

Conclusion

Therefore, the present ages are:

$$\begin{aligned} \text{Son's age} &= 10 \text{ years} \\ \text{Father's age} &= 33 \text{ years} \end{aligned}$$

C code

```
#include <stdio.h>

double *gaussian_elimination(double a[2][3]) {

  static double sol[2];
  int i, j, k;
  double ratio;
  int n = 2;

  // forward elimination (j for rows and i for columns of augmented_matrix,)
  for (i = 0; i < n - 1; i++) {

    for (j = i + 1; j < n; j++) {
      ratio = a[j][i] / a[i][i];

      for (k = i; k < n + 1; k++) { // k is also for columns
        a[j][k] -=
            ratio *
            a[i][k]; // to do R2 -> R2 - ratio*R1 , then R3 -> R3 -ratio*R1
      }
    }
  }
```

C Code

```c
 // Back substitution
  for (i = n - 1; i >= 0; i--) {

    sol[i] = a[i][n];

    for (j = i + 1; j < n; j++) {
      sol[i] -= a[i][j] * sol[j];
    }

    sol[i] /= a[i][i];
  }

  return sol;
}

int main() {
    // Augmented matrix for the father-son problem: -3x + y = 3, -2x + y = 13
    double a[2][3] = {
        {-3, 1, 3},
        {-2, 1, 13}
    };

    double *solution = gaussian_elimination(a);

    printf("Son's age (x) = %.2lf\n", solution[0]);
    printf("Father's age (y) = %.2lf\n", solution[1]);

    return 0;
}
```

Python shared output

```python
import ctypes
import sys
import os
import numpy as np
import matplotlib.pyplot as plt

# save figure in figs folder
figs_folder = os.path.join("..", "figs")
os.makedirs(figs_folder, exist_ok=True) # create folder if it doesn't exist

# load the shared object
lib = ctypes.CDLL("./points.so")

# define return type and arg type
lib.gaussian_elimination.restype = ctypes.POINTER(ctypes.c_double)
lib.gaussian_elimination.argtypes = [((ctypes.c_double * 3) * 2)]

# define augmented matrix for father-son problem
# -3x + y = 3
# -2x + y = 13
a = ((ctypes.c_double * 3) * 2)()
a[0][:] = [-3.0, 1.0, 3.0]
a[1][:] = [-2.0, 1.0, 13.0]

# call the C function
sol = lib.gaussian_elimination(a)
solution = [sol[i] for i in range(2)]
x, y = solution
print("Solution from C:", solution)
```

Python shared output

```python
# create x range for plotting
x_vals = np.linspace(0, 15, 100)

# equations solved for y
y1 = 3 * x_vals + 3 # from y = 3x + 3
y2 = 2 * x_vals + 13 # from y + 3 = 2(x+3)+10  y = 2x + 13

# plotting
plt.figure(figsize=(8, 6))

# plot the lines
plt.plot(x_vals, y1, label="y = 3x + 3", color="green")
plt.plot(x_vals, y2, label="y = 2x + 13", color="blue")

# plot the solution point
plt.scatter(x, y, color="red", label=f"P({x:.2f},{y:.2f})")
plt.text(x + 0.3, y + 0.3, f"P({x:.2f},{y:.2f})", fontsize=10, color="red")

# labels and grid
plt.xlabel("X axis (Son's Age)")
plt.ylabel("Y axis (Father's Age)")
plt.title("Father-Son Age Problem: Intersection Point")
plt.legend()
plt.grid(True)

# save figure
plt.tight_layout()
plt.savefig(os.path.join(figs_folder, "solution.png"))
plt.show()
```

PYTHON plot.py

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

# save figure in figs folder
figs_folder = os.path.join("..", "figs")
os.makedirs(figs_folder, exist_ok=True) # create folder if it doesn't exist

# solve system of equations directly using numpy
# -3x + y = 3
# -2x + y = 13
A = np.array([[-3, 1],
             [-2, 1]], dtype=float)

b = np.array([3, 13], dtype=float)

solution = np.linalg.solve(A, b)
x, y = solution
print("Solution from Python:", solution)

# create x range for plotting
x_vals = np.linspace(0, 15, 100)

# equations solved for y
y1 = 3 * x_vals + 3 # from y = 3x + 3
y2 = 2 * x_vals + 13 - 2*x_vals? let's compute properly
# original equation: -2x + y = 13  y = 2x + 13

y2 = 2 * x_vals + 13 # from -2x + y = 13  y = 2x + 13
```

PYTHON plot.py

```python
# plotting
plt.figure(figsize=(8, 6))

# plot the lines
plt.plot(x_vals, y1, label="y = 3x + 3", color="green")
plt.plot(x_vals, y2, label="y = 2x + 13", color="blue")

# plot the solution point
plt.scatter(x, y, color="red", label=f"P({x:.2f},{y:.2f})")
plt.text(x + 0.3, y + 0.3, f"P({x:.2f},{y:.2f})", fontsize=10, color="red")

# labels and grid
plt.xlabel("X axis (Son's Age)")
plt.ylabel("Y axis (Father's Age)")
plt.title("Father-Son Age Problem: Intersection Point")
plt.legend()
plt.grid(True)

# save figure
plt.tight_layout()
plt.savefig(os.path.join(figs_folder, "solution.png"))
plt.show()
```

Plot

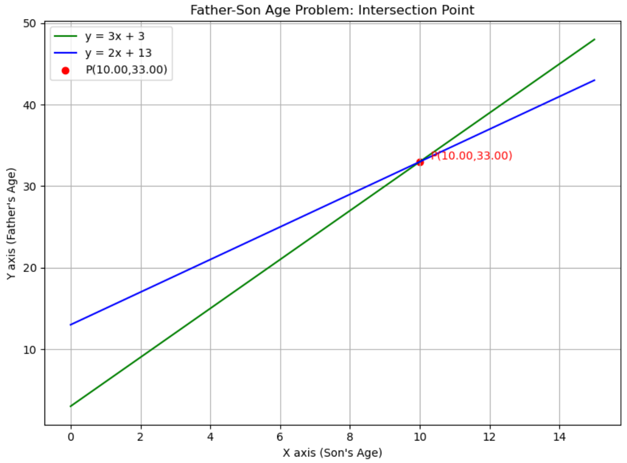

Fig : Lines