12.489

Namaswi-EE25BTECH11060

october,2025

Question

$$A = \begin{pmatrix} 2 & 0 & 2 \\ 3 & 7 & 2 \\ 5 & 1 & 7 \end{pmatrix}, b = \begin{pmatrix} 4 \\ 4 \\ 5 \end{pmatrix}$$

are given. If vector $\mathbf{x}$ is the solution to the system of equations $\mathbf{Ax} = \mathbf{b}$, which of the following is true for $\mathbf{x}$

Question

2

- el= Solution does not exist
- el= Infinite solutions exist
- el= Unique solution exists
- el= Five possible solutions exist

Given

Given,

$$A = \begin{pmatrix} 2 & 0 & 2 \\ 3 & 7 & 2 \\ 5 & 1 & 7 \end{pmatrix} \quad b = \begin{pmatrix} 4 \\ 4 \\ 5 \end{pmatrix} \tag{1}$$

Solution

Form the augmented matrix:

$$\left(\begin{array}{ccc|c} 2 & 0 & 2 & 4 \\ 3 & 7 & 2 & 4 \\ 5 & 1 & 7 & 5 \end{array}\right) \tag{2}$$

According to Gaussian elimination: Replace

$$R_2 \to R_2 - \frac{3}{2}R_1, \quad R_3 \to R_3 - \frac{5}{2}R_1$$

Solution

$$\left(\begin{array}{ccc|c} 2 & 0 & 2 & 4 \\ 0 & 7 & -1 & -2 \\ 0 & 1 & 2 & -5 \end{array}\right) \tag{3}$$

Replace

$$R_3 \to R_3 - \frac{1}{7}R_2$$

$$\left(\begin{array}{ccc|c} 2 & 0 & 2 & 4 \\ 0 & 7 & -1 & -2 \\ 0 & 0 & \frac{15}{7} & \frac{-33}{7} \end{array}\right) \tag{4}$$

Solution

Now on back-substitute:

$$\mathbf{x} = \begin{pmatrix} \frac{21}{5} \\ -\frac{3}{5} \\ -\frac{11}{5} \end{pmatrix} \tag{5}$$

Hence, a unique solution exists.

C Code

```c
    #include <stdio.h>

void solve3x3(float A[3][3], float b[3], float x[3]) {
    float ratio;
    int i, j, k;
    float aug[3][4];

    // Build augmented matrix [A|b]
    for (i = 0; i < 3; i++) {
        for (j = 0; j < 3; j++) {
            aug[i][j] = A[i][j];
        }
        aug[i][3] = b[i];
    }
```

C Code

```c
      // Forward elimination
    for (i = 0; i < 2; i++) {
        for (j = i + 1; j < 3; j++) {
            ratio = aug[j][i] / aug[i][i];
            for (k = i; k < 4; k++) {
                aug[j][k] -= ratio * aug[i][k];
            }
        }
    }
```

C Code

```c
     // Back substitution
    for (i = 2; i >= 0; i--) {
        x[i] = aug[i][3];
        for (j = i + 1; j < 3; j++) {
            x[i] -= aug[i][j] * x[j];
        }
        x[i] /= aug[i][i];
    }
}
```

Python Code

```python
  import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

# Define ranges for x and y
x = np.linspace(-2, 6, 30)
y = np.linspace(-2, 6, 30)
X, Y = np.meshgrid(x, y)
```

Python Code

```
    # Define the three planes: Ax + By + Cz = D
Z1 = (4 - 2*X - 0*Y) / 2 # 2x + 0y + 2z = 4
Z2 = (4 - 3*X - 7*Y) / 2 # 3x + 7y + 2z = 4
Z3 = (5 - 5*X - 1*Y) / 7 # 5x + y + 7z = 5

# Create 3D plot
fig = plt.figure(figsize=(9, 7))
ax = fig.add_subplot(111, projection='3d')

# Plot the planes
ax.plot_surface(X, Y, Z1, alpha=0.5, color='r', label='Plane 1')
ax.plot_surface(X, Y, Z2, alpha=0.5, color='g', label='Plane 2')
ax.plot_surface(X, Y, Z3, alpha=0.5, color='b', label='Plane 3')
```

Python Code

```python
# Intersection point (unique solution)
x_sol, y_sol, z_sol = 21/5, -3/5, -11/5
ax.scatter(x_sol, y_sol, z_sol, color='k', s=50, label='Intersection Point')
# Labels and titles
ax.set_xlabel('X-axis')
ax.set_ylabel('Y-axis')
ax.set_zlabel('Z-axis')
ax.set_title('Intersection of Three Planes: Unique Solution')
```

Python Code

```
# Legend  manually created since surfaces cant auto-label
ax.text(4, -2, 3, 'Plane 1: 2x + 2z = 4', color='r')
ax.text(-1, 5, 2, 'Plane 2: 3x + 7y + 2z = 4', color='g')
ax.text(2, 0, -2, 'Plane 3: 5x + y + 7z = 5', color='b')
ax.text(x_sol, y_sol, z_sol+1, 'Unique Solution', color='k')

plt.show()
```

C and Python Code

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

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

# Define argument and return types
solver.solve3x3.argtypes = [
    (ctypes.c_float * 3) * 3, # 3x3 matrix
    ctypes.c_float * 3, # b vector
    ctypes.c_float * 3 # x output vector
]
```

C and Python Code

```
    # Define matrices
A = np.array([[2, 0, 2],
             [3, 7, 2],
             [5, 1, 7]], dtype=np.float32)
b = np.array([4, 4, 5], dtype=np.float32)

# Prepare ctypes arrays
A_ctypes = ((ctypes.c_float * 3) * 3)(*map(lambda r: (ctypes.
    c_float * 3)(*r), A))
b_ctypes = (ctypes.c_float * 3)(*b)
x_ctypes = (ctypes.c_float * 3)()
```

C and Python Code

```
    # Call C function
solver.solve3x3(A_ctypes, b_ctypes, x_ctypes)

# Convert result back to numpy
x = np.array(list(x_ctypes))
print("Solution from C function:", x)

# === Plot the planes ===
xv = np.linspace(-2, 6, 30)
yv = np.linspace(-2, 6, 30)
X, Y = np.meshgrid(xv, yv)
Z1 = (4 - 2*X - 0*Y) / 2
Z2 = (4 - 3*X - 7*Y) / 2
Z3 = (5 - 5*X - 1*Y) / 7
```

C and Python Code

```python
    fig = plt.figure(figsize=(9, 7))
ax = fig.add_subplot(111, projection='3d')
ax.plot_surface(X, Y, Z1, alpha=0.5, color='r')
ax.plot_surface(X, Y, Z2, alpha=0.5, color='g')
ax.plot_surface(X, Y, Z3, alpha=0.5, color='b')
```

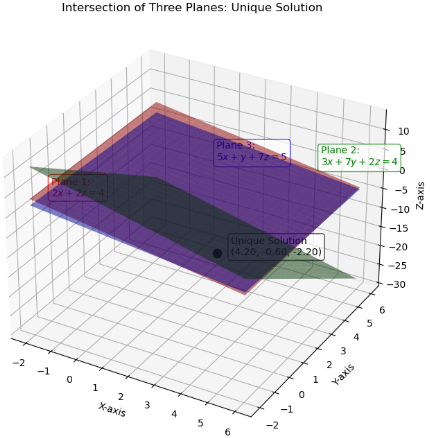
C and Python Code

```
    # Plot intersection from C
ax.scatter(x[0], x[1], x[2], color='k', s=50, label='C Solver Intersection')

ax.set_xlabel('X-axis')
ax.set_ylabel('Y-axis')
ax.set_zlabel('Z-axis')
ax.set_title('3D Planes (Solution from C Shared Library)')
ax.legend()

plt.show()
```

Plot


Intersection of Three Planes: Unique Solution
Plane 1:
$2x + 2z = 4$
Plane 2:
$3x + 7y + 2z = 4$
Plane 3:
$5x + y + 7z = 5$
Unique Solution
(4.20, -0.60, -2.20)
X-axis
Y-axis
Z-axis
−2
−1
0
1
2
3
4
5
6
10
5
0
−5
−10
−15
−20
−25
−30