MATGEO Presentation: 5.2.57

Subhodeep Chakraborty
ee25btech11055,
IIT Hyderabad.

October 2, 2025

Problem Statement

Solve the following system of linear equations.

$$
\begin{aligned}
2x + y + z &= 1 \\
x - 2y - z &= \frac{3}{2} \\
3y - 5z &= 9
\end{aligned}
$$

Given data

Given:

$$\mathbf{n_1}^\top \mathbf{x} = c_1 \qquad \mathbf{n_1} = \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix} c_1 = 1 \tag{3.1}$$

$$\mathbf{n_2}^\top \mathbf{x} = c_2 \qquad \mathbf{n_2} = \begin{pmatrix} 1 \\ -2 \\ -1 \end{pmatrix} c_2 = 3/2 \tag{3.2}$$

$$\mathbf{n_3}^\top \mathbf{x} = c_3 \qquad \mathbf{n_3} = \begin{pmatrix} 0 \\ 3 \\ -5 \end{pmatrix} c_3 = 9 \tag{3.3}$$

Formulae

Thus

$$\begin{pmatrix}\mathbf{n_1} & \mathbf{n_2} & \mathbf{n_3}\end{pmatrix}^\top \mathbf{x} = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} \tag{3.4}$$

On forming augmented matrix and applying Gaussian elimination, we can solve for $\mathbf{x}$

Solving

$$\implies \left(\begin{array}{ccc|c} 2 & 1 & 1 & 1 \\ 1 & -2 & -1 & 3/2 \\ 0 & 3 & -5 & 9 \end{array}\right) \xleftrightarrow{R_2=2R_2-R_1} \tag{3.5}$$

$$\left(\begin{array}{ccc|c} 2 & 1 & 1 & 1 \\ 0 & -5 & -3 & 2 \\ 0 & 3 & -5 & 9 \end{array}\right) \xleftrightarrow{R_3=5R_3+3R_2} \left(\begin{array}{ccc|c} 2 & 1 & 1 & 1 \\ 0 & -5 & -3 & 2 \\ 0 & 0 & -34 & 51 \end{array}\right) \tag{3.6}$$

$$\xleftrightarrow{R_3=-R_3/34;R_2=R_2+3R_3} \left(\begin{array}{ccc|c} 2 & 1 & 1 & 1 \\ 0 & -5 & 0 & -5/2 \\ 0 & 0 & 1 & -3/2 \end{array}\right) \xleftrightarrow{R_2=-R_2/5;R_1=R_1-R_2-R_3} \tag{3.7}$$

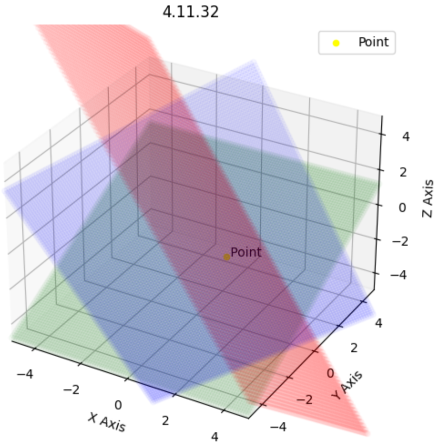
$$\left(\begin{array}{ccc|c} 2 & 0 & 0 & 2 \\ 0 & 1 & 0 & 1/2 \\ 0 & 0 & 1 & -3/2 \end{array}\right) \xleftrightarrow{R_1=R_1/2} \left(\begin{array}{ccc|c} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1/2 \\ 0 & 0 & 1 & -3/2 \end{array}\right)$$

Result

So we have:

$$\mathbf{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1 \\ 1/2 \\ -3/2 \end{pmatrix} \tag{3.9}$$

Plot



C code for generating points on plane

```
void generate_plane_points(
    // Output params
    double* x_coords, double* y_coords, double* z_coords,
    // Grid params
    double x_min, double x_max, int x_steps,
    double y_min, double y_max, int y_steps,
    // Plane stuff
    double n1, double n2, double n3, double c) {
    double x_step_val = (x_max - x_min) / (x_steps - 1);
    double y_step_val = (y_max - y_min) / (y_steps - 1);
    int index = 0;
    for (int i = 0; i < x_steps; i++) {
        for (int j = 0; j < y_steps; j++) {
            double current_x = x_min + i * x_step_val;
            double current_y = y_min + j * y_step_val;
            double current_z;
```

```
            // Vertical plane check
            if ((c < 1e-9)&&(c > -1e-9)) {
                current_z = 0.0;
            } else {
                current_z = (-n1 * current_x - n2 * current_y + c) /
                    n3;
            }
            x_coords[index] = current_x;
            y_coords[index] = current_y;
            z_coords[index] = current_z;
            index++;
        }
    }
}
```

Python code for plotting using C

```
import ctypes
import numpy as np
import numpy.linalg as LA
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

e1 = np.array([2, 1, 1, 1])
e2 = np.array([1, -2, -1, 3 / 2])
e3 = np.array([0, 3, -5, 9])



cols = ["red", "blue", "green"]
```

```
lib = ctypes.CDLL("./plane.so")

lib.generate_plane_points.argtypes = [
    ctypes.POINTER(ctypes.c_double),
    ctypes.POINTER(ctypes.c_double),
    ctypes.POINTER(ctypes.c_double),
    ctypes.c_double,
    ctypes.c_double,
    ctypes.c_int,
    ctypes.c_double,
    ctypes.c_double,
    ctypes.c_int,
    ctypes.c_double,
    ctypes.c_double,
    ctypes.c_double,
    ctypes.c_double,
]
lib.generate_plane_points.restype = None
```

```
fig = plt.figure(figsize=(8, 6))
ax = fig.add_subplot(111, projection="3d")

x_steps, y_steps = 100, 100
total_points = x_steps * y_steps
x_plane = np.zeros(total_points, dtype=np.double)
y_plane = np.zeros(total_points, dtype=np.double)
z_plane = np.zeros(total_points, dtype=np.double)
```

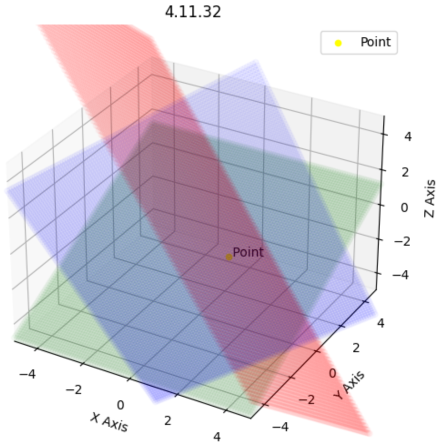
```
for i in range(1, 4):
    lib.generate_plane_points(
        x_plane.ctypes.data_as(ctypes.POINTER(ctypes.c_double)),
        y_plane.ctypes.data_as(ctypes.POINTER(ctypes.c_double)),
        z_plane.ctypes.data_as(ctypes.POINTER(ctypes.c_double)),
        -5.0,
        5.0,
        x_steps,
        -5.0,
        5.0,
        y_steps,
        eval(f"e{i}")[0],
        eval(f"e{i}")[1],
        eval(f"e{i}")[2],
        eval(f"e{i}")[3],
    )
    ax.scatter(x_plane, y_plane, z_plane, alpha=0.03, color=cols[i - 1])
```

```
ax.scatter(1, 1 / 2, -3 / 2, color="yellow", label="Point")
ax.text(1, 1 / 2, -3 / 2, " Point")

ax.set_xlabel("X Axis")
ax.set_ylabel("Y Axis")
ax.set_zlabel("Z Axis")
ax.set_title("4.11.32")
ax.set_xlim([-5, 5])
ax.set_ylim([-5, 5])
ax.set_zlim([-5, 5])
ax.legend()
ax.grid(True)

plt.savefig("../figs/plot.png")
plt.show()
```

Plot



Pure Python code

```
import numpy as np
import numpy.linalg as LA
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
e1 = np.array([2, 1, 1, 1])
e2 = np.array([1, -2, -1, 3 / 2])
e3 = np.array([0, 3, -5, 9])
cols = ["", "red", "blue", "green"]
fig = plt.figure(figsize=(8, 8))
ax = fig.add_subplot(111, projection="3d")
x, y = np.meshgrid(range(-10, 10), range(-10, 10))
for i in range(1, 4):
    z = (
        -(eval(f"e{i}")[0] * x + eval(f"e{i}")[1] * y - eval(f"e{i}")[3])
        / eval(f"e{i}")[2]
    )
    ax.plot_surface(x, y, z, alpha=0.35, color=cols[i])
```

Pure Python code

```
ax.scatter(1, 1 / 2, -3 / 2, color="yellow", label="Point")
ax.text(1, 1 / 2, -3 / 2, " Point")

ax.set_xlabel("X-axis")
ax.set_ylabel("Y-axis")
ax.set_zlabel("Z-axis")
ax.set_title("5.2.57")
ax.set_xlim([-10, 10])
ax.set_ylim([-10, 10])
ax.set_zlim([-10, 10])
ax.legend()
ax.grid(True)

plt.savefig("../figs/python.png")
plt.show()
```

Plot