4.10.13

BEERAM MADHURI - EE25BTECH11012

September 2025

Question

Find the equation of the plane passing through the line of intersection of the planes $\mathbf{r} \cdot (\hat{i} + \hat{j} + \hat{k}) = 1$ and $\mathbf{r} \cdot (2\hat{i} + 3\hat{j} - \hat{k}) + 4 = 0$ and parallel to the X axis.

given data

let P1 and P2 be the plane equations whose normals are:

Plane	**Normal vector**
P1	$\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$
P2	$\begin{pmatrix} 2 \\ 3 \\ -4 \end{pmatrix}$

Table: 4.10.13

Given equations of planes are:-

$$P_1 : \quad \mathbf{r}^\top \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = 1 \tag{1}$$

$$P_2 : \quad \mathbf{r}^\top \begin{pmatrix} 2 \\ 3 \\ -4 \end{pmatrix} = -4 \tag{2}$$

finding the equation of plane:

expressing the plane equations in matrix form:-

$$\left[\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 2 & 3 & -4 & -4 \end{array}\right] \tag{3}$$

Using row reductions:

$$R_2 \to R_2 - 2R_1 \tag{4}$$

$$\left[\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & 1 & -3 & -6 \end{array}\right] \tag{5}$$

$$\therefore \text{ equation of planes:} \tag{6}$$

$$\mathbf{r}^\top \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = 1 \tag{7}$$

$$\mathbf{r}^\top \begin{pmatrix} 0 \\ 1 \\ -3 \end{pmatrix} = -6 \tag{8}$$

Solving the equations to find the line of intersection of planes

$$\mathbf{r}(\lambda) = \begin{pmatrix} 0 \\ -\frac{3}{4} \\ \frac{7}{6} \end{pmatrix} + \lambda \begin{pmatrix} 1 \\ -3/4 \\ -1/6 \end{pmatrix} \tag{9}$$

normal to plane is orthogonal to the line and x-axis

$$\mathbf{n}^\top \mathbf{e}_1 = 0 \tag{10}$$

$$\mathbf{n}^\top \mathbf{n}_1 = 0 \tag{11}$$

$$where, \tag{12}$$

$$\mathbf{e}_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{13}$$

$$\mathbf{n}_1 = \begin{pmatrix} 1 \\ -3/4 \\ -1/6 \end{pmatrix} \tag{14}$$

Solving using row reductions:-

$$\left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 4 & -3 & -1 & 0 \end{array}\right] \tag{15}$$

$$R_2 \rightarrow R_2 - 4R_1 \tag{16}$$

$$\left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & -3 & -1 & 0 \end{array}\right] \tag{17}$$

plane equation using normal and a point on the line:

$$\mathbf{n}^{\top}(\mathbf{r} - \mathbf{r}_0) = 0 \tag{18}$$

$$\mathbf{r}_0 = \begin{pmatrix} 0 \\ -3/4 \\ -7/4 \end{pmatrix} \tag{19}$$

$$\mathbf{n} = \begin{pmatrix} 0 \\ 1 \\ -3 \end{pmatrix} \tag{20}$$

$$\mathbf{r} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{21}$$

$$\tag{22}$$

Hence, equation of the plane is: $y - 3z + 6 = 0$

Python Code

```
import numpy as np
import matplotlib.pyplot as plt
# --- 1. Setup the 3D plot ---
fig = plt.figure(figsize=(12, 9))
ax = fig.add_subplot(111, projection='3d')
ax.set_title("Visualization of Intersecting Planes", fontsize=16)
# --- 2. Define the grid for the planes ---
# Create a grid of x and y values
x = np.linspace(-10, 10, 50)
y = np.linspace(-10, 10, 50)
X, Y = np.meshgrid(x, y)
```

Python Code

```
# --- 3. Define and Plot the Three Planes ---

# Plane 1: r . (i + j + k) = 1 => x + y + z = 1
Z1 = 1 - X - Y
ax.plot_surface(X, Y, Z1, alpha=0.6, cmap='viridis', label='Plane
    1: x+y+z=1')

# Plane 2: r . (2i + 3j - k) + 4 = 0 => 2x + 3y - z = -4
Z2 = 2*X + 3*Y + 4
ax.plot_surface(X, Y, Z2, alpha=0.6, cmap='plasma', label='Plane
    2: 2x+3y-z=-4')
```

Python Code

```
# Resulting Plane: y - 3z + 6 = 0
# NOTE: This plane is parallel to the x-axis, so X is not in its
    equation.
# We define the grid differently for visualization purposes.
y_res = np.linspace(-10, 10, 50)
z_res = np.linspace(-10, 10, 50)
Y_res, Z_res = np.meshgrid(y_res, z_res)
# Equation: y - 3z + 6 = 0 => y = 3z - 6. We don't need X here.
```

Python Code

```
# To plot it, we create a constant X grid.
X_res = (3*Z_res - Y_res) * 0 # This is a trick to get a 0-filled
     grid of the right shape
# However, a better way to show a plane parallel to an axis is to
     use that axis
# in the meshgrid. Let's create a grid of x and z values instead.
x_res, z_res = np.meshgrid(np.linspace(-10,10,50), np.linspace
    (-5,5,50))
Y_res = 3*z_res - 6 # y - 3z + 6 = 0 => y = 3z - 6
ax.plot_surface(x_res, Y_res, z_res, alpha=0.7, color='cyan',
    label='Result: y-3z+6=0')
```

Python Code

```
# --- 4. Calculate and Plot the Line of Intersection ---
# Parametric equation for the line of intersection of Plane 1 and
    2
t = np.linspace(-10, 10, 100)
x_line = t
y_line = (-3 - 3*t) / 4
z_line = (7 - t) / 4
ax.plot(x_line, y_line, z_line, color='red', lw=4, label='Line of
    Intersection')
# --- 5. Plot the X-axis to show parallelism ---
ax.plot([-15, 15], [0, 0], [0, 0], color='black', lw=3, linestyle
    ='--', label='X-axis')
```

Python Code

```
# --- 6. Formatting the Plot ---
ax.set_xlabel('X axis')
ax.set_ylabel('Y axis')
ax.set_zlabel('Z axis')
# Set viewing angle for better perspective
ax.view_init(elev=20, azim=-50)
# Create proxy artists for the legend since plot_surface doesn't
    directly support labels
import matplotlib.patches as mpatches
p1 = mpatches.Patch(color='green', label='Plane 1: x+y+z=1',
    alpha=0.6)
```

Python Code

```
p2 = mpatches.Patch(color='orange', label='Plane 2: 2x+3y-z=-4',
    alpha=0.6)
p3 = mpatches.Patch(color='cyan', label='Result: y-3z+6=0', alpha
    =0.7)
from matplotlib.lines import Line2D
l1 = Line2D([0], [0], color='red', lw=4, label='Line of
    Intersection')
l2 = Line2D([0], [0], color='black', lw=3, linestyle='--', label=
    'X-axis')
ax.legend(handles=[p1, p2, p3, l1, l2], loc='upper left',
    bbox_to_anchor=(1.05, 1))
plt.tight_layout()
plt.show()
```

C Code

```c
#include <stdio.h>

// Function to compute cross product of two 3D vectors
void crossProduct(float a[3], float b[3], float result[3]) {
    result[0] = a[1]*b[2] - a[2]*b[1];
    result[1] = a[2]*b[0] - a[0]*b[2];
    result[2] = a[0]*b[1] - a[1]*b[0];
}
```

C Code

```c
int main() {
    // Define the two planes from the given question:
    // Plane 1: x + y + z = 1
    // Plane 2: 2x + 3y - z = -4

    float a1 = 1, b1 = 1, c1 = 1, d1 = 1;
    float a2 = 2, b2 = 3, c2 = -1, d2 = -4;
```

C Code

```
float n1[3] = {a1, b1, c1};
float n2[3] = {a2, b2, c2};
// Step 1: Get direction vector of line of intersection
float dir[3];
crossProduct(n1, n2, dir);
// Step 2: Find a point on the line of intersection
// Let x = 0, then solve:
// y + z = 1 => Equation A
// 3y - z = -4 => Equation B
```

C Code

```c
float y, z;
float det = b1 * c2 - b2 * c1; // b1*c2 - b2*c1 = 1*(-1) -
    3*1 = -1 - 3 = -4
if (det == 0) {
    printf("Determinant is zero, can't solve for unique point
        .\n");
    return 1;
}
y = (d1 * c2 - d2 * c1) / det;
z = (b1 * d2 - b2 * d1) / det;
float point[3] = {0, y, z};
```

C Code

```c
// Step 3: Since plane is parallel to X-axis, take vector (1,
    0, 0)
float xAxis[3] = {1, 0, 0};

// Step 4: Cross product of dir and xAxis gives normal of
    required plane
float normal[3];
crossProduct(dir, xAxis, normal);
```

C Code

```c
    // Step 5: Compute D in plane equation: ax + by + cz + d = 0
    float d = -(normal[0]*point[0] + normal[1]*point[1] + normal
        [2]*point[2]);
    printf("Equation of the required plane:\n");
    printf("%.2fx + %.2fy + %.2fz + %.2f = 0\n",
           normal[0], normal[1], normal[2], d);
return 0;
}
```

Python and C Code

```python
import ctypes
import numpy as np
import os

# Load the shared library
libname = "libgeometry.so" if os.name != "nt" else "geometry.dll"
geometry = ctypes.CDLL(libname)
```

Python and C Code

```
# Prepare cross product function signature
geometry.crossProduct.argtypes = [ctypes.POINTER(ctypes.c_float),
ctypes.POINTER(ctypes.c_float),
ctypes.POINTER(ctypes.c_float)]
geometry.crossProduct.restype = None
# --- Step 1: Define planes ---
n1 = np.array([1.0, 1.0, 1.0], dtype=np.float32)
n2 = np.array([2.0, 3.0, -1.0], dtype=np.float32)
```

Python and C Code

```
# Step 2: Get direction vector = cross(n1, n2)
dir_vector = np.zeros(3, dtype=np.float32)
geometry.crossProduct(n1.ctypes.data_as(ctypes.POINTER(ctypes.
    c_float)),
n2.ctypes.data_as(ctypes.POINTER(ctypes.c_float)),
dir_vector.ctypes.data_as(ctypes.POINTER(ctypes.c_float)))
# Step 3: Solve for a point on the intersection line (let x = 0)
# Equation A: y + z = 1
# Equation B: 3y - z = -4
```

Python and C Code

```
# Solve 2x2 system:
# From:
# y + z = 1
# 3y - z = -4
# Add: (4y = -3) => y = -0.75
# Then: z = 1 - y = 1.75
y = -0.75
z = 1.75
point = np.array([0.0, y, z], dtype=np.float32)
```

Python and C Code

```
# Step 4: Use x-axis vector
x_axis = np.array([1.0, 0.0, 0.0], dtype=np.float32)
# Step 5: Cross product of dir and x-axis = normal vector of new
    plane
normal = np.zeros(3, dtype=np.float32)
geometry.crossProduct(dir_vector.ctypes.data_as(ctypes.POINTER(
    ctypes.c_float)),
x_axis.ctypes.data_as(ctypes.POINTER(ctypes.c_float)),
normal.ctypes.data_as(ctypes.POINTER(ctypes.c_float)))
```

Python and C Code

```
# Step 6: Compute 'd' in plane equation
d = -(normal @ point) # dot product

# --- Final Result ---
print("Equation of the required plane:")
print(f"{normal[0]:.2f}x + {normal[1]:.2f}y + {normal[2]:.2f}z +
    {d:.2f} = 0")
```

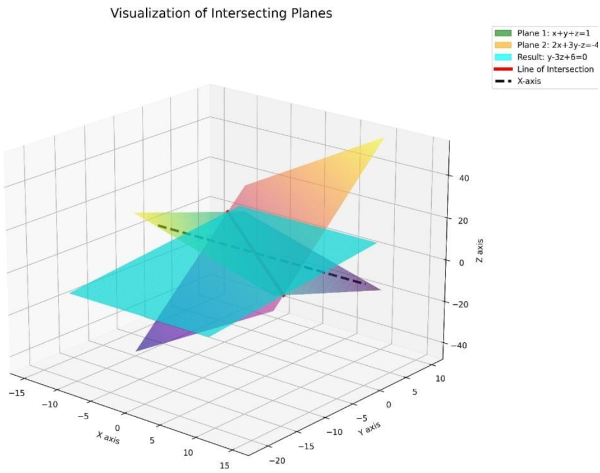
Visualization of Intersecting Planes
Plane 1: x+y+z=1
Plane 2: 2x+3y-z=-4
Result: y-3z+6=0
Line of Intersection
X-axis
40
20
0
−20
−40
Z axis
−15
−10
−5
0
5
10
15
X axis
10
5
0
−5
−10
−15
−20
Y axis

Figure: Plot