4.8.13

BALU-ai25btech11017

September 13, 2025

Question

Find the distance between the planes

$$\mathbf{r} \cdot (2\hat{i} - 3\hat{j} + 6\hat{k}) - 4 = 0 \quad \text{and} \quad \mathbf{r} \cdot (6\hat{i} - 9\hat{j} + 18\hat{k}) + 30 = 0. \qquad (1)$$

.

Theoretical Solution

Let us solve the given equation theoretically and then verify the solution computationally
According to the question,
Given two planes with direction vectors

$$\mathbf{n_1} = \begin{pmatrix} 2 \\ -3 \\ 6 \end{pmatrix} \ \mathbf{n_2} = \begin{pmatrix} 6 \\ -9 \\ 18 \end{pmatrix} \tag{2}$$

$$\mathbf{n_2} = 3\mathbf{n_1} \tag{3}$$

so the are planes are parllel
Let us take a point in plane 1

$$\mathbf{A} = \begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} \tag{4}$$

Theoretical Solution

As planes are parllel distance from **A** to plane 2 is same as distance between planes
Let distance is k

$$k = \frac{(\mathbf{A}\mathbf{n_2}^T) + 30}{\|\mathbf{n_2}\|} = 2 \tag{5}$$

C Code

```c
#include <stdio.h>
#include <math.h>

int main(void) {
    /* Planes:
       Plane 1: 2x - 3y + 6z - 4 = 0 => a=2, b=-3, c=6, d1 = -4
       Plane 2: 6x - 9y + 18z + 30 = 0 => divide by 3:
                2x - 3y + 6z + 10 = 0 => d2 = 10
    */

    double a = 2.0, b = -3.0, c = 6.0;
    double d1 = -4.0, d2 = 10.0;

    double numerator = fabs(d2 - d1); // |10 - (-4)| = 14
    double denominator = sqrt(a*a + b*b + c*c); // sqrt(4 + 9 +
        36) = sqrt(49) = 7
    double distance = numerator / denominator; // 14 / 7 = 2
```

C Code

```c
    printf("Numerator |d2 - d1| = %.0f\n", numerator);
  printf("Denominator ||n|| = sqrt(a^2+b^2+c^2) = %.0f\n",
     denominator);
  printf("Distance between the planes = %.0f\n", distance);

  return 0;
}
```

python code

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

# Define the planes:
# Plane 1: 2x - 3y + 6z - 4 = 0
# Plane 2: 6x - 9y + 18z + 30 = 0

# Normalize Plane 2 (divide by 3)  2x - 3y + 6z + 10 = 0
# Now both planes are parallel with same normal vector n = (2, -3, 6)

# Grid for plotting
xx, yy = np.meshgrid(np.linspace(-10, 10, 20), np.linspace(-10, 10, 20))

# Plane 1: solve for z
zz1 = (4 - 2*xx + 3*yy) / 6
```

python code

```
# Plane 2: solve for z
zz2 = (-10 - 2*xx + 3*yy) / 6

# Create 3D figure
fig = plt.figure(figsize=(10, 7))
ax = fig.add_subplot(111, projection='3d')

# Plot the planes
ax.plot_surface(xx, yy, zz1, alpha=0.5, color='blue', rstride
    =100, cstride=100, label='Plane 1')
ax.plot_surface(xx, yy, zz2, alpha=0.5, color='red', rstride=100,
     cstride=100, label='Plane 2')

# Normal vector
n = np.array([2, -3, 6])

# Pick a point on Plane 1 (let y=z=0, solve for x)
x0 = (4)/2 # when y=z=0
P1 = np.array([x0, 0, 0])
```

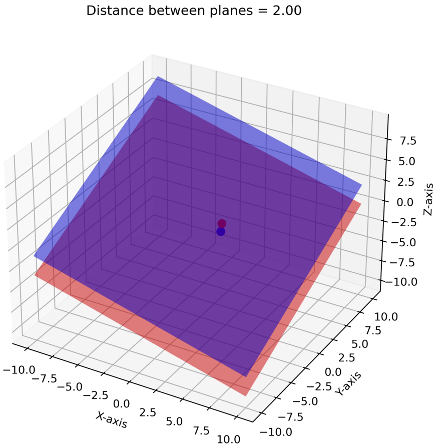
python code

```
# Distance formula: |d2 - d1| / ||n||
d1 = -4
d2 = 10
distance = abs(d2 - d1) / np.linalg.norm(n)

# Direction of normal vector (unit)
n_unit = n / np.linalg.norm(n)

# Point on Plane 2 along the normal
P2 = P1 + distance * n_unit

# Plot the connecting line (shortest distance)
ax.plot([P1[0], P2[0]], [P1[1], P2[1]], [P1[2], P2[2]], 'k--',
    linewidth=2)

# Mark points
ax.scatter(*P1, color='blue', s=50)
ax.scatter(*P2, color='red', s=50)
```

python code

```python
# Labels
ax.set_xlabel('X-axis')
ax.set_ylabel('Y-axis')
ax.set_zlabel('Z-axis')
ax.set_title(f"Distance between planes = {distance:.2f}")

# Save figure
plt.savefig("planes_distance.png", dpi=300)
plt.show()
```

Plot


Distance between planes = 2.00
X-axis
Y-axis
Z-axis
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0