4.7.14

J.NAVYASRI- EE25BTECH11028

september 2025

Question:

Find the distance of the plane 2x - 3y + 4z - 6 = 0 from the origin.

Solution:

The distance of a plane $Ax + By + Cz + D = 0$ from a point (x_0, y_0, z_0) is given by:

$$\text{Distance} = \frac{|Ax_0 + By_0 + Cz_0 + D|}{\sqrt{A^2 + B^2 + C^2}} \tag{1}$$

For the plane

$$2x - 3y + 4z - 6 = 0 \tag{2}$$

$$a = 2, b = -3, c = 4, d = -6$$

Solution:

and the origin $(0, 0, 0)$, we have:

$$\text{Distance} = \frac{|2(0) - 3(0) + 4(0) - 6|}{\sqrt{2^2 + (-3)^2 + 4^2}} = \frac{|-6|}{\sqrt{4 + 9 + 16}} = \frac{6}{\sqrt{29}} \tag{3}$$

Thus, the distance of the plane from the origin is

$$\boxed{\frac{6}{\sqrt{29}}} \tag{4}$$

Python Code

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

# Plane coefficients: 2x - 3y + 4z - 6 = 0
a, b, c, d = 2, -3, 4, -6

# Denominator
den = a*a + b*b + c*c

# Origin
origin = np.array([0.0, 0.0, 0.0])

# Foot of perpendicular from origin to plane
foot = np.array([
    -a*d/den,
    -b*d/den,
    -c*d/den
])
```

Python Code

```
# Distance from origin to plane
distance = abs(d) / np.sqrt(den)

print("Foot of perpendicular:", foot)
print("Distance:", distance)

# Create grid for plane
grid_size = 3.0
num = 40
X = np.linspace(foot[0] - grid_size, foot[0] + grid_size, num)
Y = np.linspace(foot[1] - grid_size, foot[1] + grid_size, num)
X, Y = np.meshgrid(X, Y)
Z = (6 - 2*X + 3*Y) / 4.0 # from plane equation
```

Python Code

```
# Plot
fig = plt.figure(figsize=(9,7))
ax = fig.add_subplot(111, projection='3d')

# Plane
ax.plot_surface(X, Y, Z, alpha=0.5, color='orange')

# Origin and foot points
ax.scatter(*origin, color='red', s=80, label="Origin")
ax.scatter(*foot, color='blue', s=80, label="Foot of
    perpendicular")
```

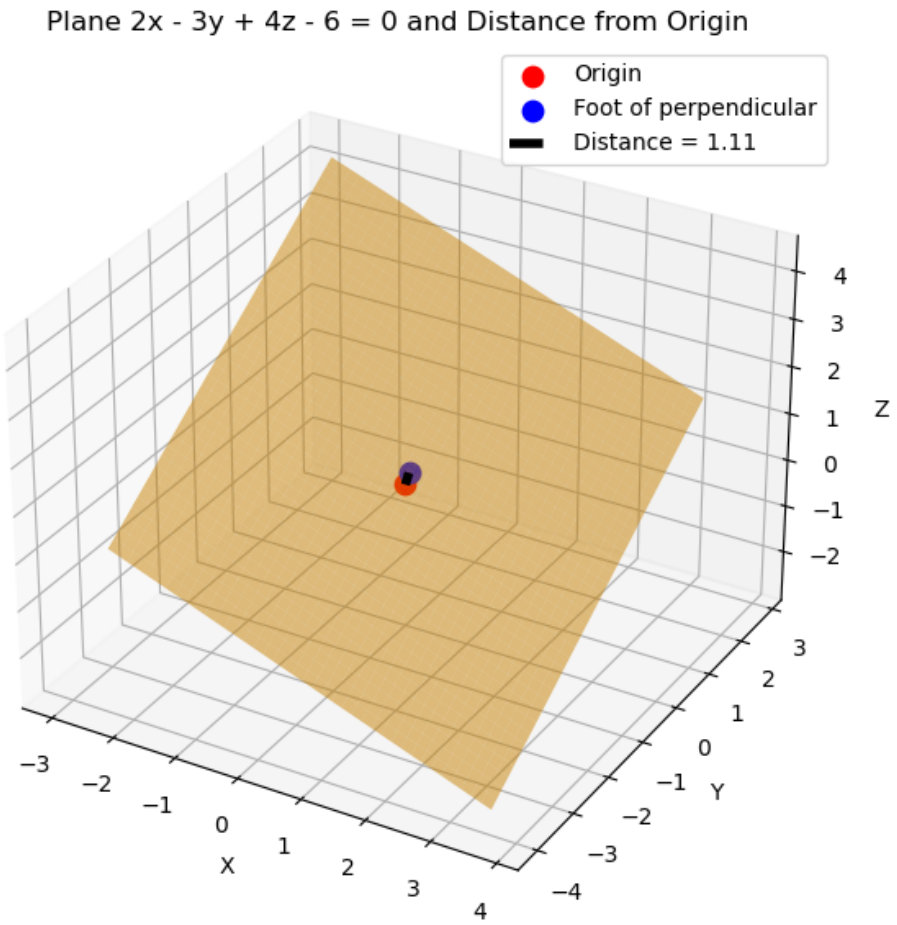
Python Code

```
 # Perpendicular line (distance line)
ax.plot(
    [origin[0], foot[0]],
    [origin[1], foot[1]],
    [origin[2], foot[2]],
    color='black', linewidth=4, linestyle='--',
    zorder=10, label=f"Distance = {distance:.2f}"
)

# Labels
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.set_title('Plane 2x - 3y + 4z - 6 = 0 and Distance from Origin
    ')
```

Python Code

```
# Equal aspect ratio
max_range = np.array([X.max()-X.min(), Y.max()-Y.min(), Z.max()-Z
    .min()]).max() / 2.0
mid_x = (X.max()+X.min()) * 0.5
mid_y = (Y.max()+Y.min()) * 0.5
mid_z = (Z.max()+Z.min()) * 0.5
ax.set_xlim(mid_x - max_range, mid_x + max_range)
ax.set_ylim(mid_y - max_range, mid_y + max_range)
ax.set_zlim(mid_z - max_range, mid_z + max_range)

# Legend
ax.legend()
plt.savefig("fig7.png")
plt.show()
```

Plot-Using Python



C Code

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

int main() {
    double A = 2, B = -3, C = 4, D = -6;

    double x0 = 0, y0 = 0, z0 = 0;

    double numerator = fabs(A*x0 + B*y0 + C*z0 + D);
    double denominator = sqrt(A*A + B*B + C*C);
    double distance = numerator / denominator;

    printf("Distance of plane %.0fx + (%.0f)y + %.0fz + (%.0f) =
        0 from origin is: %lf\n",
           A, B, C, D, distance);
    return 0;
}
```

Python and C Code

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

# Load the shared library
lib = ctypes.CDLL("./libdistance.so")

# Define argument and return types
lib.plane_distance.argtypes = [ctypes.c_double, ctypes.c_double,
                               ctypes.c_double, ctypes.c_double,
                               ctypes.c_double, ctypes.c_double,
                               ctypes.c_double]
lib.plane_distance.restype = ctypes.c_double

# Plane coefficients
A, B, C, D = 2, -3, 4, -6
x0, y0, z0 = 0.0, 0.0, 0.0 # origin
```

Python and C Code

```
# Call C function
dist = lib.plane_distance(A, B, C, D, x0, y0, z0)
print(f"Distance from origin = {dist:.4f}")

# ---------- PLOT ----------
# Generate grid for plane
xx, yy = np.meshgrid(np.linspace(-5, 5, 20), np.linspace(-5, 5,
    20))
zz = (-A*xx - B*yy - D) / C

# Normal vector
normal = np.array([A, B, C])
normal = normal / np.linalg.norm(normal)
```

Python and C Code

```
# Closest point on plane to origin = -D * (n / |n|)
closest_point = -D * normal / (A*normal[0] + B*normal[1] + C*
    normal[2])

# Plot
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

# Plane surface
ax.plot_surface(xx, yy, zz, alpha=0.5, color='cyan')

# Origin
ax.scatter([0], [0], [0], color='red', s=50, label="Origin")
```

Python and C Code

```
# Closest point
ax.scatter([closest_point[0]], [closest_point[1]], [closest_point
    [2]],
           color='blue', s=50, label="Closest Point")

# Distance line
ax.plot([0, closest_point[0]],
        [0, closest_point[1]],
        [0, closest_point[2]], color='black', linewidth=2)

ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.set_title(f"Distance = {dist:.4f}")

ax.legend()
plt.show()
```

Plot-Using by C and Python