4.7.42

Navya Priya - EE25BTECH11045

October 1,2025

Question

Find the length and the foot of perpendicular from the point $\left(1, \frac{3}{2}, 2\right)$ to the plane $2x - 2y + 4z + 5 = 0$.

Theoretical Solution

Given plane equation $2x - 2y + 4z + 5 = 0$ can be written as

$$\mathbf{n}^\top \mathbf{x} = c \tag{1}$$

Where

$$\mathbf{n} = \begin{pmatrix} 2 \\ -2 \\ 4 \end{pmatrix} \text{ and } c = -5$$

Let the point be $\mathbf{p} \begin{pmatrix} 1 \\ \frac{3}{2} \\ 2 \end{pmatrix}$ and the point on the plane be $\mathbf{x}_o$. The equation of the line joining $\mathbf{p}$ and $\mathbf{x_o}$ is

$$\mathbf{x_o} = \mathbf{p} + \lambda \mathbf{n} \tag{2}$$

Foot of Perpendicular

Multiply equation(2) on both sides by $\mathbf{n}^\top$

$$\mathbf{n}^\top(\mathbf{x_o}) = \mathbf{n}^\top(\mathbf{p} + \lambda\mathbf{n}) \tag{3}$$

$$\lambda = \frac{\mathbf{n}^\top\mathbf{x}_o}{\mathbf{n}^\top\mathbf{p} + \lambda\mathbf{n}^\top\mathbf{n}} \tag{4}$$

$$\lambda = \frac{-5}{\begin{pmatrix}2 & -2 & 4\end{pmatrix}\begin{pmatrix}1\\ \frac{3}{2}\\ 2\end{pmatrix} + \lambda\begin{pmatrix}2 & -2 & 4\end{pmatrix}\begin{pmatrix}2\\ -2\\ 4\end{pmatrix}} (\because \mathbf{n}^\top\mathbf{x_o} = -5) \tag{5}$$

$$\lambda = -\frac{1}{2} \tag{6}$$

Foot of Perpendicular

Substitute the value of λ in equation(2) to get x_o

$$\mathbf{x}_o = \begin{pmatrix} 0 \\ \frac{5}{2} \\ 0 \end{pmatrix} \tag{7}$$

$\therefore$ **Foot of perpendicular** is $\mathbf{x}_o = \begin{pmatrix} 0 \\ \frac{5}{2} \\ 0 \end{pmatrix}$

Length

The length of point $\begin{pmatrix} 1 \\ \frac{3}{2} \\ 2 \end{pmatrix}$ to the plane $2x - 2y + 4z + 5 = 0$ is

$$\begin{aligned} \|\mathbf{x_o} - \mathbf{p}\| &= \sqrt{(\mathbf{x_o} - \mathbf{p})^\top (\mathbf{x_o} - \mathbf{p})} && (8) \\ &= \sqrt{6} && (9) \end{aligned}$$

$$\therefore \|\mathbf{x_o} - \mathbf{p}\| = \sqrt{6}$$

C code

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

void foot_and_length(double px, double py, double pz,
                     double a, double b, double c, double d,
                     double *x0, double *y0, double *z0, double *
                         dist) {
    // Normal vector n = (a, b, c)
    double numerator = a*px + b*py + c*pz + d;
    double denominator = a*a + b*b + c*c;
    double lambda = - numerator / denominator;

    *x0 = px + lambda * a;
    *y0 = py + lambda * b;
    *z0 = pz + lambda * c;

    *dist = fabs(numerator) / sqrt(denominator);
}
```

C code

```c
// For testing
int main() {
    double px = 1, py = 1.5, pz = 2;
    double a = 2, b = -2, c = 4, d = 5;
    double x0, y0, z0, dist;

    foot_and_length(px, py, pz, a, b, c, d, &x0, &y0, &z0, &dist)
        ;

    printf("Foot of perpendicular: (%.4f, %.4f, %.4f)\n", x0, y0,
         z0);
    printf("Perpendicular length: %.4f\n", dist);

    return 0;
}
```

CallC.py

```
import ctypes

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

# Define argument and return types
lib.foot_and_length.argtypes = [ctypes.c_double, ctypes.c_double,
     ctypes.c_double,
                              ctypes.c_double, ctypes.c_double,
                                  ctypes.c_double, ctypes.c_double
                                  ,
                              ctypes.POINTER(ctypes.c_double),
                              ctypes.POINTER(ctypes.c_double),
                              ctypes.POINTER(ctypes.c_double),
                              ctypes.POINTER(ctypes.c_double)]
```

CallC.py

```
# Inputs
px, py, pz = 1.0, 1.5, 2.0
a, b, c, d = 2.0, -2.0, 4.0, 5.0

# Outputs
x0 = ctypes.c_double()
y0 = ctypes.c_double()
z0 = ctypes.c_double()
dist = ctypes.c_double()

# Call the C function
lib.foot_and_length(px, py, pz, a, b, c, d,
                    ctypes.byref(x0), ctypes.byref(y0), ctypes.
                        byref(z0), ctypes.byref(dist))

print(f"Foot of perpendicular = ({x0.value:.4f}, {y0.value:.4f},
    {z0.value:.4f})")
print(f"Length of perpendicular = {dist.value:.4f}")
```

Plot.py

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

# Plane: 2x - 2y + 4z + 5 = 0
# Point: p(1, 3/2, 2)
p = np.array([1.0, 1.5, 2.0])
n = np.array([2.0, -2.0, 4.0])

# Foot of perpendicular (xo)
lam = - (np.dot(n, p) + 5) / np.dot(n, n)
xo = p + lam * n

# Distance (perpendicular length)
distance = abs(np.dot(n, p) + 5) / LA.norm(n)

# Create mesh for plane
x_vals = np.linspace(-2, 2, 40)
```

Plot.py

```
y_vals = np.linspace(0, 3.5, 40)
xx, yy = np.meshgrid(x_vals, y_vals)
zz = (-2*xx + 2*yy - 5)/4

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

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

# Scatter points
ax.scatter(*p, color='red', s=80, label='p(1, 3/2, 2)')
ax.scatter(*xo, color='blue', s=80, label=f'xo(0, 2.5, 0)')

# Perpendicular line pxo
ax.plot([p[0], xo[0]], [p[1], xo[1]], [p[2], xo[2]],
        color='black', linestyle='--', linewidth=2, label='pxo
            plane')
```

Plot.py

```
# ---- Draw right-angle square marker at xo ----
eps = 0.15 # size of the square
# Two perpendicular directions in the plane
u = np.cross(n, [1,0,0])
if LA.norm(u) == 0:
    u = np.cross(n, [0,1,0])
u = u / LA.norm(u)
v = np.cross(n, u)
v = v / LA.norm(v)

# Tiny square centered at xo
square_pts = [xo, xo + eps*u, xo + eps*u + eps*v, xo + eps*v, xo]
ax.plot([pt[0] for pt in square_pts],
        [pt[1] for pt in square_pts],
        [pt[2] for pt in square_pts], color="k")
```

Plot.py

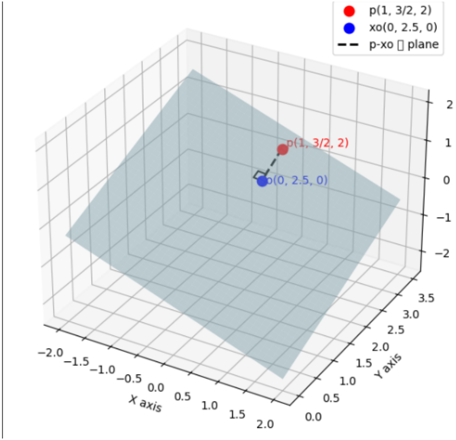
```python
# Annotations
ax.text(p[0]+0.05, p[1]+0.05, p[2]+0.05, "p(1, 3/2, 2)", color='
    red')
ax.text(xo[0]+0.05, xo[1]-0.15, xo[2]+0.05, "xo(0, 2.5, 0)",
    color='blue')

# Labels and title
ax.set_xlabel('X axis')
ax.set_ylabel('Y axis')
ax.set_zlabel('Z axis')
ax.set_title(f'Perpendicular from p to plane (Length = {distance
    :.4f})')
ax.legend()

plt.show()
```

Plot

From the graph, theoretical solution matches with the computational solution.



Perpendicular to the plane (length= $\sqrt{6}$)