4.7.40

EE25BTECH11043 - Nishid Khandagre

September 29, 2025

Question

Find the foot of the perpendicular and the perpendicular distance from the point $\begin{pmatrix} 2 \\ 3 \\ -8 \end{pmatrix}$ to the line $\frac{4-x}{2} = \frac{y}{6} = \frac{1-z}{3}$.

Theoretical Solution

Given line:

$$\frac{4-x}{2} = \frac{y}{6} = \frac{1-z}{3} = t \tag{1}$$

$$x = 4 - 2t \tag{2}$$
$$y = 6t \tag{3}$$
$$z = 1 - 3t \tag{4}$$

Line in vector form:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 4 \\ 0 \\ 1 \end{pmatrix} + t \begin{pmatrix} -2 \\ 6 \\ -3 \end{pmatrix} \tag{5}$$

$$\mathbf{r} = \mathbf{a} + t\mathbf{m} \tag{6}$$

Theoretical Solution

$$\mathbf{a} = \begin{pmatrix} 4 \\ 0 \\ 1 \end{pmatrix} \tag{7}$$

$$\mathbf{m} = \begin{pmatrix} -2 \\ 6 \\ -3 \end{pmatrix} \tag{8}$$

The given point is $\mathbf{p} = \begin{pmatrix} 2 \\ 3 \\ -8 \end{pmatrix}$

Let the foot of the perpendicular be $\mathbf{f}$. Since $\mathbf{f}$ lies on the line, we can write:

$$\mathbf{f} = \mathbf{a} + \alpha \mathbf{m} \tag{9}$$

$\left(\mathbf{p} - \mathbf{f}\right)$ must be orthogonal to the direction vector of the line $\mathbf{m}$.

Therefore

$$\left(\mathbf{p} - \mathbf{f}\right)^\top \mathbf{m} = 0 \tag{10}$$

$$\left(\mathbf{p} - \left(\mathbf{a} + \alpha \mathbf{m}\right)\right)^\top \mathbf{m} = 0 \tag{11}$$

$$\left(\mathbf{p} - \mathbf{a} - \alpha \mathbf{m}\right)^\top \mathbf{m} = 0 \tag{12}$$

$$\left(\mathbf{p} - \mathbf{a}\right)^\top \mathbf{m} - \alpha \left(\mathbf{m}^\top \mathbf{m}\right) = 0 \tag{13}$$

Theoretical Solution

$$\alpha = \frac{\left(\mathbf{p}-\mathbf{a}\right)^{\top}\mathbf{m}}{\mathbf{m}^{\top}\mathbf{m}} \tag{14}$$

$$\mathbf{p}-\mathbf{a} = \begin{pmatrix} 2 \\ 3 \\ -8 \end{pmatrix} - \begin{pmatrix} 4 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -2 \\ 3 \\ -9 \end{pmatrix} \tag{15}$$

$$\left(\mathbf{p}-\mathbf{a}\right)^{\top}\mathbf{m} = \begin{pmatrix} -2 & 3 & -9 \end{pmatrix} \begin{pmatrix} -2 \\ 6 \\ -3 \end{pmatrix} \tag{16}$$

$$= (-2)(-2) + (3)(6) + (-9)(-3) \tag{17}$$

$$= 4 + 18 + 27 = 49 \tag{18}$$

Theoretical Solution

$$\mathbf{m}^\top \mathbf{m} = (-2)^2 + 6^2 + (-3)^2 \tag{19}$$
$$= 4 + 36 + 9 = 49 \tag{20}$$

Therefore

$$\alpha = \frac{49}{49} = 1 \tag{21}$$

foot of the perpendicular **f**:

$$\mathbf{f} = \mathbf{a} + \alpha \mathbf{m} \tag{22}$$
$$= \begin{pmatrix} 4 \\ 0 \\ 1 \end{pmatrix} + 1 \begin{pmatrix} -2 \\ 6 \\ -3 \end{pmatrix} \tag{23}$$
$$= \begin{pmatrix} 2 \\ 6 \\ -2 \end{pmatrix} \tag{24}$$

Theoretical Solution

$$\text{Perpendicular Distance} = \|\mathbf{p} - \mathbf{f}\| \tag{25}$$

$$\mathbf{p} - \mathbf{f} = \begin{pmatrix} 2 \\ 3 \\ -8 \end{pmatrix} - \begin{pmatrix} 2 \\ 6 \\ -2 \end{pmatrix} \tag{26}$$

$$= \begin{pmatrix} 0 \\ -3 \\ -6 \end{pmatrix} \tag{27}$$

Theoretical Solution

$$\|\mathbf{p} - \mathbf{f}\| = \sqrt{\left(\mathbf{p} - \mathbf{f}\right)^{\top} \left(\mathbf{p} - \mathbf{f}\right)} \tag{28}$$

$$= \sqrt{0^2 + (-3)^2 + (-6)^2} \tag{29}$$

$$= \sqrt{0 + 9 + 36} \tag{30}$$

$$= \sqrt{45} \tag{31}$$

$$= 3\sqrt{5} \tag{32}$$

The perpendicular distance is $3\sqrt{5}$.

C Code

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

// Function to find the foot of the perpendicular and the
    perpendicular distance
// from a point (x0, y0, z0) to a line
// (x - x1)/a = (y - y1)/b = (z - z1)/c
void find_perpendicular_details(
double x0, double y0, double z0, // Point P
double x1, double y1, double z1, // Point on the line L
double a, double b, double c, // Direction ratios of the line L
double *foot_x, double *foot_y, double *foot_z, // Output: Foot
    of perpendicular
double *distance // Output: Perpendicular distance
) {
```

C Code

```c

// Vector PL (P - L)
double PL_x = x0 - x1;
double PL_y = y0 - y1;
double PL_z = z0 - z1;

// Direction vector of the line L (D)
double D_x = a;
double D_y = b;
double D_z = c;

// Calculate projection of PL onto D: t = (PL . D) / ||D||^2
double dot_product = PL_x * D_x + PL_y * D_y + PL_z * D_z;
double magnitude_D_squared = D_x * D_x + D_y * D_y + D_z * D_z;

double t = dot_product / magnitude_D_squared;
```

C Code

```
// Foot of the perpendicular F = L + t * D
*foot_x = x1 + t * D_x;
*foot_y = y1 + t * D_y;
*foot_z = z1 + t * D_z;

// Perpendicular vector PF = F - P
double PF_x = *foot_x - x0;
double PF_y = *foot_y - y0;
double PF_z = *foot_z - z0;

// Perpendicular distance = ||PF||
*distance = sqrt(PF_x * PF_x + PF_y * PF_y + PF_z * PF_z);

}
```

Python Code using C shared output

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

# Load the shared library
lib_perpendicular = ctypes.CDLL(./code7.so)

# Define the argument types and return type for the C function
lib_perpendicular.find_perpendicular_details.argtypes = [
ctypes.c_double, ctypes.c_double, ctypes.c_double, # P(x0, y0, z0)
ctypes.c_double, ctypes.c_double, ctypes.c_double, # L(x1, y1, z1)
ctypes.c_double, ctypes.c_double, ctypes.c_double, # D(a, b, c)
```

Python Code using C shared output

```
ctypes.POINTER(ctypes.c_double), # foot_x
ctypes.POINTER(ctypes.c_double), # foot_y
ctypes.POINTER(ctypes.c_double), # foot_z
ctypes.POINTER(ctypes.c_double) # distance
]
lib_perpendicular.find_perpendicular_details.restype = None

# Given point P
P_x, P_y, P_z = 2.0, 3.0, -8.0

# Given line: (4 - x)/2 = y/6 = (1 - z)/3
# Rewrite in standard form: (x - x1)/a = (y - y1)/b = (z - z1)/c
# (x - 4)/(-2) = (y - 0)/6 = (z - 1)/(-3)
# Point on the line L
L_x, L_y, L_z = 4.0, 0.0, 1.0

# Direction ratios of the line D
D_a, D_b, D_c = -2.0, 6.0, -3.0
```

Python Code using C shared output

```
# Create ctypes doubles to hold the results
foot_x_result = ctypes.c_double()
foot_y_result = ctypes.c_double()
foot_z_result = ctypes.c_double()
distance_result = ctypes.c_double()

# Call the C function
lib_perpendicular.find_perpendicular_details(
P_x, P_y, P_z,
L_x, L_y, L_z,
D_a, D_b, D_c,
ctypes.byref(foot_x_result),
ctypes.byref(foot_y_result),
ctypes.byref(foot_z_result),
ctypes.byref(distance_result)
)
```

Python Code using C shared output

```
foot_x_found = foot_x_result.value
foot_y_found = foot_y_result.value
foot_z_found = foot_z_result.value
distance_found = distance_result.value

print(fThe foot of the perpendicular is ({foot_x_found:.2f}, {foot_y_found:.2f}, {foot_z_found:.2f}))
print(fThe perpendicular distance is {distance_found:.2f})

# Plotting
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')

# Plot the given point P
ax.scatter(P_x, P_y, P_z, color='black', s=100, label=f'Point P ({P_x},{P_y},{P_z})')
ax.text(P_x, P_y, P_z, f' P ', color='black')
```

Python Code using C shared output

```python
# Plot the foot of the perpendicular F
ax.scatter(foot_x_found, foot_y_found, foot_z_found, color='red',
     s=100, label=f'Foot of Perpendicular F ({foot_x_found:.2f},{
    foot_y_found:.2f},{foot_z_found:.2f})')
ax.text(foot_x_found, foot_y_found, foot_z_found, f' F ', color='
    red')

# Plot the line
# Parameter t for the line equation
t = np.linspace(-5, 5, 100) # Extend the line for better
    visualization
line_x = L_x + t * D_a
line_y = L_y + t * D_b
line_z = L_z + t * D_c
ax.plot(line_x, line_y, line_z, color='green', label='Line L')
```

Python Code using C shared output

```
# Plot the perpendicular line segment PF
ax.plot([P_x, foot_x_found], [P_y, foot_y_found], [P_z,
    foot_z_found], color='purple', linestyle='--', label='
    Perpendicular PF')

ax.set_xlabel('X-axis')
ax.set_ylabel('Y-axis')
ax.set_zlabel('Z-axis')
ax.set_title('Foot of Perpendicular and Perpendicular Distance in
     3D')
ax.legend()
ax.grid(True)
plt.savefig(fig1.png)
plt.show()
```

Python Code (Direct)

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

def line_gen_num(point1, point2, num_points):

    Generates points along a line segment between two 3D points.

    point1 = np.array(point1).flatten()
    point2 = np.array(point2).flatten()
    t = np.linspace(0, 1, num_points)
    points = np.outer(point1, (1-t)) + np.outer(point2, t)
    return points

def plot_3d_line(ax, point1, point2, label=, color=blue,
     linestyle=-):
```

Python Code (Direct)

```
    Plots a 3D line segment on a given matplotlib axis.

    line_points = line_gen_num(point1, point2, 2)
    ax.plot(line_points[0], line_points[1], line_points[2], color
        =color, linestyle=linestyle, label=label)

# Given point P
P = np.array([2, 3, -8])

# Given line in symmetric form: (4-x)/2 = y/6 = (1-z)/3
# Rewrite in standard form: (x-4)/(-2) = (y-0)/6 = (z-1)/(-3)
# This means the line passes through point A = (4, 0, 1) and has
    direction vector d = (-2, 6, -3)
A_line = np.array([4, 0, 1])
d_line = np.array([-2, 6, -3])
```

Python Code (Direct)

```
# The foot of the perpendicular M on the line can be represented
    as:
# M = A_line + t * d_line = (4 - 2t, 6t, 1 - 3t)
# The vector PM is perpendicular to the direction vector d_line
# PM = M - P = (4 - 2t - 2, 6t - 3, 1 - 3t - (-8))
# PM = (2 - 2t, 6t - 3, 9 - 3t)
# The dot product of PM and d_line must be zero: PM . d_line = 0
# (2 - 2t)(-2) + (6t - 3)(6) + (9 - 3t)(-3) = 0
# -4 + 4t + 36t - 18 - 27 + 9t = 0
# 49t - 49 = 0
# 49t = 49 => t = 1
t_val = 1
```

Python Code (Direct)

```
# Calculate the coordinates of the foot of the perpendicular M
M = A_line + t_val * d_line
# M = np.array([4 - 2 * t_val, 6 * t_val, 1 - 3 * t_val]) # Alternative calculation
print(fThe foot of the perpendicular is M = ({M[0]}, {M[1]}, {M[2]}))

# Calculate the perpendicular distance from P to the line
perpendicular_distance = np.linalg.norm(P - M)
print(fThe perpendicular distance from P to the line is = {perpendicular_distance:.2f})

# --- Plotting the 3D scene ---
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')
```

Python Code (Direct)

```
# Plot the given point P
ax.scatter(P[0], P[1], P[2], color='black', s=100, label=f'Point P({P[0]},{P[1]},{P[2]})')
ax.text(P[0], P[1], P[2] + 0.5, ' P ', color='black')

# Plot the foot of the perpendicular M
ax.scatter(M[0], M[1], M[2], color='green', s=100, label=f'Foot of Perpendicular M({M[0]},{M[1]},{M[2]})')
ax.text(M[0], M[1], M[2] + 0.5, ' M ', color='green')

# Plot the line
# We have A_line and M is also on the line. Pick another point using t = 0 (A_line) and t = 2
line_points_for_plot = np.array([
    A_line + 5 * d_line,
    A_line - 5 * d_line
])
```

Python Code (Direct)

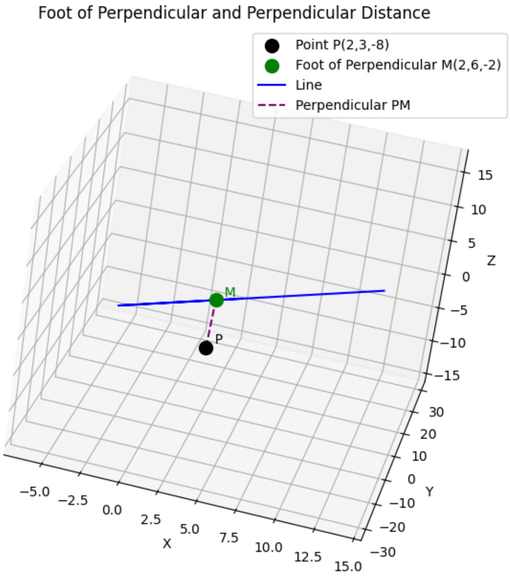
```
ax.plot(line_points_for_plot[:,0], line_points_for_plot[:,1],
    line_points_for_plot[:,2], color='blue', label='Line')

# Plot the perpendicular line segment PM
plot_3d_line(ax, P, M, label='Perpendicular PM', color='purple',
    linestyle='--')

ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.set_title('Foot of Perpendicular and Perpendicular Distance')
ax.legend()
ax.grid(True)
plt.tight_layout()
plt.savefig(fig2.png)
plt.show()

print(3D plot saved as fig2.png)
```

Plot by Python using shared output from C



Plot by Python only