4.7.57

Namaswi-EE25BTECH11060

september 14,2025

Question

Find distance of $(3, -5)$ from line 3x-4y-26=0

solution

The given line is

$$3x - 4y - 26 = 0$$

This can be written in the form

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

where

$$\mathbf{n} = \begin{pmatrix} 3 \\ -4 \end{pmatrix}, \quad c = 26.$$

Let the point be

$$\mathbf{P} = \begin{pmatrix} 3 \\ -5 \end{pmatrix}.$$

solution

The distance of point P from the line is

$$d = \frac{|\mathbf{n}^\top \mathbf{P} - c|}{\|\mathbf{n}\|}. \tag{2}$$

$$\mathbf{n}^\top \mathbf{P} = \begin{pmatrix} 3 & -4 \end{pmatrix} \begin{pmatrix} 3 \\ -5 \end{pmatrix} \tag{3}$$

$$= 3(3) + (-4)(-5) \tag{4}$$

$$= 9 + 20 = 29. \tag{5}$$

$$\mathbf{n}^\top \mathbf{P} - c \tag{6}$$

$$= 29 - 26 = 3. \tag{7}$$

$$\|\mathbf{n}\| = \sqrt{3^2 + (-4)^2} \tag{8}$$

$$= \sqrt{9 + 16} = 5. \tag{9}$$

$$So, d = \frac{|3|}{5} = \frac{3}{5}. \tag{10}$$

C Code

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

int main() {
    // Line: 3x - 4y - 26 = 0
    double n[2] = {3, -4}; // normal vector
    double c = 26;
    double P[2] = {3, -5}; // point (3, -5)

    // Compute n^T * P
    double dot = n[0]*P[0] + n[1]*P[1];
```

C Code

```
 // Numerator |n^T P - c|
    double numerator = fabs(dot - c);
    // Denominator ||n||
    double norm = sqrt(n[0]*n[0] + n[1]*n[1]);
    // Distance
    double distance = numerator / norm;
    printf("Distance = %lf\n", distance);
    return 0;
}
```

Python Code

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

# Line: 3x - 4y - 26 = 0
n = np.array([3, -4]) # normal vector
c = 26
P = np.array([3, -5]) # given point

# Foot of perpendicular formula: Q = P - ((n^T P - c)/||n||^2) *
    n
dot = n @ P
Q = P - ((dot - c) / (np.dot(n, n))) * n

# Prepare line for plotting
x_vals = np.linspace(-10, 10, 400)
y_vals = (3*x_vals - 26)/4 # from 3x - 4y - 26 = 0
```

Python Code

```
# Plot line
plt.plot(x_vals, y_vals, 'b', label=r'$3x-4y-26=0$')

# Plot point P
plt.scatter(P[0], P[1], color='red', zorder=5)
plt.text(P[0]+0.3, P[1]-0.3, 'P(3,-5)', fontsize=12, color='red')

# Plot foot of perpendicular Q
plt.scatter(Q[0], Q[1], color='green', zorder=5)
plt.text(Q[0]+0.3, Q[1]+0.3, f'Q({Q[0]:.2f},{Q[1]:.2f})',
    fontsize=12, color='green')
```

Python Code

```
# Dotted line PQ
plt.plot([P[0], Q[0]], [P[1], Q[1]], 'r--', label='Shortest
    Distance')

# Axes & labels
plt.axhline(0, color='black', linewidth=0.8)
plt.axvline(0, color='black', linewidth=0.8)
plt.gca().set_aspect('equal', adjustable='box')
plt.legend()
plt.grid(True, linestyle='--', alpha=0.6)
plt.title("Distance from Point to Line")
plt.show()
```

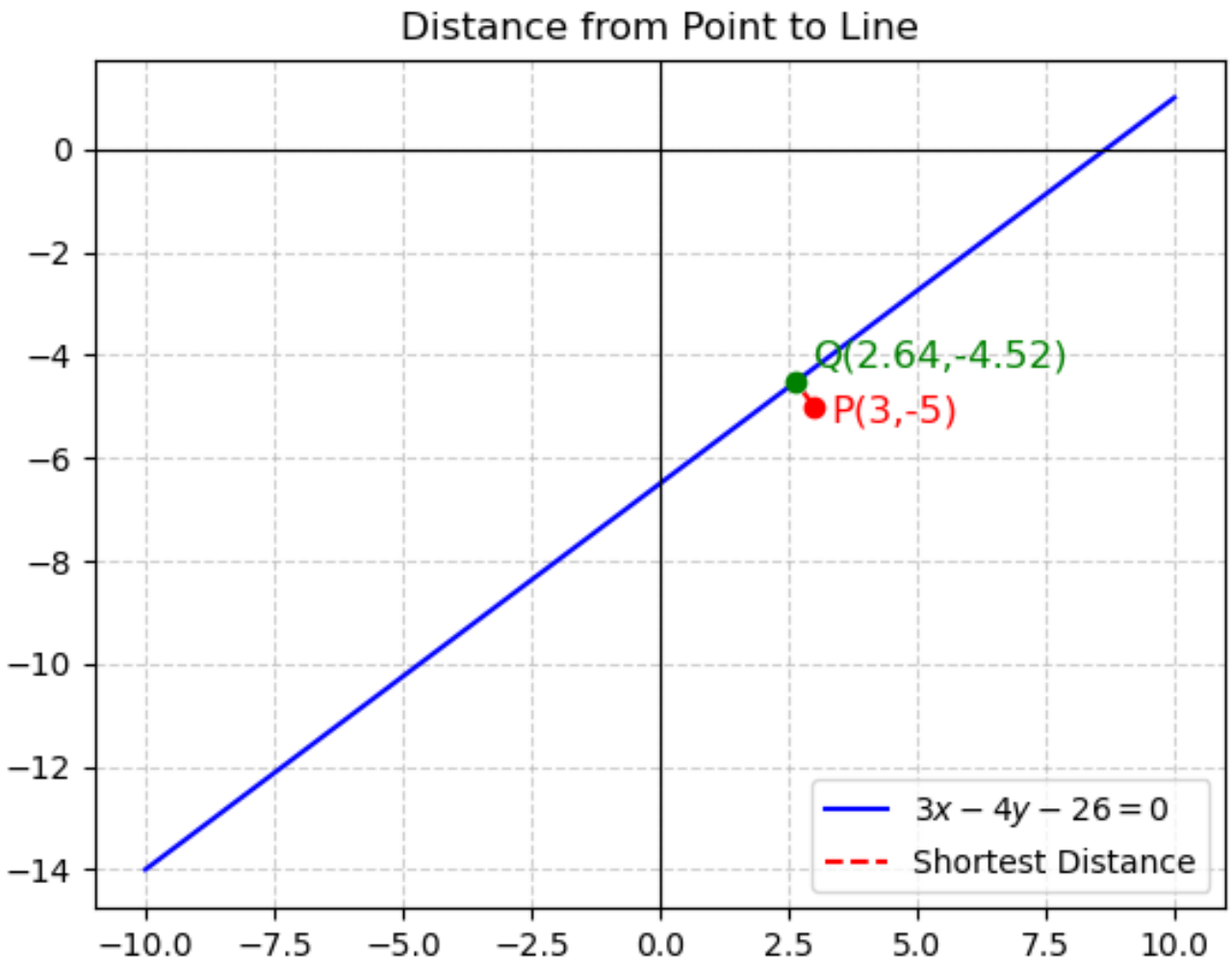
C and Python Code

```
import ctypes

# Load the shared object file
lib = ctypes.CDLL('./distance.so')

# Set argument and return types
lib.point_to_line_distance.argtypes = [ctypes.c_double, ctypes.
    c_double, ctypes.c_double,
                                       ctypes.c_double, ctypes.
                                           c_double]
lib.point_to_line_distance.restype = ctypes.c_double
```

C and Python Code

```
# Line: 3x - 4y - 26 = 0  a=3, b=-4, c=-26
a, b, c = 3, -4, -26
px, py = 3, -5 # Point

# Call the C function
distance = lib.point_to_line_distance(a, b, c, px, py)
print(f"Distance from P({px},{py}) to line 3x-4y-26=0 is {
    distance}")
```

Plot


Distance from Point to Line
0
−2
−4
−6
−8
−10
−12
−14
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
Q(2.64,-4.52)
P(3,-5)
3x − 4y − 26 = 0
Shortest Distance