4.7.39

Nipun Dasari - EE25BTECH11042

September 20, 2025

Question

The distance of the point P(2, 3) from the x-axis is?

Theoretical Solution

Consider the general line equation where

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

Using the fact that all y-coordinates of x axis are zero

$$\mathbf{n} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \text{ and } c = 0 \tag{2}$$

The distance between a **p** to its foot of perpendicular to a line is:

Theoretical Solution

$$\text{Distance} = \frac{|\mathbf{n}^\top \mathbf{p} - c|}{\|\mathbf{n}\|} \tag{3}$$

By (3) and (2):

$$\text{Distance} = \frac{|\begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} 2 \\ 3 \end{pmatrix} - 0|}{1} \tag{4}$$

$$= 2 \times 0 + 3 \times 1 = 3 \tag{5}$$

Therefore, the distance of point P from the x-axis is 3 units.

C Code- distance

```c
#include <math.h>
void calculate_distance_from_xaxis(
double* input_P, // Pointer to a 2-element
    array [Px, Py]
double* output_distance // Pointer to a 1-
    element array to be filled
) {
        // Unpack the y-coordinate from the
            input point
        double Py = input_P[1];

        // The distance is the absolute
            value of the y-coordinate
        double distance = fabs(Py);

        // Fill the output array with the
            calculated distance
        output_distance[0] = distance;
}
```

Python Code using shared output

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

# --- Step 1: Load the shared library ---
lib = ctypes.CDLL('./4.7.39.so')

# --- Step 2: Define the C function signature
    using NumPy-aware pointers ---
calculate_distance = lib.
    calculate_distance_from_xaxis

# Define the argument types
calculate_distance.argtypes = [
np.ctypeslib.ndpointer(dtype=np.double, ndim=1,
    flags='C_CONTIGUOUS'), # input_P
np.ctypeslib.ndpointer(dtype=np.double, ndim=1,
    flags='C_CONTIGUOUS') # output_distance
]
```

Python Code using shared output

```
    # --- Step 3: Prepare NumPy arrays and call the C function ---
    # Define the point P as a NumPy array
    P = np.array([2.0, 3.0], dtype=np.double)

    # Create an empty NumPy array for the C function to fill
    output_data = np.zeros(1, dtype=np.double)

    # Call the C function. NumPy arrays are passed directly.
    calculate_distance(P, output_data)

    # --- Step 4: Extract the result and plot ---
    # The calculated distance is the first (and only) element in the output array
    distance = output_data[0]
    print(fPoint P Coordinates: ({P[0]}, {P[1]}))
    print(fDistance from x-axis (calculated by C):{
```

Python Code using shared output

```
# Setup for plotting
fig, ax = plt.subplots(figsize=(8, 8))
ax.set_aspect('equal', adjustable='box')
ax.grid(True, linestyle=':', alpha=0.7)

# The projection of P onto the x-axis is Q
Q = np.array([P[0], 0.0])

# Plot the distance line between P and Q
ax.plot([P[0], Q[0]], [P[1], Q[1]], 'g--',
    label=f'Distance = {distance:.2f}')

# Plot point P and its projection Q
ax.plot(P[0], P[1], 'o', markersize=10,
    color='red', label=f'Point P({P[0]}, {P
    [1]})')
ax.text(P[0] + 0.1, P[1] + 0.1, 'P',
    fontsize=14, fontweight='bold', color='
    red')
```

Python Code using shared output

```python
# Axes and Title
ax.axhline(0, color='black', linewidth=0.8)
ax.axvline(0, color='black', linewidth=0.8)
ax.set_xlim(-1, 5)
ax.set_ylim(-1, 5)
ax.set_title('Distance of Point P from the
    x-axis', fontsize=16)
ax.legend(loc=upper left)

# Save the figure to be used in the LaTeX
    document
plt.savefig('distance_plot.png')

plt.show()
```

Plot by python using shared output from c

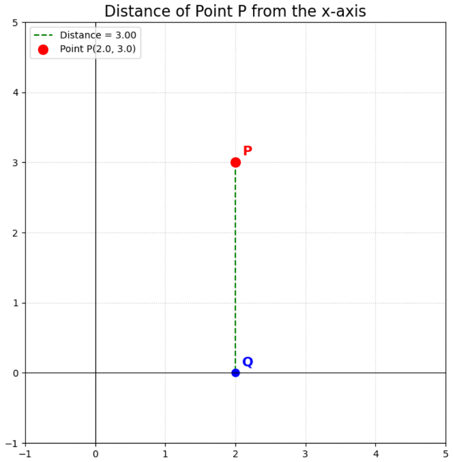

Figure: *

Plot by python only

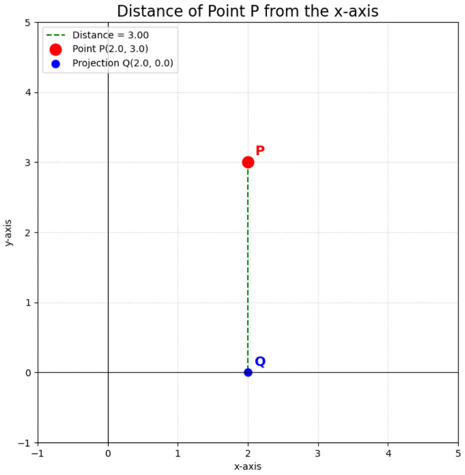

Figure: *