1.9.22

Kavin B-EE25BTECH11033

August 25,2025

Question

Find the value of y for which the distance between the points $\mathbf{P}(2, -3)$ and $\mathbf{Q}(10, y)$ is 10 units.

Theoretical Solution

Given the points,

$$\mathbf{P} = \begin{pmatrix} 2 \\ -3 \end{pmatrix} \quad \mathbf{Q} = \begin{pmatrix} 10 \\ y \end{pmatrix} \tag{1}$$

The distance between the points P and Q is given as,

$$d = \|\mathbf{P} - \mathbf{Q}\| = 10 \tag{2}$$

Formulae

The length of a vector is defined as

$$\|\mathbf{P}-\mathbf{Q}\| \triangleq \sqrt{(\mathbf{P}-\mathbf{Q})^{\top}(\mathbf{P}-\mathbf{Q})} \tag{3}$$

Theoretical Solution

$$(\mathbf{P}-\mathbf{Q})^\top(\mathbf{P}-\mathbf{Q}) = \|\mathbf{P}-\mathbf{Q}\|^2 \tag{4}$$

$$(\mathbf{P}-\mathbf{Q})^\top(\mathbf{P}-\mathbf{Q}) = 10^2 \tag{5}$$

$$\because \mathbf{P}-\mathbf{Q} = \begin{pmatrix} 2 \\ -3 \end{pmatrix} - \begin{pmatrix} 10 \\ y \end{pmatrix} = \begin{pmatrix} -8 \\ -3-y \end{pmatrix}, \tag{6}$$

$$\implies \begin{pmatrix} -8 \\ -3-y \end{pmatrix}^\top \begin{pmatrix} -8 \\ -3-y \end{pmatrix} = 10^2 \tag{7}$$

Theoretical Solution

$$\implies \begin{pmatrix} -8 & -3-y \end{pmatrix} \begin{pmatrix} -8 \\ -3-y \end{pmatrix} = 100 \tag{8}$$

$$\implies 8^2 + (3+y)^2 = 100 \tag{9}$$

$$\implies (3+y)^2 = 36 \tag{10}$$

$$\implies 3+y = \pm 6 \tag{11}$$

$$\implies y = 3, -9 \tag{12}$$

Therefore the points $(10, 3)$ and $(10, -9)$ are at a distance of 10 units

from the point P.

C Code - A function to find the y coordinates of Q

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

int findYCoordinates(double px, double py, double qx, double d,
    double *y1, double *y2) {
    double term = pow(d, 2) - pow(qx - px, 2);
    double sqrt_term = sqrt(term);
    *y1 = py + sqrt_term;
    *y2 = py - sqrt_term;

    return 1;
}
```

Python Code

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

c_lib=ctypes.CDLL('./code.so')

# --- 2. Define the C Function Signature in Python ---
# Get a handle to the C function
find_y_coordinates = c_lib.findYCoordinates
```

Python Code

```
# Define the argument types (argtypes) for the C function
# double, double, double, double, *double, *double
find_y_coordinates.argtypes = [
    ctypes.c_double, ctypes.c_double,
    ctypes.c_double, ctypes.c_double,
    ctypes.POINTER(ctypes.c_double),
    ctypes.POINTER(ctypes.c_double)
]
# Define the return type (restype)
find_y_coordinates.restype = ctypes.c_int
```

Python Code

```python
# --- Prepare Inputs and Call the C Function ---
# Problem parameters
px, py = 2.0, -3.0
qx = 10.0
distance = 10.0

# Create C-compatible variables to hold the results (y1 and y2)
y1_c = ctypes.c_double()
y2_c = ctypes.c_double()

# Call the C function. Use ctypes.byref() to pass the variables
    by reference.
success = find_y_coordinates(px, py, qx, distance, ctypes.byref(
    y1_c), ctypes.byref(y2_c))
```

Python Code

```python
if not success:
    print("C function failed to find real coordinates. Check your
        inputs.")
    exit()

# Extract the Python values from the C-type objects
y1 = y1_c.value
y2 = y2_c.value

print(f"Values calculated by C function: y1 = {y1}, y2 = {y2}")
# --- Plot the Results ---
# Define the points using the values from the C function
P = np.array([px, py]).reshape(-1, 1)
Q1 = np.array([qx, y1]).reshape(-1, 1) # Use y1 from C
Q2 = np.array([qx, y2]).reshape(-1, 1) # Use y2 from C
```

Python Code

```
# Plotting the lines from P to Q1 and P to Q2
plt.plot([P[0,0], Q1[0,0]], [P[1,0], Q1[1,0]], label=f'$PQ_1$ (
    distance={distance})')
plt.plot([P[0,0], Q2[0,0]], [P[1,0], Q2[1,0]], label=f'$PQ_2$ (
    distance={distance})')

# Combining all points for easy plotting and labeling
coords = np.block([[P, Q1, Q2]])
plt.scatter(coords[0, :], coords[1, :], color='red', zorder=5)
# Adding labels for each point
vert_labels = ['P', 'Q', 'Q']
for i, txt in enumerate(vert_labels):
    plt.annotate(f'{txt}\n({coords[0, i]:.0f}, {coords[1, i]:.0f
        })',
                 (coords[0, i], coords[1, i]),
                 textcoords="offset points",
                 xytext=(0, 10),
                 ha='center')
```

Python Code

```
# --- Plot Formatting ---
ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['left'].set_position('zero')
ax.spines['right'].set_color('none')
ax.spines['bottom'].set_position('zero')

plt.legend(loc='best')
plt.grid(True)
plt.axis('equal')
plt.title("Plot generated using values from C function")
# Save the plot to a file
plt.savefig('../figs/fig.png')


plt.show()
```

Plot