1.8.24

Aniket-EE25BTECH11007

October 3, 2025

Question

If (a, b) is the mid-point of the line segment joining the point **A** $(10, -6)$ and **B** $(k, 4)$ and $a - 2b = 18$, find the value of a, b and the distance **AB**.

Theoretical Solution

Let $\mathbf{A} = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$ and $\mathbf{B} = \begin{pmatrix} x_2 \\ y_2 \end{pmatrix}$. By the (matrix) section formula, the point dividing $\mathbf{AB}$ in the ratio $k : 1$ is

$$R_{\text{int}} = \frac{1}{k+1} [\,\mathbf{A}\ \mathbf{B}\,] \begin{pmatrix} 1 \\ k \end{pmatrix}. \tag{1}$$

With $\mathbf{A} = \begin{pmatrix} 10 \\ -6 \end{pmatrix}$ and $\mathbf{B} = \begin{pmatrix} k \\ 4 \end{pmatrix}$ and $\mathbf{O} = \begin{pmatrix} a \\ b \end{pmatrix}$ the midpoint $(k = 1)$ is

$$\mathbf{O} = \frac{1}{2} [\,\mathbf{A}\ \mathbf{B}\,] \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 10 & k \\ -6 & 4 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{10+k}{2} \\ -1 \end{pmatrix}. \tag{2}$$

Theoretical Solution

Thus,

$$a = \frac{10+k}{2}, \quad b = -1 \tag{3}$$

Using the given condition $a - 2b = 18$:

$$\frac{10+k}{2} - 2(-1) = 18 \tag{4}$$

$$k = 22 \tag{5}$$

So,

$$a = \boxed{16}, \quad b = \boxed{-1} \tag{6}$$

Theoretical Solution

Distance Between AB:

$$\|\mathbf{A}-\mathbf{B}\| = \sqrt{(\mathbf{A}-\mathbf{B})^\top(\mathbf{A}-\mathbf{B})}. \tag{7}$$

Given,

$$\mathbf{A} = \begin{pmatrix} 10 \\ -6 \end{pmatrix}, \qquad \mathbf{B} = \begin{pmatrix} 22 \\ 4 \end{pmatrix}. \tag{8}$$

$$\|\mathbf{A}-\mathbf{B}\| = \sqrt{(-12)^2 + (-10)^2} \tag{9}$$

$$= \boxed{2\sqrt{61}} \tag{10}$$

C Code - Finding Midpoint and Distance between 2 points

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

// Solve the midpoint + constraint problem
// A = (Ax, Ay)
// B = (k, By) (Bx is unknown, denoted k)
// Constraint: p*a + q*b = r, where (a,b) is midpoint
//
// Outputs:
// *a, *b  midpoint coordinates
// *k  solved x-coordinate of B
// *ABx,*ABy  vector A-B
// *norm2  squared distance (A-B).(A-B)
void solve_problem(double Ax, double Ay, double By,
                   double p, double q, double r,
                   double *a, double *b, double *k,
                   double *ABx, double *ABy, double *norm2)
```

C Code - Finding Midpoint and Distance between 2 points

```
{
    // Midpoint coordinates
    *b = (Ay + By) / 2.0;
    *k = (2.0 * r - p * Ax - 2.0 * q * (*b)) / p;
    *a = (Ax + *k) / 2.0;

    // Vector A - B
    *ABx = Ax - *k;
    *ABy = Ay - By;

    // Squared distance
    *norm2 = (*ABx) * (*ABx) + (*ABy) * (*ABy);
}
```

Python+C Code

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

# Load shared object
lib = ctypes.CDLL("./mg2.so")

# Define argument and return types
lib.solve_problem.argtypes = [
    ctypes.c_double, ctypes.c_double, ctypes.c_double, # Ax, Ay,
        By
    ctypes.c_double, ctypes.c_double, ctypes.c_double, # p, q, r
    ctypes.POINTER(ctypes.c_double), ctypes.POINTER(ctypes.
        c_double), ctypes.POINTER(ctypes.c_double),
    ctypes.POINTER(ctypes.c_double), ctypes.POINTER(ctypes.
        c_double), ctypes.POINTER(ctypes.c_double)
]
```

Python+C Code

```
lib.solve_problem.restype = None

# Inputs
Ax, Ay = 10.0, -6.0
By = 4.0
p, q, r = 1.0, -2.0, 18.0 # constraint: a - 2b = 18

# Outputs (ctypes)
a = ctypes.c_double()
b = ctypes.c_double()
k = ctypes.c_double()
ABx = ctypes.c_double()
ABy = ctypes.c_double()
norm2 = ctypes.c_double()
```

Python+C Code

```python
# Call C function
lib.solve_problem(
    Ax, Ay, By, p, q, r,
    ctypes.byref(a), ctypes.byref(b), ctypes.byref(k),
    ctypes.byref(ABx), ctypes.byref(ABy), ctypes.byref(norm2)
)

# Convert results into NumPy arrays
A = np.array([Ax, Ay])
B = np.array([k.value, By])
O = np.array([a.value, b.value])

print("A =", A)
print("B =", B)
print("O (midpoint) =", O)
print("Vector A-B =", A - B)
print("Squared distance =", norm2.value)
```

Python+C Code

```

# Plotting
plt.figure()
plt.plot([A[0], B[0]], [A[1], B[1]], 'k-') # segment AB
plt.scatter(*A, color='red', label='A')
plt.scatter(*B, color='blue', label='B')
plt.scatter(*O, color='green', label='Midpoint O')

plt.text(A[0]+0.5, A[1], f"A{tuple(A.astype(int))}")
plt.text(B[0]+0.5, B[1], f"B{tuple(B.astype(int))}")
plt.text(O[0]+0.5, O[1], f"O{tuple(O.astype(int))}")

plt.gca().set_aspect('equal', adjustable='box')
plt.legend()
plt.title("Segment AB and Midpoint O")
plt.show()
```

Python Code

```

   import numpy as np
import matplotlib.pyplot as plt

# Given data
A = np.array([10, -6])
B = np.array([22, 4])
O = (A + B) // 2 # midpoint (integer division gives exact
    midpoint here)

# Convert to Python tuples for clean display
A_t = tuple(A.tolist())
B_t = tuple(B.tolist())
O_t = tuple(O.tolist())
```

Python Code

```python

# Print results
print("A =", A_t)
print("B =", B_t)
print("O =", O_t)

# Plotting
plt.figure()
plt.plot([A[0], B[0]], [A[1], B[1]], 'k-', label="Segment AB") #
    line AB
plt.scatter(*A, color='red', label=f"A{A_t}")
plt.scatter(*B, color='blue', label=f"B{B_t}")
plt.scatter(*O, color='green', label=f"O{O_t}")
```

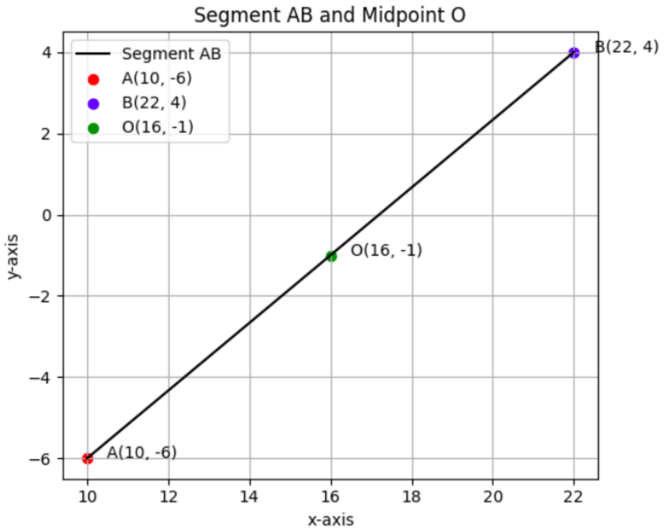
Python Code

```python

# Annotate points with clean integer tuples
plt.text(A[0]+0.5, A[1], f"A{A_t}")
plt.text(B[0]+0.5, B[1], f"B{B_t}")
plt.text(O[0]+0.5, O[1], f"O{O_t}")

plt.gca().set_aspect('equal', adjustable='box')
plt.legend()
plt.title("Segment AB and Midpoint O")
plt.xlabel("x-axis")
plt.ylabel("y-axis")
plt.grid(True)
plt.show()
```

Plot


Segment AB and Midpoint O
Segment AB
A(10, -6)
B(22, 4)
O(16, -1)
B(22, 4)
O(16, -1)
A(10, -6)
y-axis
x-axis
4
2
0
−2
−4
−6
10
12
14
16
18
20
22