

1.5.36

Sai Krishna Bakki - EE25BTECH11049

Question

Point $P(x, 4)$ lies on the line segment joining the points $\mathbf{A}(-5, 8)$ and $\mathbf{B}(4, -10)$. Find the ratio in which point P divides the line segment AB . Also, find the value of x .

Theoretical Solution

Let

$$\mathbf{A} = \begin{pmatrix} -5 \\ 8 \end{pmatrix}, \mathbf{B} = \begin{pmatrix} 4 \\ -10 \end{pmatrix}, \mathbf{P} = \begin{pmatrix} x \\ 4 \end{pmatrix} \quad (1)$$

Since \mathbf{P} lies on \mathbf{A} and \mathbf{B} , they must be collinear

$$\therefore \text{rank} \begin{pmatrix} \mathbf{P} - \mathbf{A} & \mathbf{B} - \mathbf{A} \end{pmatrix} = 1 \quad (2)$$

$$\text{rank} \begin{pmatrix} x + 5 & 9 \\ -4 & -18 \end{pmatrix} = 1 \quad (3)$$

By transformation $R_2 \rightarrow R_2 + 2R_1$

$$\text{rank} \begin{pmatrix} x + 5 & 9 \\ 2x + 6 & 0 \end{pmatrix} = 1 \quad (4)$$

Thus for rank to be 1

Theoretical Solution

$$2x + 6 = 0 \quad (5)$$

$$\therefore x = -3 \quad (6)$$

Thus **P** is :

$$\mathbf{P} = \begin{pmatrix} -3 \\ 4 \end{pmatrix} \quad (7)$$

let **P** divide the line joining points **A** and **B** in the ratio $k : 1$.

$$\mathbf{P} = \frac{k\mathbf{B} + \mathbf{A}}{k + 1}$$

$$k(\mathbf{P} - \mathbf{B}) = \mathbf{A} - \mathbf{P} \quad k = \frac{(\mathbf{P} - \mathbf{B})^T (\mathbf{A} - \mathbf{P})}{\|(\mathbf{P} - \mathbf{B})\|^2}$$

Theoretical Solution

$$k = \frac{\begin{pmatrix} x-4 \\ -14 \end{pmatrix} \cdot \begin{pmatrix} -5-x \\ 4 \end{pmatrix}}{\left\| \begin{pmatrix} x-4 \\ -14 \end{pmatrix} \right\|^2}$$

substituting the value of x as , we get the value of k as

$$k = 2/7$$

C Code

```
#include <stdio.h>

/*
 * Compute AP and PB (vertical distances) and x-coordinate of P (
   using section formula).
 *
 * Inputs:
 * Ax, Ay, Bx, By, yP
 *
 * Outputs (via pointers):
 * *ratio_AP : AP (vertical distance Ay - yP)
 * *ratio_PB : PB (vertical distance yP - By)
 * *xP : x-coordinate of P computed by section formula
 */
void section_point(double Ax, double Ay, double Bx, double By,
    double yP,
    double *ratio_AP, double *ratio_PB, double *xP)
{
```

C Code

```
double m = Ay - yP; /* vertical distance AP */
double n = yP - By; /* vertical distance PB */

if (m + n == 0.0) {
    /* degenerate: cannot determine location */
    fprintf(stderr, "Error: m + n == 0, cannot compute section
    .\n");
    if (ratio_AP) *ratio_AP = 0.0;
    if (ratio_PB) *ratio_PB = 0.0;
    if (xP) *xP = 0.0;
    return;
}

if (ratio_AP) *ratio_AP = m;
if (ratio_PB) *ratio_PB = n;

/* Section formula for internal division:
   
$$x = \frac{n \cdot Ax + m \cdot Bx}{m + n}$$

```

(because $AP:PB = m:n$, weight on A is n, on B is m)

*/

if (xP) *xP = (n*Ax + m*Bx) / (m + n);

}

Python Code through shared output

```
#!/usr/bin/env python3
import os
import subprocess
import ctypes
import math
import matplotlib.pyplot as plt

C_FILE = section.c
SO_FILE = ./libsection.so

# Auto-compile if shared lib not present
if not os.path.exists(SO_FILE):
    print(libsection.so not found compiling section.c ...)
    cmd = [gcc, -shared, -o, libsection.so, -fPIC, C_FILE]
    try:
        subprocess.run(cmd, check=True)
        print(Compiled libsection.so)
```

Python Code through shared output

```
except subprocess.CalledProcessError as e:
    print(Compilation failed:, e)
    raise SystemExit(1)

# Load shared library
lib = ctypes.CDLL(SO_FILE)

# Set function signature:
# void section_point(double, double, double, double, double,
# double*, double*, double*)
lib.section_point.argtypes = [ctypes.c_double, ctypes.c_double,
                               ctypes.c_double, ctypes.c_double,
                               ctypes.c_double,
                               ctypes.POINTER(ctypes.c_double),
                               ctypes.POINTER(ctypes.c_double),
                               ctypes.POINTER(ctypes.c_double)]
lib.section_point.restype = None
```

Python Code through shared output

```
# Input points
Ax, Ay = -5.0, 8.0
Bx, By = 4.0, -10.0
yP = 4.0

# Prepare output holders
ratio_AP = ctypes.c_double()
ratio_PB = ctypes.c_double()
xP = ctypes.c_double()

# Call C function
lib.section_point(Ax, Ay, Bx, By, yP,
                  ctypes.byref(ratio_AP),
                  ctypes.byref(ratio_PB),
                  ctypes.byref(xP))

# Read outputs
m = ratio_AP.value # AP vertical distance (Ay - yP)
n = ratio_PB.value # PB vertical distance (yP - By)
```

Python Code through shared output

```
x_val = xP.value

# Convert ratio to smallest integer ratio (if sensible)
# We'll round to nearest integer then reduce by gcd if both
  nonzero
def reduced_ratio(a, b):
    ia = int(round(a))
    ib = int(round(b))
    if ia == 0 and ib == 0:
        return (0, 0)
    if ia < 0: ia = -ia
    if ib < 0: ib = -ib
    g = math.gcd(ia, ib) if (ia != 0 and ib != 0) else (ia or ib)
    if g == 0:
        return (ia, ib)
    return (ia // g, ib // g)

ratA_int, ratB_int = reduced_ratio(m, n)
```

Python Code through shared output

```
print(f Raw m (AP) = {m}, n (PB) = {n})
print(f AP:PB (reduced) = {ratA_int}:{ratB_int})
print(f x = {x_val})

# ---- Plot ----
A = (Ax, Ay)
B = (Bx, By)
P = (x_val, yP)

plt.plot([A[0], B[0]], [A[1], B[1]], linestyle='--', label=Line
AB)
plt.scatter(*A, marker='o', label=fA{A}, zorder=5)
plt.scatter(*B, marker='o', label=fB{B}, zorder=5)
plt.scatter(*P, marker='o', label=fP({x_val:.3g},{yP}), zorder=5)

plt.text(A[0]-0.6, A[1]+0.4, fA{A}, color=red)
plt.text(B[0]+0.4, B[1]-0.6, fB{B}, color=blue)
```

Python Code through shared output

```
plt.text(P[0]+0.4, P[1]+0.4, fP({x_val:.3g},{yP}), color=green)

plt.axhline(0, color=gray, lw=0.5)
plt.axvline(0, color=gray, lw=0.5)
plt.grid(True, linestyle=--, alpha=0.5)
plt.xlabel(x)
plt.ylabel(y)
plt.title(fP divides AB in ratio {ratA_int}:{ratB_int})
plt.legend()
plt.gca().set_aspect('equal', adjustable='box')
plt.show()
```

Python Code

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

# Local imports (your setup)
from libs.line.funcs import *

# Section formula
def section_point(A, B, m, n):
    return (m*B + n*A) / (m+n)

# Given points
A = np.array([-5, 8]).reshape(-1, 1)
B = np.array([4, -10]).reshape(-1, 1)

# Given y of P
yP = 4
```

Python Code

```
# Solve for ratio m:n using y-coordinate
#  $4(m+n) = m*y_B + n*y_A$ 
#  $\Rightarrow 4m + 4n = m*(-10) + n*8$ 
#  $\Rightarrow 14m = 4n \Rightarrow m/n = 2/7$ 
m, n = 2, 7

# Compute P using section formula
P = section_point(A, B, m, n).reshape(-1, 1)

print(fRatio AP:PB = {m}:{n})
print(fValue of x = {P[0,0]})

# Plot
x_AB = line_gen_num(A, B, 20)
plt.plot(x_AB[0,:], x_AB[1:], 'g--', label=Line Segment AB)

plot_coords = np.block([[A, B, P]])
plt.scatter(plot_coords[0,:], plot_coords[1:], color='blue')
```

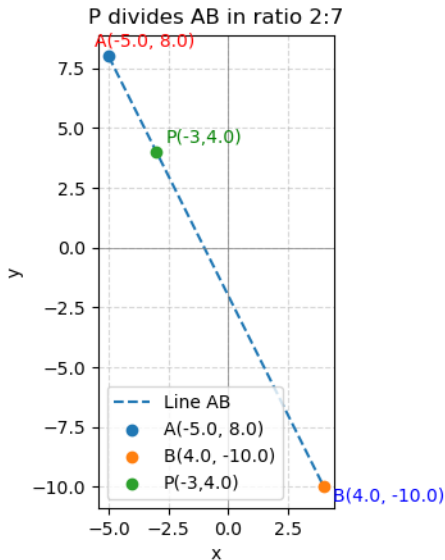

Python Code

```
vert_labels = [  
    f'A({A[0,0]}, {A[1,0]})',  
    f'B({B[0,0]}, {B[1,0]})',  
    f'P({P[0,0]}, {P[1,0]})'  
]  
  
for i, txt in enumerate(vert_labels):  
    plt.annotate(txt,  
        (plot_coords[0,i], plot_coords[1,i]),  
        textcoords=offset points,  
        xytext=(0,10),  
        ha='center')  
  
plt.xlabel('$x$')  
plt.ylabel('$y$')
```

```
plt.title(Division of Line Segment AB by Point P)
plt.legend(loc='best')
plt.grid()
plt.axis('equal')

plt.savefig(../figs/section_formula_plot.jpg)
plt.show()
```

Plot by python using shared output from c



Plot by using Python only

