

# Problem 1.5.11

Venkata Sai

August 27, 2025

## 1 Problem

## 2 Solution

- Section Formula
- Obtaining  $k$  Value
- Obtaining Point
- Plot

## 3 C Code

## 4 Python Code

# Problem Statement

The point **R** divides the line segment AB, where **A**(-4, 0) and **B**(0, 6) such that  $AR = \frac{3}{4}AB$ . Find the coordinates of **R**.

Variable	Description	Values
A	Point	(-5, 0)
B	Point	(0, 6)

Table: Variables given

## Section Formula

Formula:

$$\mathbf{P} = \frac{k(\mathbf{B}) + (\mathbf{A})}{k + 1} \quad (3.1)$$

Where:

'k' is the ratio in which the point divides the line segment

$$\mathbf{A} = \begin{pmatrix} -4 \\ 0 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 0 \\ 6 \end{pmatrix} \quad (3.2)$$

## Obtaining $k$ Value

$$AR = \frac{3}{4}AB \quad (3.3)$$

$$AR = \frac{3}{4}(AR + RB) \quad (3.4)$$

$$4AR = 3AR + 3RB \quad (3.5)$$

$$AR = 3RB \quad (3.6)$$

$$\frac{AR}{RB} = 3 \quad (3.7)$$

Hence  $k=3$

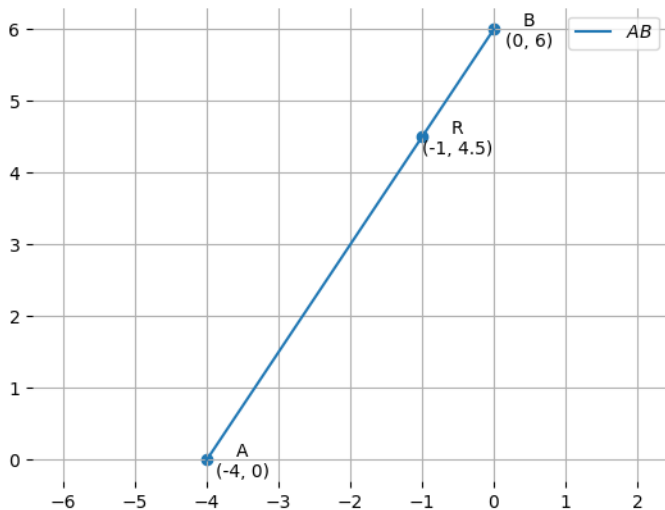
## Obtaining Point

$$\mathbf{P} = \frac{3\mathbf{B} + \mathbf{A}}{4} = \frac{3\begin{pmatrix} 0 \\ 6 \end{pmatrix} + \begin{pmatrix} -4 \\ 0 \end{pmatrix}}{4} = \frac{\begin{pmatrix} -4 \\ 18 \end{pmatrix}}{4} \quad (3.8)$$

$$\mathbf{P} = \begin{pmatrix} -1 \\ \frac{9}{2} \end{pmatrix} \quad (3.9)$$

Hence the coordinates of  $\mathbf{P}$  are  $(-1, \frac{9}{2})$

# Plot



## C Code for generating points on line

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/socket.h>
#include <netinet/in.h>

#include "libs/matfun.h"
#include "libs/geofun.h"

int main() {
    double **k, **M, **C;
    int x1 = -4, x2 = 0, y1 = 0, y2 = 6;

    // Create matrices
    M = createMat(2, 2);
```



## C Code for generating points on line

```
k = createMat(2, 1);
C = createMat(2, 1);

M[0][1] = x1; // x1 = -4
M[1][1] = y1; // y1 = 0

M[0][0] = x2; // x2 = 0
M[1][0] = y2; // y2 = 6

k[0][0] = 3.0 / 4; // weight for B (column 0)
k[1][0] = 1.0 / 4; // weight for A (column 1)

C = Matmul(M, k, 2, 2, 1);

// Write result to file
FILE *file = fopen("values.dat", "w");
if (file == NULL) {
```

## C Code for generating points on line

```
    printf("Error opening file!\n");  
    return 1;  
}  
  
fprintf(file, "x\ty\t of C\n");  
fprintf(file, "%.02lf\t%.02lf\n", C[0][0], C[1][0]); // x and  
    y of C  
  
fclose(file);  
printf("Results have been written to values.dat\n");  
  
// Free memory  
freeMat(M, 2);  
freeMat(k, 2);  
freeMat(C, 2);  
  
return 0;  
}
```

# Python Code for Plotting

```
# Code by /sdcard/github/matgeo/codes/CoordGeoVV Sharma  
# September 12, 2023  
# Revised July 21, 2024  
# Released under GNU GPL  
# Section Formula  
  
import sys  
sys.path.insert(0, '/workspaces/urban-potato/matgeo/codes/  
    CoordGeo/') # path to my scripts  
  
import numpy as np  
import matplotlib.pyplot as plt  
import matplotlib.image as mpimg  
# Local imports  
from line.funcs import *  
from triangle.funcs import *  
from conics.funcs import circ_gen
```

# Python Code for Plotting

```
# Read data
data = np.loadtxt("values.dat", skiprows=1)

xc = data[0] # Extract x-coordinate (e.g., -1)
yc = data[1] # Extract y-coordinate (e.g., 4.5)

# Given points
A = np.array([-4, 0]).reshape(-1, 1)
B = np.array([0, 6]).reshape(-1, 1)
R = np.array([xc, yc]).reshape(-1, 1)

# Generating line AB
x_AB = line_gen(A, B)

# Plotting
plt.plot(x_AB[0, :], x_AB[1, :], label='$AB$')
```

# Python Code for Plotting

```
# Labeling the coordinates
tri_coords = np.block([[A, B, R]])
plt.scatter(tri_coords[0, :], tri_coords[1, :])

vert_labels = ['A', 'B', 'R']

# Helper function: format number with decimal only if needed
def fmt(val):
    return f"{val:.1f}" if abs(val - round(val)) > 1e-6 else f"{int(val)}"

for i, txt in enumerate(vert_labels):
    x = tri_coords[0, i]
    y = tri_coords[1, i]
    plt.annotate(f'{txt}\n({fmt(x)}, {fmt(y)})',
                (x, y),
                textcoords="offset points",
```

# Python Code for Plotting

```
xytext=(20, -10),  
ha='center')
```

```
# Styling
```

```
ax = plt.gca()  
ax.spines['left'].set_visible(False)  
ax.spines['right'].set_visible(False)  
ax.spines['top'].set_visible(False)  
ax.spines['bottom'].set_visible(False)
```

```
plt.legend(loc='best')  
plt.grid()  
plt.axis('equal')
```

```
plt.savefig('../figs/fig1.png')  
plt.show()
```