Problem 1.9.12

ee25btech11023-Venkata Sai

August 26, 2025

Problem Statement

Find the length of the segment joining $\mathbf{A}(-6, 7)$ and $\mathbf{B}(-1, -5)$. Also, find the midpoint of AB.

Variable	**Description**	**Values**
A	Point	$(-6, 7)$
B	Point	$(-1, -5)$
P	Midpoint of AB	(x,y)

Table: Variables given

Formulas

Formula:
Let (x_1, y_1) and (x_2, y_2) be any two points. Distance Formula is given by

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{3.1}$$

Midpoint Formula:

$$\mathbf{P} = \frac{k(\mathbf{B}) + (\mathbf{A})}{k + 1} \tag{3.2}$$

Where:

$$k = 1$$

$$\mathbf{A} = \begin{pmatrix} -6 \\ 7 \end{pmatrix} \qquad \mathbf{B} = \begin{pmatrix} -1 \\ -5 \end{pmatrix} \tag{3.3}$$

Obtaining Distance

$$\begin{pmatrix} -6 \\ 7 \end{pmatrix} = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} \tag{3.4}$$

$$\begin{pmatrix} -1 \\ -5 \end{pmatrix} = \begin{pmatrix} x_2 \\ y_2 \end{pmatrix} \tag{3.5}$$

$$\text{Distance} = \sqrt{(-1-(-6))^2 + (-5-7)^2} = \sqrt{(5)^2 + (-12)^2} \tag{3.6}$$

$$= \sqrt{25 + 144} \tag{3.7}$$

$$= \sqrt{169} \tag{3.8}$$

$$= 13 \tag{3.9}$$

Finding Midpoint

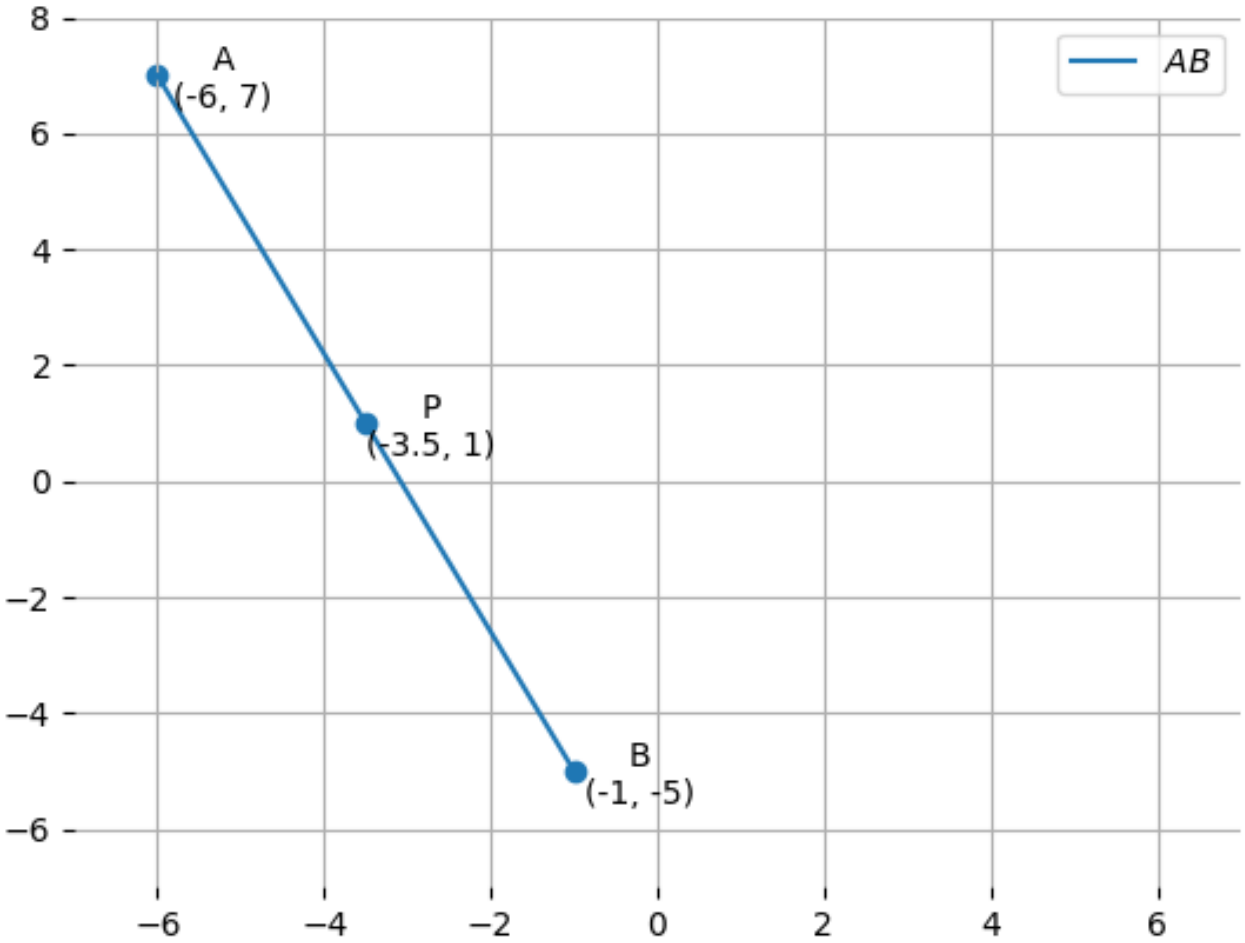
$$P = \frac{B + A}{2} = \frac{\begin{pmatrix} -6 \\ 7 \end{pmatrix} + \begin{pmatrix} -1 \\ -5 \end{pmatrix}}{2} = \frac{\begin{pmatrix} -7 \\ 2 \end{pmatrix}}{2} \tag{3.10}$$

$$= \begin{pmatrix} \frac{-7}{2} \\ 1 \end{pmatrix} \tag{3.11}$$

$$\tag{3.12}$$

Hence the coordinates of **P** are $\left(\frac{-7}{2}, 1\right)$

Plot



Figure

C Code for generating points on line

```c
#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 = -6, x2 = -1, y1 = 7, y2 = -5;

    // Create matrices
    M = createMat(2, 2);
    k = createMat(2, 1);
    C = createMat(2, 1);
```

C Code for generating points on line

```
M[0][1] = x1;
  M[1][1] = y1;


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


  k[0][0] =1.0 / 2; // weight for B (column 0)
  k[1][0] = 1.0 / 2; // weight for A (column 1)

  // Matrix multiplication: C = M * k
  C = Matmul(M, k, 2, 2, 1);

  // Write result to file
  FILE *file = fopen("values.dat", "w");
  if (file == NULL) {
      printf("Error opening file!\n");
      return 1;
```

C Code for generating points on line

```
    }
     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([-6, 7]).reshape(-1, 1)
B = np.array([-1, -5]).reshape(-1, 1)
P = 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, P]])
plt.scatter(tri_coords[0, :], tri_coords[1, :])

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

# 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].item()
    y = tri_coords[1, i].item()
    plt.annotate(f'{txt}\n({fmt(x)}, {fmt(y)})',
                 (x, y),
                 textcoords="offset points",
                 xytext=(20, -10),
                 ha='center')
```

Python Code for Plotting

```
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()

# Increase y-axis from -8 to 8 to show full range
plt.ylim(-7, 8)
plt.xlim(-7,7)

# Save and open
plt.show()
plt.savefig('../figs/fig1.png')
```