Presentation Template

B. Sai Likith Reddy
Dept. of Artificial Intelligence,
IIT Hyderabad.

November 12, 2024

1. Problem

2. Solution
   - section formula in matrix form
   - solving for k
   - solving for y

3. codes
   - C code to calculate points

4. codes
   - Python code to Plot

Problem Statement

P $(5, -3)$ and **Q** $(3, y)$ are the points of trisection of the line segment joining **A** $(7, -2)$ and **B** $(1, -5)$. Theny equals

Section formula

If a point **R** divides the line segment joining the points **A** and **B** in the ratio $k : 1$ then the point **R** can be found by using the section formula below

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

Now let **P** divides the Line segment in the ratio $1 : k$ And then we will find k if we know k then we will know how **Q** divides the line segment and hence we can find the value of y

solving for k

$$\mathbf{P} = \frac{k\mathbf{A} + \mathbf{B}}{k+1} \tag{3.1}$$

$$\begin{pmatrix} 5 \\ -3 \end{pmatrix} = \frac{k\begin{pmatrix} 7 \\ -2 \end{pmatrix} + \begin{pmatrix} 1 \\ -5 \end{pmatrix}}{1+k} \tag{3.2}$$

solving x coordinate

$$5\,(1+k) = 7k+1 \tag{3.3}$$

hence $k = 2$

solving for y

As the value of $k = 2$
Q divides **AB** in the ratio $2:1$

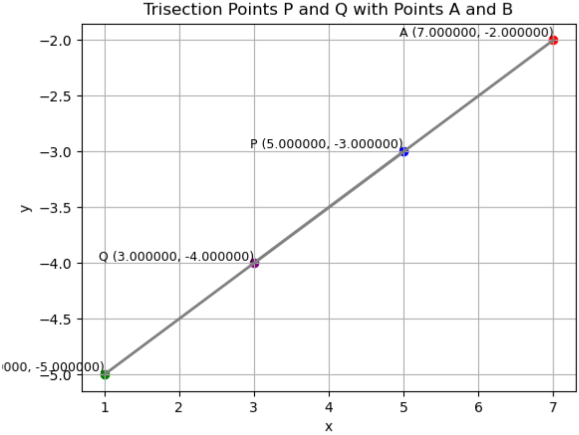
$$\begin{pmatrix} 3 \\ y \end{pmatrix} = \frac{\begin{pmatrix} 1 \\ -5 \end{pmatrix} + \frac{1}{2}\begin{pmatrix} 7 \\ -2 \end{pmatrix}}{1 + \frac{1}{2}} \tag{3.4}$$

solving $y-$ coordinate

$$\frac{3}{2}y = -5 - 2\left(\frac{1}{2}\right) \tag{3.5}$$

Therefore $y = -4$

Plot of the Points



generating points and the line I

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "libs/matfun.h"
#include "libs/geofun.h"  // Include geofun.h for geometric operations

// Function to calculate trisection points P and Q
void calculate_trisection(double **A, double **B, double **Q, double **P) {
    // Trisection formulas
    Q[0][0] = (2 * B[0][0] + A[0][0]) / 3;  // P (1:2)
    Q[1][0] = (2 * B[1][0] + A[1][0]) / 3;

    P[0][0] = (B[0][0] + 2 * A[0][0]) / 3;  // Q (2:1)
    P[1][0] = (B[1][0] + 2 * A[1][0]) / 3;
}

// Function to generate points of trisection and write them to a file
void point_gen(double **A, double **B, const char *filename) {
    FILE *file = fopen(filename, "w");
```

generating points and the line II

```c
    if (file == NULL) {
        printf("Error opening file.\n");
        return;
    }

    // Allocate memory for trisection points
    double **Q = createMat(2, 1);
    double **P = createMat(2, 1);

    // Calculate trisection points
    calculate_trisection(A, B, Q, P);

    // Write the points to the file
    fprintf(file, "Point A: (%lf, %lf)\n", A[0][0], A[1][0]);
    fprintf(file, "Point B: (%lf, %lf)\n", B[0][0], B[1][0]);
    fprintf(file, "Trisection Point P (1:2): (%lf, %lf)\n", P[0][0], P[1][0]);
    fprintf(file, "Trisection Point Q (2:1): (%lf, %lf)\n", Q[0][0], Q[1][0]);

```

generating points and the line III

```c
    // Close the file
    fclose(file);

    // Free allocated memory
    freeMat(Q, 2);
    freeMat(P, 2);
}

int main() {
    // Initialize points A(7, -2) and B(1, -5)
    double **A = createMat(2, 1);
    double **B = createMat(2, 1);

    A[0][0] = 7;
    A[1][0] = -2;
    B[0][0] = 1;
    B[1][0] = -5;
    // Generate trisection points and save them to asgn2.dat
    point_gen(A, B, "asgn2.txt");
    // Free the allocated memory for A and B
```

generating points and the line IV

```
    freeMat(A, 2);
    freeMat(B, 2);
    return 0;}
```

Plotting the figure using Python I

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

# Load the points from the text file
points = []
with open("asgn2.txt", 'r') as file:
    for line in file:
        # Check if the line contains coordinates
        if '(' in line and ')' in line:
            # Isolate the part with the coordinates
            coords_part = line.split('(')[-1].split(')')[0].strip()  # Get part between '(' and ')'
            try:
                # Split the coordinates and convert them to floats
                x, y = map(float, coords_part.split(','))
                points.append((x, y))  # Append as a tuple
            except ValueError as e:
                print(f"Error converting coordinates in line: '{line.strip()}': {e}")

```

Plotting the figure using Python II

```python
# Convert to numpy array for easier manipulation
points = np.array(points)

# Check if points were loaded correctly
if points.shape[0] < 4:
    raise ValueError("Data must contain at least four coordinates.")

# Extract the coordinates of points P, Q, B, and A
A = points[0]  # Trisection point Q
B = points[1]  # Trisection point P
P = points[2]  # Point B
Q = points[3]  # Point A


# Plot the points
plt.figure()

# Plot thick lines between points A and B, and P and Q
plt.plot([A[0], B[0]], [A[1], B[1]], color='gray', linewidth=2, label='Line AB')
```

Plotting the figure using Python III

```python
plt.plot([P[0], Q[0]], [P[1], Q[1]], color='gray', linewidth=2, label='Line PQ')

# Plot the points A, B, P, and Q
plt.scatter(A[0], A[1], color='red', marker='o')  # Point A
plt.scatter(B[0], B[1], color='green', marker='o')  # Point B
plt.scatter(P[0], P[1], color='blue', marker='o')   # Point P
plt.scatter(Q[0], Q[1], color='purple', marker='o')  # Point Q

# Label the points with coordinates
plt.text(A[0], A[1], f"A ({A[0]:.6f}, {A[1]:.6f})", fontsize=9, verticalalignment='bottom', horizontalalignment='right')
plt.text(B[0], B[1], f"B ({B[0]:.6f}, {B[1]:.6f})", fontsize=9, verticalalignment='bottom', horizontalalignment='right')
plt.text(P[0], P[1], f"P ({P[0]:.6f}, {P[1]:.6f})", fontsize=9, verticalalignment='bottom', horizontalalignment='right')
plt.text(Q[0], Q[1], f"Q ({Q[0]:.6f}, {Q[1]:.6f})", fontsize=9, verticalalignment='bottom', horizontalalignment='right')

# Label the axes and add a title
```

Plotting the figure using Python IV

```python
plt.xlabel("x")
plt.ylabel("y")
plt.title("Trisection Points P and Q with Points A and B")
plt.grid(True)

# Save the resulting figure

plt.show()

```