Collinearity of points

SRIHAAS GUNDA- EE24BTECH11026

Question

For what value of P are the points $(2, 1), (P, -1), and (-1, 3)$ collinear?

(10, 2019)

Variable name	Description	Formula
A	The point in 2-D plane with coordinates	$\begin{pmatrix} 2 \\ 1 \end{pmatrix}$
B	The point with unknown coordinate	$\begin{pmatrix} P \\ -1 \end{pmatrix}$
C	The point in 2-D plane with coordinates	$\begin{pmatrix} -1 \\ 3 \end{pmatrix}$

Collinearity Condition

Three points $\mathbf{A}, \mathbf{B}, \textit{and} \mathbf{C}$ are collinear if:

$$\text{rank}\begin{pmatrix}\mathbf{C}-\mathbf{B} & \mathbf{B}-\mathbf{A}\end{pmatrix} = 1$$

Setting Up the Matrix

$$\begin{pmatrix} -1-P & P-2 \\ 4 & -2 \end{pmatrix} \xleftrightarrow{R_1 \to R_1/(-P-1)} \tag{0.1}$$

Row Reduction

$$\begin{pmatrix} 1 & \frac{2-P}{P+1} \\ 4 & -2 \end{pmatrix} \xleftrightarrow{R_2 \to R_2 - 4R_1} \tag{0.2}$$

$$\begin{pmatrix} 1 & \frac{2-P}{P+1} \\ 0 & \frac{2P-10}{P+1} \end{pmatrix} \tag{0.3}$$

Finding P

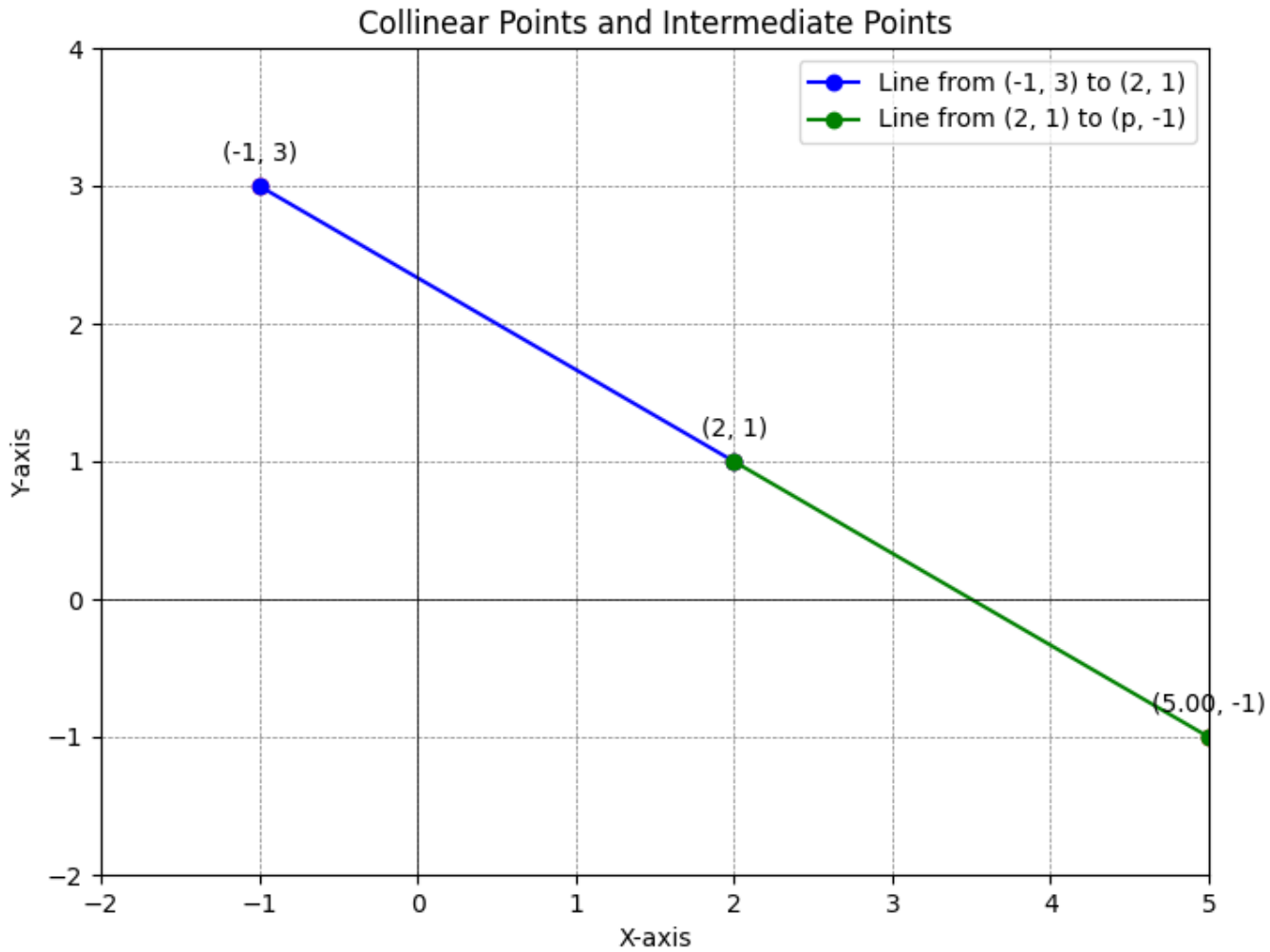
For the rank to be one:

$$\begin{aligned} 2P - 10 &= 0 \\ P - 5 &= 0 \\ P &= 5 \end{aligned} \tag{0.4}$$

Conclusion

The value of P for which the points are collinear is $P = 5$.

Plotting of Points



```c
#include <stdio.h>
#include <stdlib.h>
#include "libs/matfun.h"
#include "libs/geofun.h"
void calculate_p(double *p, double points[5][2]) {
    double x1 = 2.0, y1 = 1.0;
    double x3 = -1.0, y3 = 3.0;
    double **matrix = createMat(3, 3);
    matrix[0][0] = x1; matrix[0][1] = y1; matrix[0][2] = 1.0; // Point (2, 1)
    matrix[1][0] = *p; matrix[1][1] = -1.0; matrix[1][2] = 1.0; // Point (p, -1)
    matrix[2][0] = x3; matrix[2][1] = y3; matrix[2][2] = 1.0; // Point (-1, 3)
    int rank = 3;
    for (int i = 0; i < 3; i++) {
        if (matrix[i][0] == 0 && matrix[i][1] == 0 && matrix[i][2] == 0) {
            rank--;
            continue;
        }
        for (int j = i + 1; j < 3; j++) {
            if (matrix[j][i] != 0) {
                double ratio = matrix[j][i] / matrix[i][i];
                for (int k = 0; k < 3; k++) {
                    matrix[j][k] -= ratio * matrix[i][k];
                }
            }
        }
    }
    if (rank < 3) {
        *p = (y3 - y1) / (x3 - x1) * (x3 - x1) + y1;
    } else {
```

```
        *p = 5.0;
    }

    for (int i = 0; i < 5; i++) {
        double t = (double)i / 4;
        points[i][0] = (1 - t) * x1 + t * x3;
        points[i][1] = (1 - t) * y1 + t * y3;
    }
    for (int i = 0; i < 3; i++) {
        free(matrix[i]);
    }
    free(matrix);
}
```

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

lib = ctypes.CDLL('./libcalculate_p.so')
lib.calculate_p.argtypes = [ctypes.POINTER(ctypes.c_double), ctypes.POINTER(ctypes.c_double * 10)]

p = ctypes.c_double(0.0)

points_array_type = ctypes.c_double * 10

points = points_array_type()
lib.calculate_p(ctypes.byref(p), points)

points_list = np.array([[points[i * 2], points[i * 2 + 1]] for i in range(5)])
original_points = np.array([[2, 1], [p.value, -1], [-1, 3]])

plt.figure(figsize=(8, 6))
plt.plot([2, -1], [1, 3], label='Line from (-1, 3) to (2, 1)', color='blue', marker='o')
plt.plot([2, p.value], [1, -1], label='Line from (2, 1) to (p, -1)', color='green', marker='o')

plt.scatter(original_points[:, 0], original_points[:, 1], color='red')

plt.annotate('(-1, 3)', xy=(-1, 3), textcoords="offset points", xytext=(0,10), ha='center')
plt.annotate('(2, 1)', xy=(2, 1), textcoords="offset points", xytext=(0,10), ha='center')
plt.annotate(f'({p.value:.2f}, -1)', xy=(p.value, -1), textcoords="offset points", xytext=(0,10), ha='
    center')

plt.xlim(-2, 5)
```

```
plt.ylim(-2, 4)

plt.title('Collinear Points and Intermediate Points')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
plt.axhline(0, color='black', linewidth=0.5, ls='-')
plt.axvline(0, color='black', linewidth=0.5, ls='-')
plt.grid(color='gray', linestyle='--', linewidth=0.5)
plt.legend()
plt.savefig('../figs/fig.png')
plt.show()
```