4.13.34

Jnanesh Sathisha karmar - EE25BTECH11029

September 30,2025

Question

The equations to a pair of opposite sides of parallogram are $x^2 - 5x + 6 = 0$ and $y^2 - 6y + 5 = 0$, the equations to its diagonals are

1. $x + 4y = 13, y = 4x - 7$
2. $4x + y = 13, y = 4x - 7$
3. $4x + y = 13, 4y = x - 7$
4. $y - 4x = 13, y + 4x = 7$

Equation

Given details

Equation 1:

$$x^2 - 5x + 6 = 0 \tag{1}$$

This equation can be factored into: (2)

$$(x - 2)(x - 3) = 0 \tag{3}$$

This gives us two vertical lines: (4)

$$x = 2 \tag{5}$$

$$x = 3 \tag{6}$$

Equation

Equation 2:

$$y^2 - 6y + 5 = 0 \tag{7}$$

This equation can be factored into: (8)

$$(y - 1)(y - 5) = 0 \tag{9}$$

This gives us two horizontal lines: (10)

$$y = 1 \tag{11}$$

$$y = 5 \tag{12}$$

Theoretical Solution

Through the intersection of these 4 lines we can find the 4 vertices of the parallelogram

$$\text{Intersection of } x = 2 \text{ and } y = 1 \text{is the point } (2,1) \text{ .Let vector } \mathbf{A} = \begin{pmatrix} 2 \\ 1 \end{pmatrix} \tag{13}$$

$$\text{Intersection of } x = 3 \text{ and } y = 1 \text{is the point } (3,1) \text{ .Let vector } \mathbf{B} = \begin{pmatrix} 3 \\ 1 \end{pmatrix} \tag{14}$$

$$\text{Intersection of } x = 3 \text{ and } y = 5 \text{is the point } (3,5) \text{ .Let vector } \mathbf{C} = \begin{pmatrix} 3 \\ 5 \end{pmatrix} \tag{15}$$

$$\text{Intersection of } x = 2 \text{ and } y = 5 \text{is the point } (2,5) \text{ .Let vector } \mathbf{D} = \begin{pmatrix} 2 \\ 5 \end{pmatrix} \tag{16}$$

Theoretical Solution

The equations of the diagonals can be found using the vector equation of a line, which is given by $\mathbf{r}(t) = \mathbf{p_0} + t\mathbf{d}$, where $\mathbf{p_0}$ is a starting point and $\mathbf{d}$ is the direction vector.

The equation of the diagonal **AC**, passing through $\mathbf{A}\,(2,1)$ and $\mathbf{C}\,(3,5)$, is:

$$\text{The starting point vector is } \mathbf{p_0} = \mathbf{A} = \begin{pmatrix} 2 \\ 1 \end{pmatrix} \tag{18}$$

$$\text{The direction vector is } \mathbf{d} = \mathbf{C} - \mathbf{A} = \begin{pmatrix} 3 \\ 5 \end{pmatrix} - \begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 4 \end{pmatrix} \tag{19}$$

$$\text{The vector equation is } \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \end{pmatrix} + t \begin{pmatrix} 1 \\ 4 \end{pmatrix} \tag{20}$$

$$\tag{21}$$

Theoretical Solution

$$\text{From this, we get } x = 2 + t \implies t = x - 2 \tag{22}$$

$$\text{And } y = 1 + 4t \tag{23}$$

$$\text{Substituting for t: } y = 1 + 4(x - 2) \tag{24}$$

$$y = 1 + 4x - 8 \tag{25}$$

$$y = 4x - 7 \tag{26}$$

The equation of the diagonal **BD**, passing through **B** $(3, 1)$ and **D** $(2, 5)$, is:

Theoretical Solution

$$\text{The vector equation is } \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 3 \\ 1 \end{pmatrix} + t \begin{pmatrix} -1 \\ 4 \end{pmatrix} \tag{27}$$

$$\text{From this, we get } x = 3 - t \implies t = 3 - x \tag{28}$$

$$\text{And } y = 1 + 4t \tag{29}$$

$$\text{Substituting for t: } y = 1 + 4(3 - x) \tag{30}$$

$$y = 1 + 12 - 4x \tag{31}$$

$$4x + y = 13 \tag{32}$$

$$\tag{33}$$

Therefore the equations of both the diagonals are:

$$y = 4x - 7 \tag{34}$$

$$4x + y = 13 \tag{35}$$

Hence the answer is option 2.

C Code (1) - Function to store the points

```c
#include <stddef.h>

void find_diagonals(double x1, double x2, double y1, double y2,
    double* diag1_coeffs, double* diag2_coeffs) {
    if (diag1_coeffs == NULL || diag2_coeffs == NULL) {
        return;
    }

    diag1_coeffs[0] = y2 - y1;
    diag1_coeffs[1] = x1 - x2;
    diag1_coeffs[2] = (x2 * y1) - (x1 * y2);

    diag2_coeffs[0] = y1 - y2;
    diag2_coeffs[1] = x1 - x2;
    diag2_coeffs[2] = (x2 * y2) - (x1 * y1);
}
```

Python Code - Using Shared Object

```
import ctypes
import numpy as np
import matplotlib.pyplot as plt
def solve_quadratic(a, b, c):
    discriminant = b**2 - 4*a*c
    if discriminant < 0:
        return None, None
    r1 = (-b + np.sqrt(discriminant)) / (2*a)
    r2 = (-b - np.sqrt(discriminant)) / (2*a)
    return r1, r2
def main():
    lib_path = "./diagonals.so"
    try:
        diag_lib = ctypes.CDLL(lib_path)
    except OSError as e:
        print(f"Error loading shared library: {e}")
        print("Please ensure 'diag_calculator.so' exists. You may
            need to compile the C code first using 'sh compile.sh
        '")
```

Python Code - Using Shared Object

```
diag_lib.find_diagonals.argtypes = [
    ctypes.c_double, ctypes.c_double,
    ctypes.c_double, ctypes.c_double,
    ctypes.POINTER(ctypes.c_double),
    ctypes.POINTER(ctypes.c_double)
]
diag_lib.find_diagonals.restype = None

x1, x2 = solve_quadratic(1, -5, 6)
y1, y2 = solve_quadratic(1, -6, 5)

if x1 is None or y1 is None:
    print("Error: Could not solve quadratic equations. Check
        coefficients.")
    return
```

Python Code - Using Shared Object

```
    print(f"Parallelogram lines: x={x1}, x={x2}, y={y1}, y={y2
        }")

Diag1Coeffs = (ctypes.c_double * 3)()
Diag2Coeffs = (ctypes.c_double * 3)()

diag_lib.find_diagonals(x1, x2, y1, y2, Diag1Coeffs,
    Diag2Coeffs)

d1 = [c for c in Diag1Coeffs]
d2 = [c for c in Diag2Coeffs]
print(f"Diagonal 1 (Ax+By+C=0): A={d1[0]:.1f}, B={d1[1]:.1f},
     C={d1[2]:.1f}")
print(f"Diagonal 2 (Ax+By+C=0): A={d2[0]:.1f}, B={d2[1]:.1f},
     C={d2[2]:.1f}")
```

Python Code - Using Shared Object

```
fig, ax = plt.subplots(figsize=(8, 8))

    parallelogram_x = [x1, x2, x2, x1, x1]
    parallelogram_y = [y1, y1, y2, y2, y1]
    ax.plot(parallelogram_x, parallelogram_y, 'b-', label='
        Parallelogram Sides', linewidth=2)

    plot_x_range = np.linspace(min(x1, x2) - 1, max(x1, x2) + 1,
        100)

    def plot_line(coeffs, style, label):
        A, B, C = coeffs
        if abs(B) > 0:
            y_vals = (-A * plot_x_range - C) / B
            ax.plot(plot_x_range, y_vals, style, label=label)
        else:
            x_val = -C / A
            ax.axvline(x=x_val, linestyle=style.strip('-'), color=
                style[0], label=label)
```

Python Code - Using Shared Object

```
    plot_line(d1, 'r--', 'Diagonal 1')
    plot_line(d2, 'g--', 'Diagonal 2')

    ax.set_title('Parallelogram and its Diagonals')
    ax.set_xlabel('X-axis')
    ax.set_ylabel('Y-axis')
    ax.grid(True, linestyle=':', alpha=0.6)
    ax.legend()

    padding = 2
    ax.set_xlim(min(x1, x2) - padding, max(x1, x2) + padding)
    ax.set_ylim(min(y1, y2) - padding, max(y1, y2) + padding)
    ax.set_aspect('equal', adjustable='box')
```

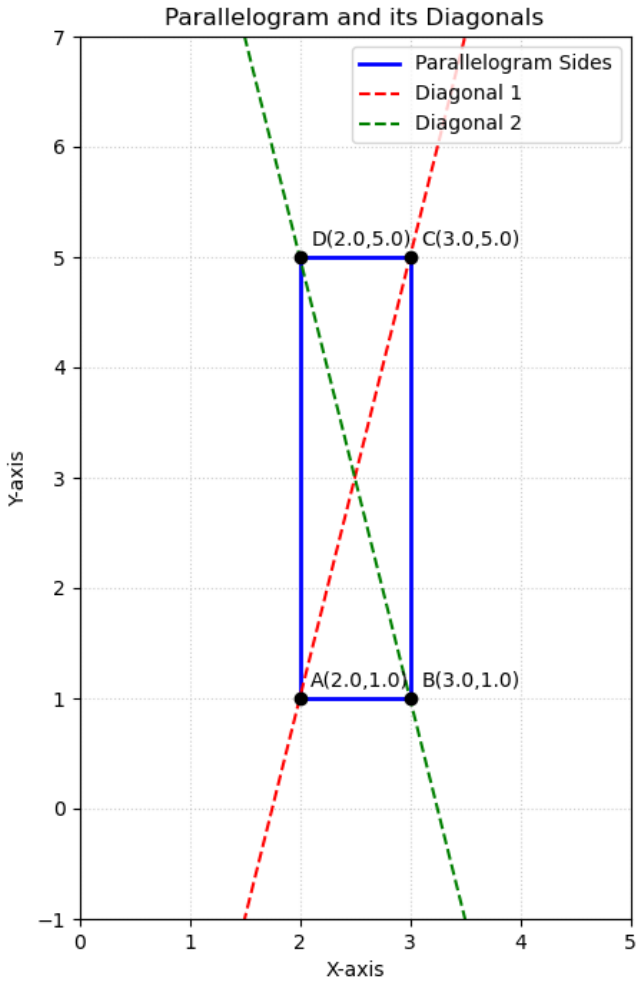
Python Code - Using Shared Object

```
    x_min, x_max = min(x1, x2), max(x1, x2)
    y_min, y_max = min(y1, y2), max(y1, y2)
    vertices = {'A':(x_min, y_min), 'B':(x_max, y_min), 'C':(
        x_max, y_max), 'D':(x_min, y_max)}
    for name, (px, py) in vertices.items():
        ax.plot(px, py, 'ko')
        ax.text(px + 0.1, py + 0.1, f'{name}({px:.1f},{py:.1f})')

    plt.savefig("./figs/diagonals.png")
    subprocess.run(shlex.split('termux-open ../figs/diagonals.png
        '))
    plt.show()

if __name__ == "__main__":
    main()
```

Plot-Using Both C and Python



Python Code

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

def solve_quadratic(a, b, c):
    discriminant = b**2 - 4*a*c
    if discriminant < 0:
        return None, None
    r1 = (-b + np.sqrt(discriminant)) / (2*a)
    r2 = (-b - np.sqrt(discriminant)) / (2*a)
    return r1, r2

def find_diagonals_python(x1, x2, y1, y2):
    A1 = y2 - y1
    B1 = x1 - x2
    C1 = (x2 * y1) - (x1 * y2)
    diag1_coeffs = [A1, B1, C1]
```

Python Code

```
    A2 = y1 - y2
    B2 = x1 - x2
    C2 = (x2 * y2) - (x1 * y1)
    diag2_coeffs = [A2, B2, C2]

    return diag1_coeffs, diag2_coeffs

def main():
    x1, x2 = solve_quadratic(1, -5, 6)
    y1, y2 = solve_quadratic(1, -6, 5)

    if x1 is None or y1 is None:
        print("Error: Could not solve quadratic equations. Check
            coefficients.")
        return
```

Python Code

```
print(f"Parallelogram lines: x={x1}, x={x2}, y={y1}, y={y2}")

d1, d2 = find_diagonals_python(x1, x2, y1, y2)

print(f"Diagonal 1 (Ax+By+C=0): A={d1[0]:.1f}, B={d1[1]:.1f},
    C={d1[2]:.1f}")
print(f"Diagonal 2 (Ax+By+C=0): A={d2[0]:.1f}, B={d2[1]:.1f},
    C={d2[2]:.1f}")

fig, ax = plt.subplots(figsize=(8, 8))

parallelogram_x = [x1, x2, x2, x1, x1]
parallelogram_y = [y1, y1, y2, y2, y1]
ax.plot(parallelogram_x, parallelogram_y, 'b-', label='
   Parallelogram Sides', linewidth=2)
```

Python Code

```
    plot_x_range = np.linspace(min(x1, x2) - 1, max(x1, x2) + 1,
        100)

    def plot_line(coeffs, style, label):
        A, B, C = coeffs
        if abs(B) > 0:
            y_vals = (-A * plot_x_range - C) / B
            ax.plot(plot_x_range, y_vals, style, label=label)
        else:
            x_val = -C / A
            ax.axvline(x=x_val, linestyle=style.strip('-'), color=
                style[0], label=label)

    plot_line(d1, 'r--', 'Diagonal 1')
    plot_line(d2, 'g--', 'Diagonal 2')
```

Python Code

```
    ax.set_title('Parallelogram and its Diagonals')
    ax.set_xlabel('X-axis')
    ax.set_ylabel('Y-axis')
    ax.grid(True, linestyle=':', alpha=0.6)
    ax.legend()

    padding = 2
    ax.set_xlim(min(x1, x2) - padding, max(x1, x2) + padding)
    ax.set_ylim(min(y1, y2) - padding, max(y1, y2) + padding)
    ax.set_aspect('equal', adjustable='box')
```

Python Code

```
    x_min, x_max = min(x1, x2), max(x1, x2)
    y_min, y_max = min(y1, y2), max(y1, y2)
    vertices = {'A':(x_min, y_min), 'B':(x_max, y_min), 'C':(
        x_max, y_max), 'D':(x_min, y_max)}
    for name, (px, py) in vertices.items():
        ax.plot(px, py, 'ko')
        ax.text(px + 0.1, py + 0.1, f'{name}({px:.1f},{py:.1f})')
    plt.savefig("./figs/diagonals2.png")
    plt.show()

if __name__ == "__main__":
    main()
```

Plot-Using only Python