4.13.36

Sai Sreevallabh - EE25BTECH11031

October 4, 2025

Question

Let PQR be a right angled isosceles triangle, right at $P(2, 1)$. If the equation of the line QR is $2x + y = 3$, then the equation representing the pair of lines PQ and PR is

1. $3x^2 - 3y^2 + 8xy + 20x + 10y + 25 = 0$
2. $3x^2 - 3y^2 + 8xy - 20x - 10y + 25 = 0$
3. $3x^2 - 3y^2 + 8xy + 10x + 15y + 20 = 0$
4. $3x^2 - 3y^2 - 8xy - 10x - 15y - 20 = 0$

Theoretical Solution

Given point is $\mathbf{P} = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$ and given line can be written as

$$\mathbf{n}^\top \mathbf{x} = c \tag{1}$$

where, $\mathbf{n} = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$ and $c = 3$.
Parametric form of line through $\mathbf{P}$ is

$$\mathbf{r} = \mathbf{P} + \lambda \mathbf{m} \tag{2}$$

Theoretical Solution

Using this, we can represent points Q and R as

$$\mathbf{Q} = \mathbf{P} + \lambda_1 \mathbf{m_1} \tag{3}$$

$$\mathbf{R} = \mathbf{P} + \lambda_2 \mathbf{m_2} \tag{4}$$

where, $\mathbf{m_1} = \begin{pmatrix} 1 \\ m_1 \end{pmatrix}$ and $\mathbf{m_2} = \begin{pmatrix} 1 \\ m_2 \end{pmatrix}$ are direction vectors of lines $\mathbf{Q} - \mathbf{P}$ and $\mathbf{R} - \mathbf{P}$, while m_1 and m_2 are the respective slopes.

Theoretical Solution

Given that the lines are perpendicular,

$$\mathbf{m_1}^\top \mathbf{m_2} = 0 \tag{5}$$

$$\implies m_1 m_2 = -1 \tag{6}$$

Substituting equation (3) in (1)

$$\mathbf{n}^\top (\mathbf{P} + \lambda_1 \mathbf{m_1}) = c \tag{7}$$

$$\implies \lambda_1 = \frac{c - \mathbf{n}^\top \mathbf{P}}{\mathbf{n}^\top \mathbf{m_1}} \tag{8}$$

Theoretical Solution

Substituting the values, we get

$$\lambda_1 = \frac{-2}{2+m_1} \tag{9}$$

Similarly, substituting equation (4) in (1)

$$\lambda_2 = \frac{c - \mathbf{n}^\top \mathbf{P}}{\mathbf{n}^\top \mathbf{m_2}} \tag{10}$$

Substituting values,

$$\lambda_2 = \frac{-2}{2+m_2} \tag{11}$$

$$\implies \lambda_2 = \frac{-2m_1}{2m_1 - 1} \tag{12}$$

Theoretical Solution

Given that the triangle is isosceles,

$$\|\mathbf{Q}-\mathbf{P}\| = \|\mathbf{R}-\mathbf{P}\| \tag{13}$$

$$\implies |\lambda_1|\,\|\mathbf{m_1}\| = |\lambda_2|\,\|\mathbf{m_2}\| \tag{14}$$

$$\implies \left|\frac{-2}{2+m_1}\right|\sqrt{1+m_1^2} = \left|\frac{-2m_1}{2m_1-1}\right|\sqrt{1+\left(\frac{-1}{m_1}\right)^2} \tag{15}$$

$$\implies \left|\frac{2}{2+m_1}\right| = \left|\frac{2}{2m_1-1}\right| \tag{16}$$

Solving the above, we get

$$m_1 = 3 \text{ or } m_1 = \frac{-1}{3} \tag{17}$$

Theoretical Solution

Correspondingly,

$$m_2 = \frac{-1}{3} \text{ or } m_2 = 3 \tag{18}$$

So, the equations of the two required lines are

$$3x - y - 5 = 0 \quad \text{and} \quad x + 3y - 5 = 0 \tag{19}$$

$\therefore$ Multiplying the above two equations, we get the pair of straight lines to be

$$3x^2 - 3y^2 + 8xy - 20x - 10y + 25 = 0$$

C Code - Solving Using Gaussian Elimination

```c
#include <stdio.h>

void Solve_Gaussian(double A[3], double B[3], double sol
    [2]) {
    // If A[0] == 0, swap rows to avoid division by zero
    //Also covers the case where the matrix is diagonal.
    if (A[0] == 0) {
        for (int i = 0; i < 3; i++) {
            double temp = A[i];
            A[i] = B[i];
            B[i] = temp;
        }
    }
```

C Code - Solving Using Gaussian Elimination

```c

    double factor = B[0] / A[0];
    for (int i = 0; i < 3; i++) {
        B[i] = B[i] - factor * A[i];
    }

    sol[1] = B[2] / B[1];
    sol[0] = (A[2] - A[1] * sol[1]) / A[0];
}
```

Python Code - Using Shared Object

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

c_lib = ctypes.CDLL("./code.so")

c_lib.Solve_Gaussian.argtypes = [ctypes.c_double*3,
    ctypes.c_double*3, ctypes.c_double*2]

#line QR
A = (ctypes.c_double*3)(2,1,3)

#line PR
B = (ctypes.c_double*3)(3,-1,5)

#line PQ
C = (ctypes.c_double*3)(1,3,5)
```

```
P = np.array([2,1])

Q = (ctypes.c_double*2)(0.0,0.0)
c_lib.Solve_Gaussian(A,C,Q)

R = (ctypes.c_double*2)(0.0,0.0)
c_lib.Solve_Gaussian(A,B,R)

plt.scatter([P[0],Q[0],R[0]], [P[1],Q[1],R[1]])

plt.plot([P[0],R[0]],[P[1],R[1]], label = "PR: 3x-y=5")
plt.plot([P[0],Q[0]],[P[1],Q[1]], label = "PQ: x+3y=5")
plt.plot([0,2], [3,-1], c='green', label = "QR: $2x+y=3$"
    )
```

Python Code - Using Shared Object

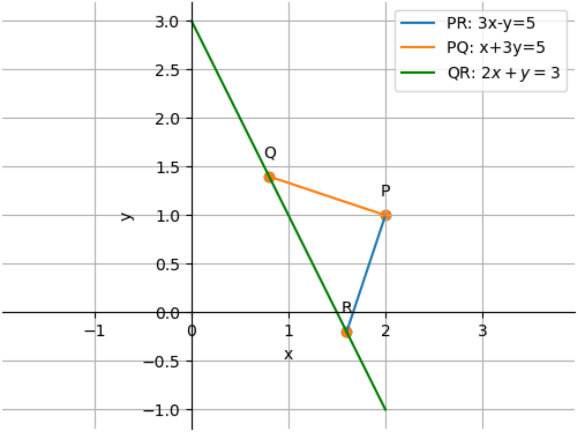
```
R_p = np.array([R[0],R[1]], dtype=np.float64).reshape
    (-1,1)
Q_p = np.array([Q[0],Q[1]], dtype=np.float64).reshape
    (-1,1)
P_p = np.array([2,1]).reshape(-1,1)


tri_coords = np.block([[P_p,Q_p,R_p]])
plt.scatter(tri_coords[0,:], tri_coords[1,:])
vert_labels = ['P','Q','R']
for i, txt in enumerate(vert_labels):
    plt.annotate(f'{txt}\n',
                 (tri_coords[0,i], tri_coords[1,i]),
                 textcoords="offset points",
                 xytext=(0.2,0.2),
                 ha='center')
```

```
ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position('zero')
ax.spines['right'].set_color('none')
ax.spines['left'].set_position('zero')
plt.xlabel('x')
plt.ylabel('y')
plt.legend(loc='best')
plt.grid()
plt.axis('equal')

plt.savefig("../Figs/plot(py+C).png")
plt.show()
```

Plot-Using Both C and Python



Python Code

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

P = np.array([2, 1])

#solving Ax=b, to find x
A = np.array([[3, -1],
           [2, 1]])
b = np.array([5, 3])
R = LA.solve(A, b)

A = np.array([[1, 3],
           [2, 1]])
b = np.array([5, 3])
Q = LA.solve(A, b)
```

Python Code

```
plt.scatter([P[0],Q[0],R[0]], [P[1],Q[1],R[1]])

plt.plot([P[0],R[0]],[P[1],R[1]], label = "PR: 3x-y=5")
plt.plot([P[0],Q[0]],[P[1],Q[1]], label = "PQ: x+3y=5")
plt.plot([0,2], [3,-1], c='green', label = "QR: $2x+y=3$"
    )

R_p = np.array([R[0],R[1]], dtype=np.float64).reshape
    (-1,1)
Q_p = np.array([Q[0],Q[1]], dtype=np.float64).reshape
    (-1,1)
P_p = np.array([2,1]).reshape(-1,1)
```

Python Code

```
tri_coords = np.block([[P_p,Q_p,R_p]])
plt.scatter(tri_coords[0,:], tri_coords[1,:])
vert_labels = ['P','Q','R']
for i, txt in enumerate(vert_labels):
   plt.annotate(f'{txt}\n',
               (tri_coords[0,i], tri_coords[1,i]),
               textcoords="offset points",
               xytext=(0.2,0.2),
               ha='center')
```

```python

ax = plt.gca()
ax.spines['top'].set_color('none')
ax.spines['bottom'].set_position('zero')
ax.spines['right'].set_color('none')
ax.spines['left'].set_position('zero')
plt.xlabel('x')
plt.ylabel('y')
plt.legend(loc='best')
plt.grid()
plt.axis('equal')

plt.savefig("../Figs/plot(py).png")
plt.show()
```

Plot-Using Python only