1.5.24

M Chanakya Srinivas- EE25BTECH11036

Problem Statement

A line intersects the Y-axis and X-axis at the points

$$P = (0, b), \quad Q = (c, 0)$$

respectively. If $(2, -5)$ is the midpoint of $\overline{PQ}$, then find the coordinates of P and Q.

Vector Representation

We represent the points as column vectors:

$$\mathbf{P} = \begin{pmatrix} 0 \\ b \end{pmatrix}, \quad \mathbf{Q} = \begin{pmatrix} c \\ 0 \end{pmatrix}, \quad \mathbf{M} = \begin{pmatrix} 2 \\ -5 \end{pmatrix}.$$

(i) Rank/Collinearity Condition

Since P, Q, M are collinear,

$$\operatorname{rank}\begin{pmatrix}\mathbf{P}-\mathbf{M} & \mathbf{Q}-\mathbf{M}\end{pmatrix}^{\top} = 1.$$

$$\begin{pmatrix}\mathbf{P}-\mathbf{M} & \mathbf{Q}-\mathbf{M}\end{pmatrix}^{\top} = \begin{pmatrix}-2 & c-2 \\ b+5 & 5\end{pmatrix}$$

$$\xrightarrow{R_2 \leftarrow -2R_2-(b+5)R_1} \begin{pmatrix}-2 & c-2 \\ 0 & -10-(b+5)(c-2)\end{pmatrix}.$$

For rank $= 1$, the last entry must vanish:

$$(b+5)(c-2) = -10. \qquad (*)$$

(ii) Midpoint Condition

The midpoint of P and Q is:

$$\mathbf{M} = \frac{\mathbf{P} + \mathbf{Q}}{2}$$

$$\begin{pmatrix} 2 \\ -5 \end{pmatrix} = \frac{1}{2}\begin{pmatrix} 0 \\ b \end{pmatrix} + \frac{1}{2}\begin{pmatrix} c \\ 0 \end{pmatrix} = \frac{1}{2}\begin{pmatrix} c \\ b \end{pmatrix}.$$

Thus,

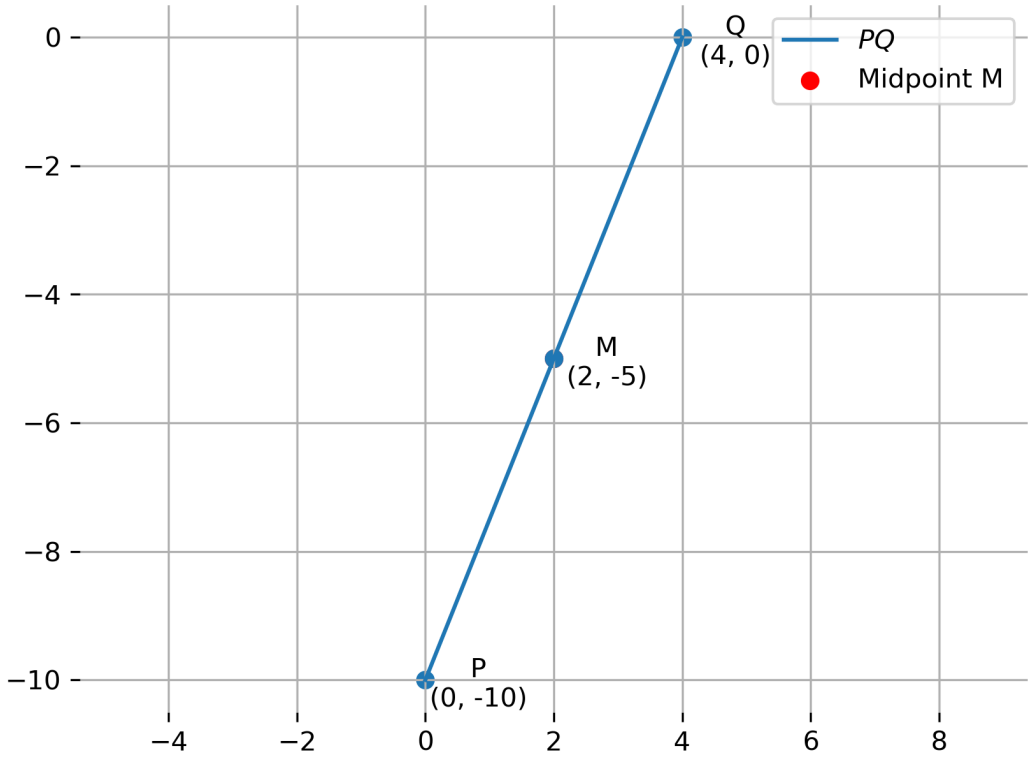
$$\begin{pmatrix} c \\ b \end{pmatrix} = \begin{pmatrix} 4 \\ -10 \end{pmatrix} \quad \Longrightarrow \quad c = 4,\ b = -10.$$

Final Answer

Therefore, the intercept points are:

$$\mathbf{P} = \begin{pmatrix} 0 \\ -10 \end{pmatrix}, \qquad \mathbf{Q} = \begin{pmatrix} 4 \\ 0 \end{pmatrix}.$$

Illustration



Plot shows $P(0,-10)$, $Q(4,0)$ and midpoint $M(2,-5)$.

C Code - Section formula function

```c
#include <stdio.h>

int main() {

    printf(Problem 1.5.24:\n);
    printf(A line intersects the Y-axis and X-axis at P=(0,b) and
         Q=(c,0).\n);
    printf(If (2,-5) is the midpoint of PQ, find P and Q.\n\n);


    printf(Step (i): Rank / Collinearity condition\n);
    printf(From collinearity, (b+5)(c-2) = -10\n\n);


    printf(Step (ii): Midpoint condition\n);
    int Mx = 2, My = -5;
    int c = 2 * Mx; // c/2 = 2 -> c = 4
    int b = 2 * My; // b/2 = -5 -> b = -10
    printf(Midpoint gives: c = %d, b = %d\n\n, c, b);
```

C Code - Section formula function

```
    int Px = 0, Py = b;
    int Qx = c, Qy = 0;
    printf(Final Answer:\n);
    printf(P = (%d, %d)\n, Px, Py);
    printf(Q = (%d, %d)\n, Qx, Qy);


    int midx = (Px + Qx) / 2;
    int midy = (Py + Qy) / 2;
    printf(\nVerification:\n);
    printf(Midpoint of P and Q = (%d, %d)\n, midx, midy);

    return 0;
}
```

Python Code through shared output

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

print(Step 1: Rank condition (collinearity))
print(Matrix formed from (P-M) and (Q-M):)
print([[-2, c-2], [b+5, 5]])

print(Row operation: R2 -> -2*R2 - (b+5)*R1)
print(=> [[-2, c-2], [0, -10 - (b+5)(c-2)]])

print(For rank=1: (b+5)(c-2) = -10)
```

Python Code through shared output

```
print(Step 2: Midpoint condition)
Mx, My = 2, -5
print(M = ((0+c)/2, (b+0)/2) = (2,-5))

c = 2*Mx
b = 2*My
print(fFrom midpoint: c = {c}, b = {b})
```

Python Code through shared output

```
P = (0, b)
Q = (c, 0)

print(Final Answer:)
print(fP = {P})
print(fQ = {Q})
```

Python Code through shared output

```
midpoint = ((P[0]+Q[0])/2, (P[1]+Q[1])/2)
print(Verification:)
print(fMidpoint of P and Q = {midpoint})
```

Python Code through shared output

```
# --- Step 5: Plot the graph ---
plt.figure(figsize=(6,6))
plt.axhline(0, color='black', linewidth=0.8) # X-axis
plt.axvline(0, color='black', linewidth=0.8) # Y-axis

# Plot line PQ
plt.plot([P[0], Q[0]], [P[1], Q[1]], 'b-', label=Line PQ)

# Mark points
plt.scatter(*P, color='red')
plt.scatter(*Q, color='green')
plt.scatter(*M, color='purple')

plt.text(P[0]-0.5, P[1], fP({int(P[0])},{int(P[1])}), fontsize
    =10, color=red)
plt.text(Q[0]+0.2, Q[1], fQ({int(Q[0])},{int(Q[1])}), fontsize
    =10, color=green)
plt.text(M[0]+0.2, M[1], fM({int(M[0])},{int(M[1])}), fontsize
    =10, color=purple)
```

Python Code through shared output

```
plt.title(Line PQ with Midpoint M(2,-5))
plt.grid(True)
plt.legend()
plt.axis(equal)

plt.show()
```

Direct Python Code

```
import sys # for path to external scripts
import numpy as np
import numpy.linalg as LA
import matplotlib.pyplot as plt

# local imports
from libs.line.funcs import *
from libs.triangle.funcs import *
from libs.conics.funcs import circ_gen

# --- Step 1: Rank Matrix condition ---
print(Step 1: Rank condition between b and c)
print(Take M = (2,-5), P = (0,b), Q = (c,0))

# Construct rank matrix for collinearity of P, Q, M
# Vectors (P-M) and (Q-M) must be linearly dependent => rank = 1
# Matrix form: [[Px-Mx, Qx-Mx], [Py-My, Qy-My]]
# Here M=(2,-5)
M = np.array(([2, -5])).reshape(-1,1)
```

Direct Python Code

```
b, c = symbols = (None, None) # placeholders for explanation

print(Matrix formed from (P-M) and (Q-M):)
print([[-2, c-2], [b+5, 5]])

print(Row operation: R2 -> -2*R2 - (b+5)*R1)
print(=> [[-2, c-2], [0, -10 - (b+5)(c-2)]])

print(For rank=1: (b+5)(c-2) = -10 (Relation 1)\n)

# --- Step 2: Midpoint relation ---
print(Step 2: Midpoint relation)
print(Midpoint M = ((0+c)/2, (b+0)/2) = (2, -5))

c = 2 * M[0,0]
b = 2 * M[1,0]
print(fc/2 = 2 => c = {c})
print(fb/2 = -5 => b = {b} (Relation 2)\n)
```

Direct Python Code

```
print(Step 3: Solve)
print(fCoordinates of P = (0, {b}))
print(fCoordinates of Q = ({c}, 0)\n)

# --- Step 4: Verification ---
mid = (P+Q)/2
print(Verification: midpoint of P and Q =, mid.ravel())

# --- Step 5: Plotting ---
x_PQ = line_gen(P,Q)
plt.plot(x_PQ[0,:], x_PQ[1,:], label='$PQ$')

# Mark points
coords = np.block([[P,Q,M]])
vert_labels = ['P','Q','M']
plt.scatter(coords[0,:], coords[1,:], color=['green','red','
    magenta'])
for i, txt in enumerate(vert_labels):
    plt.annotate(f'{txt}\n({coords[0,i]:.0f},{coords[1,i]:.0f})',
```

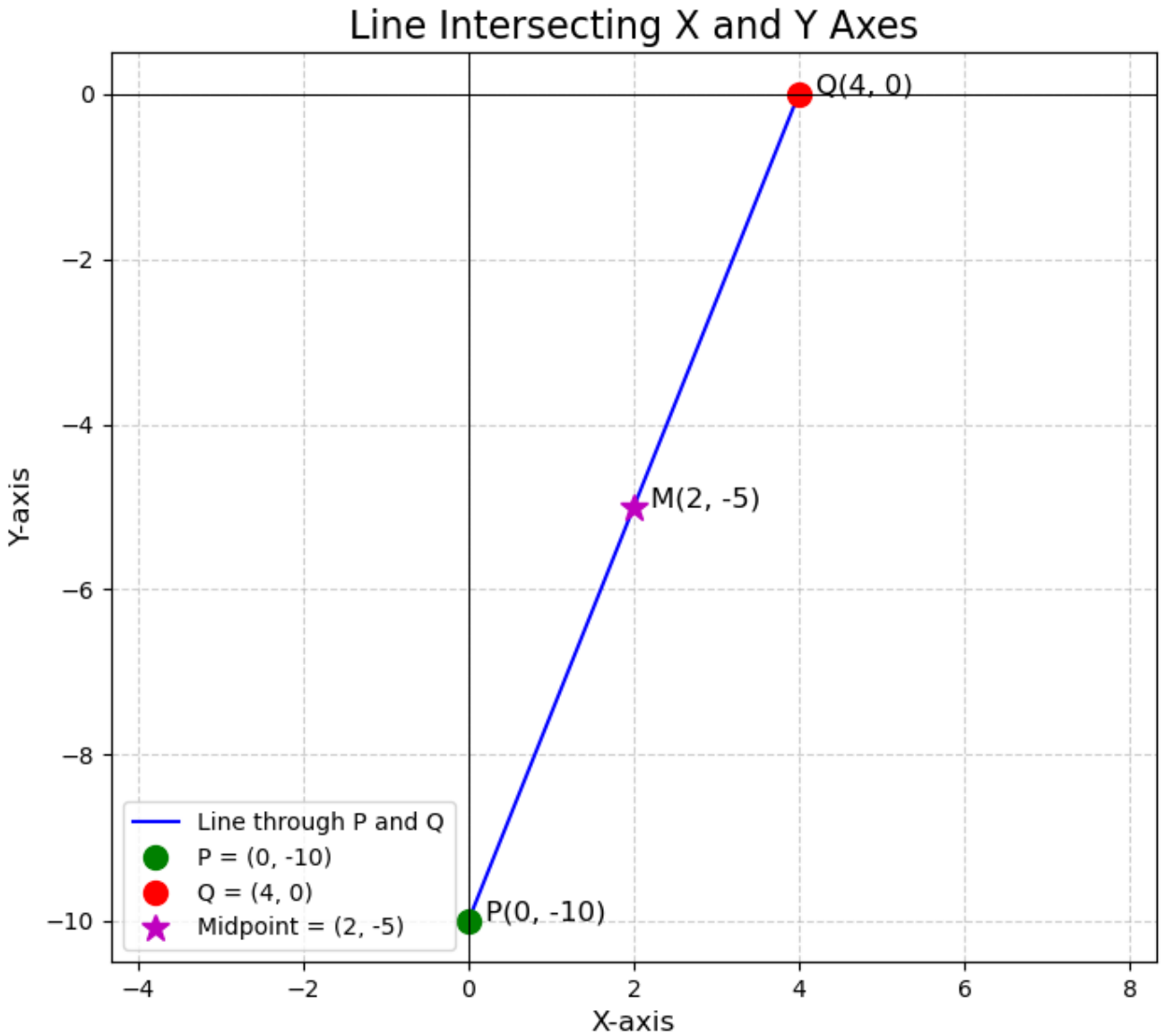
Direct Python Code

```
# Axis styling
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()
plt.axis('equal')

# Save figure as PDF
outfile_pdf = 'chapters/10/7/2/2/figs/fig.pdf'
plt.savefig(outfile_pdf)

# Save figure as PNG
outfile_png = 'chapters/10/7/2/2/figs/fig.png'
plt.savefig(outfile_png, dpi=300)
```

Direct Python Code

```
# Open image depending on system
try:
    import platform, subprocess, shlex
    if termux in platform.platform().lower(): # Android Termux
        subprocess.run(shlex.split(ftermux-open {outfile_png}))
    else: # Linux desktop
        subprocess.run(shlex.split(fxdg-open {outfile_png}))
except Exception as e:
    print(fCould not auto-open file. Saved at {outfile_png})
```

Plot by python using shared output from c



Plot by python only