4.3.50

Namaswi-EE25BTECH11060

september 14,2025

Question

Find the equation of the lines which makes intercepts -3 and 2 on the x and y axes respectively

solution

Let $(-3, 0)$ and $(0, 2)$ be the intercept points

$$\mathbf{m} = \begin{pmatrix} -3 \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ 2 \end{pmatrix} \tag{1}$$

$$\mathbf{m} = \begin{pmatrix} 1 \\ \frac{2}{3} \end{pmatrix} \tag{2}$$

$$\mathbf{n} = \begin{pmatrix} \frac{-2}{3} \end{pmatrix} \tag{3}$$

solution

Equation of line is given by $n^{\top}(x-h)=0$

$$\begin{pmatrix} \frac{-2}{3} & 1 \end{pmatrix} \left(x - \begin{pmatrix} 0 \\ 2 \end{pmatrix} \right) = 0 \tag{4}$$

$$\begin{pmatrix} \frac{-2}{3} & 1 \end{pmatrix} x = 2 \tag{5}$$

C Code

```c
#include <stdio.h>
int main() {
    // Intercept points
    double A[2] = {-3, 0}; // x-intercept (-3,0)
    double B[2] = {0, 2}; // y-intercept (0,2)

    // Direction vector m = A - B
    double m[2];
    m[0] = A[0] - B[0];
    m[1] = A[1] - B[1];

    printf("Direction vector m = (%.2f, %.2f)\n", m[0], m[1]);

    // Normal vector n (perpendicular to m)
    double n[2];
    n[0] = -m[1]; // -y
    n[1] = m[0]; // x
```

C Code

```
 printf("Normal vector n = (%.2f, %.2f)\n", n[0], n[1]);

    // Point h (we take y-intercept B as reference point)
    double h[2] = {0, 2};

    // Equation: n^T * (x - h) = 0
    // Expanding: n^T * x = n^T * h
    double c = n[0]*h[0] + n[1]*h[1];

    printf("Equation of line: %.2fx + %.2fy = %.2f\n", n[0], n
        [1], c);

    return 0;
}
```

Python Code

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

# Line equation: -2x + 3y = 6
# Solve for y: y = (2x + 6)/3

# Define x values for plotting
x = np.linspace(-10, 10, 400)
y = (2 * x + 6) / 3

# Find intercepts
# X-intercept: set y = 0  -2x = 6  x = -3  A = (-3, 0)
# Y-intercept: set x = 0  3y = 6  y = 2  B = (0, 2)
A = (-3, 0)
B = (0, 2)
```

Python Code

```
  # Plot the line
plt.plot(x, y, label='Line: -2x + 3y = 6', color='blue')

# Mark the intercepts
plt.scatter(*A, color='red', zorder=5)
plt.scatter(*B, color='green', zorder=5)

# Annotate the points
plt.text(A[0]-1, A[1]-0.5, f'A {A}', color='red', fontsize=12)
plt.text(B[0]+0.2, B[1]+0.2, f'B {B}', color='green', fontsize
    =12)

# Axes lines
plt.axhline(0, color='black', linewidth=1)
plt.axvline(0, color='black', linewidth=1)
```

Python Code

```
# Graph settings
plt.title('Graph of the Line -2x + 3y = 6')
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.grid(True)
plt.legend()
plt.axis('equal')
plt.xlim(-10, 10)
plt.ylim(-10, 10)

# Show the plot
plt.show()
```

C and Python Code

```
 import ctypes
import os

# Load the shared object file
lib_path = os.path.abspath("liblineeq.so")
lib = ctypes.CDLL(lib_path)

# Define the function's argument types
lib.line_from_intercepts.argtypes = [ctypes.c_double, ctypes.
    c_double]

# Optional: Define the return type (void function, so None)
lib.line_from_intercepts.restype = None
```

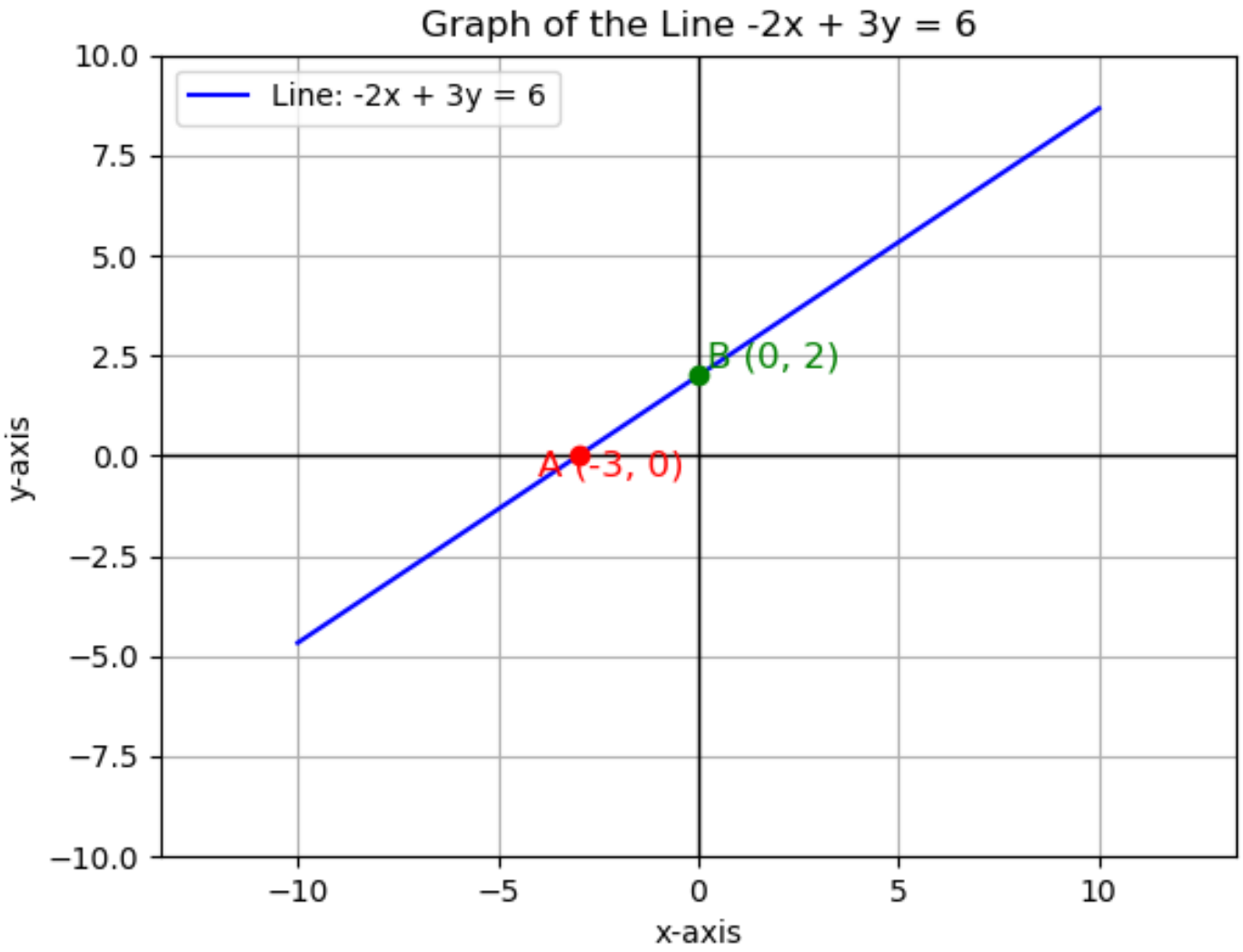
C and Python Code

```
 # Example intercepts
x_intercept = -3.0
y_intercept = 2.0

print("Calling C function from Python with:")
print(f" X-intercept = {x_intercept}")
print(f" Y-intercept = {y_intercept}\n")

# Call the function
lib.line_from_intercepts(x_intercept, y_intercept)
```

Plot


Graph of the Line -2x + 3y = 6
Line: -2x + 3y = 6
B (0, 2)
A (-3, 0)
y-axis
x-axis
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10
−5
0
5
10