4.2.15

BEERAM MADHURI - EE25BTECH11012

September 2025

Question

Find the direction and normal vectors of $y = 2x$.

given data

Line equation is $y = 2x$

finding direction and normal vectors of $y = 2x$

The line can be written as:

$$-2x + 1y = 0 \tag{1}$$

This equation can be expressed in terms of matrices as:

$$\mathbf{n}^\top \mathbf{x} = c \tag{2}$$

$$\mathbf{n}^\top = \begin{pmatrix} -2 & 1 \end{pmatrix} \tag{3}$$

$$\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} \tag{4}$$

$$c = 0 \tag{5}$$

where $\mathbf{n}$ is normal vector of the given line.
The direction vector is:

$$\mathbf{m} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}. \tag{6}$$

This is true because, if the direction vector is represented as

$$\mathbf{m} = \begin{pmatrix} 1 \\ m \end{pmatrix} \tag{7}$$

then the normal vector can be expressed as

$$\mathbf{n} = \begin{pmatrix} -m \\ 1 \end{pmatrix} \tag{8}$$

$$\mathbf{n}^\top \mathbf{m} = 0 \tag{9}$$

$$\begin{pmatrix} -2 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = 0 \tag{10}$$

Hence, normal vector $\mathbf{n} = \begin{pmatrix} -2 \\ 1 \end{pmatrix}$ and direction vector $\mathbf{m} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$.

Python Code

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

# --- 1. Vector Calculation ---
# The equation of the line is y = 2x.
# The standard form is Ax + By + C = 0, which is 2x - y = 0.
A = 2
B = -1

# The normal vector is <A, B>. It's perpendicular to the line.
normal_vector = np.array([A, B])
```

Python Code

```
# The direction vector is <-B, A>. It's parallel to the line.
direction_vector = np.array([-B, A])

print(f"Line Equation: y = 2x")
print("-" * 25)
print(f"Direction Vector: {tuple(direction_vector)}")
print(f"Normal Vector: {tuple(normal_vector)}")
```

Python Code

```
# Create a figure and a set of subplots
fig, ax = plt.subplots(figsize=(8, 8))

# Generate x values for our line
x_vals = np.linspace(-2.5, 2.5, 100)
# Calculate the corresponding y values for the line y = 2x
y_vals = 2 * x_vals

# Plot the main line
ax.plot(x_vals, y_vals, label='Line: y = 2x', color='blue',
    zorder=1)
```

Python Code

```
# Plot the vectors as arrows starting from the origin (0,0)
# The 'quiver' function is used to plot arrows.

# Plot the Direction Vector (green)
ax.quiver(0, 0, direction_vector[0], direction_vector[1],
          angles='xy', scale_units='xy', scale=1,
          color='green', label=f'Direction Vector: {tuple(
              direction_vector)}', zorder=2)

# Plot the Normal Vector (red)
ax.quiver(0, 0, normal_vector[0], normal_vector[1],
          angles='xy', scale_units='xy', scale=1,
          color='red', label=f'Normal Vector: {tuple(normal_vector
              )}', zorder=2)
```

Python Code

```
# Set the aspect ratio of the plot to be equal, so 90-degree
    angles look correct
ax.set_aspect('equal')

# Set the limits for the x and y axes
ax.set_xlim(-4, 4)
ax.set_ylim(-4, 4)

# Move the x and y axes to the center of the plot
ax.spines['left'].set_position('zero')
ax.spines['bottom'].set_position('zero')
ax.spines['right'].set_color('none')
ax.spines['top'].set_color('none')
```

Python Code

```
# Add a grid for better readability
ax.grid(True, linestyle='--')

# Add a title and a legend
ax.set_title("Line y=2x with its Direction and Normal Vectors",
    fontsize=14)
ax.legend(loc='upper left')

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

C Code

```c
#include <stdio.h>

int main() {
    // For the line equation y = 2x, which can be rewritten as 2x
        - 1y = 0.
    // This is in the general form Ax + By + C = 0.

    // Coefficients from the equation 2x - y = 0
    float A = 2.0;
    float B = -1.0;
```

C Code

```c
// The normal vector is given by the coefficients (A, B)
float normal_vector_x = A;
float normal_vector_y = B;

// The direction vector is perpendicular to the normal vector
    .
// A vector perpendicular to (A, B) is (-B, A).
float direction_vector_x = -B;
float direction_vector_y = A;
```

C Code

```c
// Print the results
    printf("For the line y = 2x (or %.1fx + %.1fy = 0):\n\n", A,
        B);

    printf("A Normal Vector is: (%.1f, %.1f)\n", normal_vector_x,
         normal_vector_y);
    printf("A Direction Vector is: (%.1f, %.1f)\n",
        direction_vector_x, direction_vector_y);

    return 0;}
```

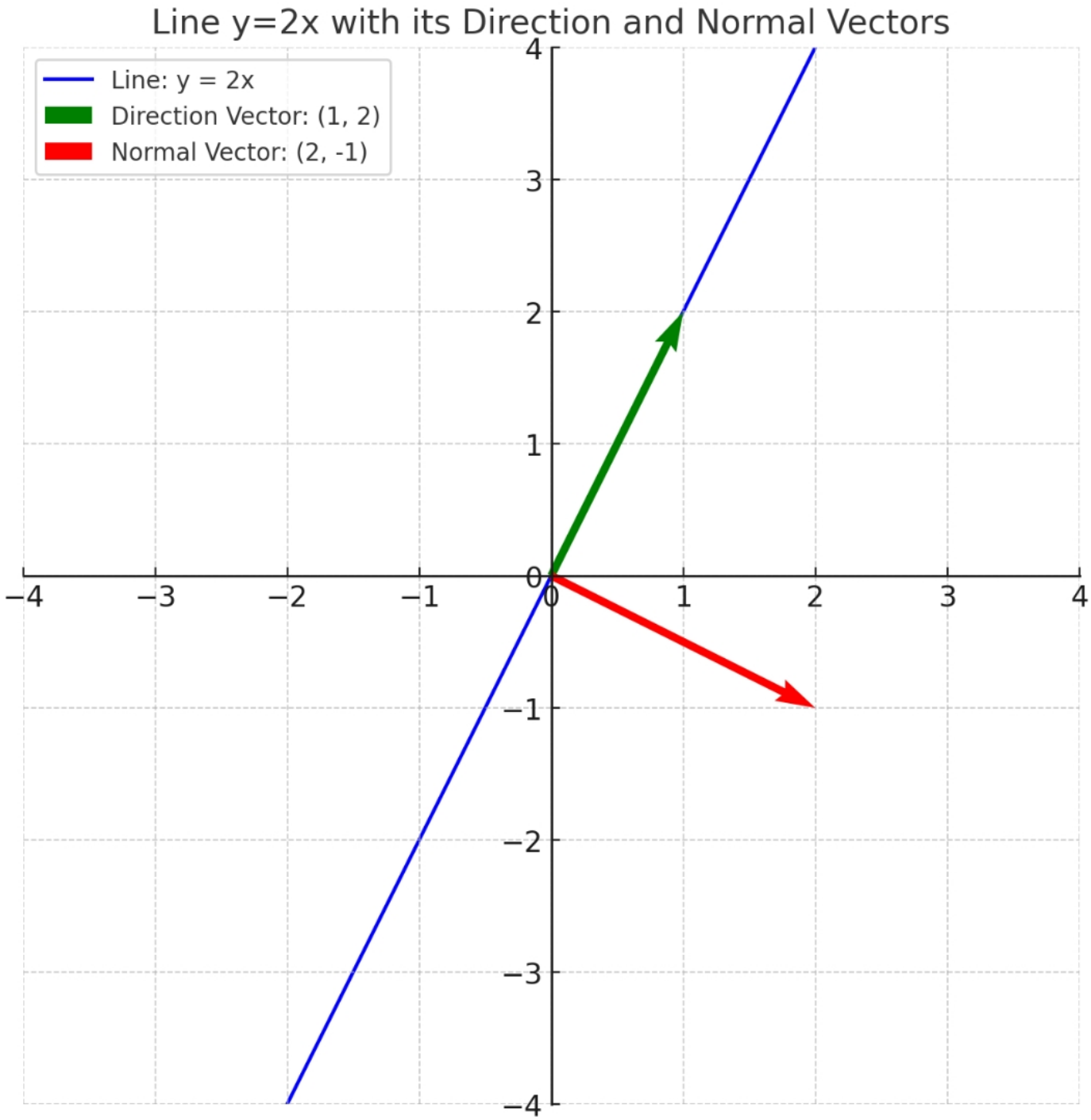
Python and C Code

```
import subprocess

# 1. Compile the C program
subprocess.run(["gcc", "line.c", "-o", "line"])

# 2. Run the compiled C program
result = subprocess.run(["./line"], capture_output=True, text=
    True)

# 3. Print the output from the C program
print(result.stdout)
```


Line y=2x with its Direction and Normal Vectors
Line: y = 2x
Direction Vector: (1, 2)
Normal Vector: (2, -1)
4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3
4