4.2.4

EE25BTECH11001 - Aarush Dilawri

September 29, 2025

Question:
Find the direction and normal vector for the line

$$x = 3y \tag{1}$$

Solution

The line can be written as:

$$x - 3y = 0 \tag{2}$$

This equation can be expressed in terms of matrices:
Let

$$\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} \tag{3}$$

$$\mathbf{n}^{\mathbf{T}} = \begin{pmatrix} 1 & -3 \end{pmatrix} \tag{4}$$

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

The line equation can be written as:

$$\mathbf{n}^{\mathbf{T}}\mathbf{x} = c \tag{6}$$

Where **n** is the normal vector of the given line

Direction Vector

The direction vector of the line can be found by observing the normal vector.

$$\mathbf{m} = \begin{pmatrix} 3 \\ 1 \end{pmatrix} \tag{7}$$

This is true because if the director vector is represented as

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

then the normal vector can be represented as

$$\mathbf{n} = \begin{pmatrix} -m \\ 1 \end{pmatrix} \tag{9}$$

This can be verified by the following equation:

$$\mathbf{n}^\mathbf{T}\mathbf{m} = 0 \tag{10}$$

Conclusion

$$\begin{pmatrix} 1 & -3 \end{pmatrix} \begin{pmatrix} 3 \\ 1 \end{pmatrix} = 0 \tag{11}$$

The normal vector of the line is

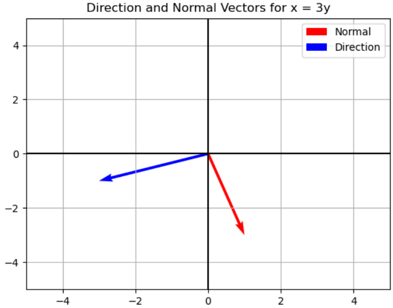
$$\mathbf{n} = \begin{pmatrix} 1 \\ -3 \end{pmatrix} \tag{12}$$

The director vector of the line is

$$\mathbf{m} = \begin{pmatrix} 3 \\ 1 \end{pmatrix} \tag{13}$$

Figure

From the figure, it is clearly verified that the theoretical solution matches with the computational solution.



C Code (code.c)

```
#include <stdio.h>

// Generalized dot product of two n-dimensional vectors
int dot_product(const int* a, const int* b, int n) {
    int sum = 0;
    for(int i = 0; i < n; i++) {
        sum += a[i] * b[i];
    }
    return sum;
}

// Check if two n-dimensional vectors are orthogonal
int is_orthogonal(const int* a, const int* b, int n) {
    return dot_product(a, b, n) == 0;
}
```

C Code (code.c)

```c
// Compute normal vector of a 2D line Ax + By = C
// Returns result in nvec array (size 2)
void normal_vector(int A, int B, int* nvec) {
    nvec[0] = A;
    nvec[1] = B;
}

// Compute direction vector of a 2D line Ax + By = C
// Returns result in dvec array (size 2)
void direction_vector(int A, int B, int* dvec) {
    dvec[0] = B;
    dvec[1] = -A;
}
```

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

# Generalized dot product
def dot_product(a, b):
    if len(a) != len(b):
        raise ValueError("Vectors must have same length")
    return sum(a[i]*b[i] for i in range(len(a)))

# Check orthogonality
def is_orthogonal(a, b):
    return dot_product(a, b) == 0

# Normal and direction vectors for a 2D line Ax + By = C
def normal_vector(A, B):
    return np.array([A, B])
```

Python Code (code.py)

```
def direction_vector(A, B):
    return np.array([B, -A])

# Example: x = 3y => 1*x - 3*y = 0
A, B = 1, -3
nvec = normal_vector(A, B)
dvec = direction_vector(A, B)

print("Normal-vector:", nvec)
print("Direction-vector:", dvec)
print("Orthogonal-check:", is_orthogonal(nvec, dvec))

# Plotting
origin = np.array([[0,0]])
```

Python Code (code.py)

```
plt.quiver(*origin.T, nvec[0], nvec[1], color='r', scale=1, scale_units='xy',
     angles='xy', label='Normal')
plt.quiver(*origin.T, dvec[0], dvec[1], color='b', scale=1, scale_units='xy',
      angles='xy', label='Direction')

plt.xlim(-5,5)
plt.ylim(-5,5)
plt.grid(True)
plt.axhline(0, color='black')
plt.axvline(0, color='black')
plt.legend()
plt.title('Direction and Normal Vectors for x = 3y')
plt.show()
```

Python Code (nativecode.py)

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

# Load shared library
lib = ctypes.CDLL('./code.so')

# Set argument and return types
lib.dot_product.argtypes = [ctypes.POINTER(ctypes.c_int), ctypes.
    POINTER(ctypes.c_int), ctypes.c_int]
lib.dot_product.restype = ctypes.c_int

lib.is_orthogonal.argtypes = [ctypes.POINTER(ctypes.c_int), ctypes.
    POINTER(ctypes.c_int), ctypes.c_int]
lib.is_orthogonal.restype = ctypes.c_int

lib.normal_vector.argtypes = [ctypes.c_int, ctypes.c_int, ctypes.POINTER(
```

Python Code (nativecode.py)

```
# Prepare arrays for 2D vectors
nvec = (ctypes.c_int * 2)()
dvec = (ctypes.c_int * 2)()

# Line: x = 3y => 1*x - 3*y = 0
A, B = 1, -3

lib.normal_vector(A, B, nvec)
lib.direction_vector(A, B, dvec)

print("Normal-vector:", list(nvec))
print("Direction-vector:", list(dvec))

# Plot vectors
origin = np.array([[0,0]])
```

Python Code (nativecode.py)

```
plt.quiver(*origin.T, nvec[0], nvec[1], color='r', scale=1, scale_units='xy',
    angles='xy', label='Normal')
plt.quiver(*origin.T, dvec[0], dvec[1], color='b', scale=1, scale_units='xy',
    angles='xy', label='Direction')

plt.xlim(-5,5)
plt.ylim(-5,5)
plt.grid(True)
plt.axhline(0, color='black')
plt.axvline(0, color='black')
plt.legend()
plt.title('Direction and Normal Vectors for x = 3y')
plt.show()
```