

Matgeo: 4-4.2-21

J. Adishesh Balaji
AI24BTECH11016

November 6, 2024

1 Problem

2 Solution

- Direction Vector
- Normal Vector

Problem Statement

Find the direction and normal vectors of the line.

$$F = \frac{9}{5}C + 32 \quad (3.1)$$

Direction Vector

The equation of line is given by

$$\begin{pmatrix} C \\ F \end{pmatrix} = \begin{pmatrix} C \\ \frac{9}{5}C + 32 \end{pmatrix} + \begin{pmatrix} 0 \\ 32 \end{pmatrix} \quad (3.2)$$

which can be compared with

$$\mathbf{x} = \mathbf{h} + k\mathbf{m} \quad (3.3)$$

Where \mathbf{h} is any point on the line and

$$\mathbf{m} = \begin{pmatrix} 1 \\ \frac{9}{5} \end{pmatrix} \quad (3.4)$$

is the direction vector

Normal Vector

The normal vector can be found as follows

$$\mathbf{m}^T \mathbf{n} = 0 \quad (3.5)$$

$$\mathbf{n}^T \mathbf{x} = \mathbf{n}^T \mathbf{h} + k \mathbf{n}^T \mathbf{m} \quad (3.6)$$

$$\mathbf{n} = \begin{pmatrix} -m \\ 1 \end{pmatrix} \quad (3.7)$$

Hence, the normal vector

$$\mathbf{n} = \begin{pmatrix} -\frac{9}{5} \\ 1 \end{pmatrix} \quad (3.8)$$

The code in `/bmrsgn/asgn1/codes/line.py` verifies (3.4) and (3.8)

C-code to generate Data I

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  // Function to generate points on the line  $F = (9/5)C + 32$ 
5  void point_gen(FILE *fptr, double c1, double f1, double c2, double f2, int
↪ num_points) {
6      for (int i = 0; i <= num_points; i++) {
7          double t = (double)i / num_points;
8          double c = c1 + t * (c2 - c1); // Linear interpolation for C
9          double f = f1 + t * (f2 - f1); // Linear interpolation for F
10         fprintf(fptr, "%lf,%lf\n", c, f);
11     }
12 }
13
14 int main() {
15     // Define two points on the line  $F = (9/5)C + 32$ 
16     double c1 = -50.0, f1 = (9.0 / 5.0) * c1 + 32.0; // First point
↪     (C=-50, F)
17     double c2 = 100.0, f2 = (9.0 / 5.0) * c2 + 32.0; // Second point
↪     (C=100, F)
```

C-code to generate Data II

```
18
19 // Open the file to save the points
20 FILE *fptr = fopen("line_points.txt", "w");
21 if (fptr == NULL) {
22     printf("Error opening file!\n");
23     return 1;
24 }
25
26 // Generate points on the line
27 point_gen(fptr, c1, f1, c2, f2, 63); // Generate 63 points on the
    ↪ line
28
29 // Normal vector generation
30 // The slope of the line is 9/5, so the slope of the normal is -5/9.
31 double c_normal = c1, f_normal = f1; // Take the first point for
    ↪ normal vector
32 double norm_slope = -5.0 / 9.0; // Slope of the normal line
33 double norm_length = 50.0; // Arbitrary length for the
    ↪ normal line
34
```

C-code to generate Data III

```
35     double c_norm_end = c_normal + norm_length;
36     double f_norm_end = f_normal + norm_slope * (c_norm_end - c_normal);
37
38     // Generate points on the normal line
39     point_gen(fp_ptr, c_normal, f_normal, c_norm_end, f_norm_end, 20);
40
41     // Close the file
42     fclose(fp_ptr);
43
44     printf("Points on the line and normal vector saved to  

45     ↪ line_points.txt\n");
46
47     return 0;
48 }
49
```


Python code to plot graph I

```
1         import sys # for path to external scripts
2 sys.path.insert(0, '/home/adishesh-balaji/github/matgeo/codes/CoordGeo')
   ↪ # path to my scripts
3 import numpy as np
4 import numpy.linalg as LA
5 import matplotlib.pyplot as plt
6
7 # local imports
8 from line.funcs import *
9 from triangle.funcs import *
10 from conics.funcs import circ_gen
11
12 # Function to read points from the txt file
13 def read_points_from_file(filename):
14     c_values = []
15     f_values = []
16
17     with open(filename, 'r') as file:
18         for line in file:
19             # Split the line by comma and convert to float
```

Python code to plot graph II

```
20         c, f = map(float, line.strip().split(','))
21         c_values.append(c)
22         f_values.append(f)
23
24     return np.array(c_values), np.array(f_values)
25
26     # Function to plot the line and its normal vector
27 def plot_line_and_normal(c_values, f_values):
28     # Plot the line as a dotted line
29     plt.plot(c_values, f_values, label=r'$F = \frac{9}{5}C + 32$',
30             ↪ color='blue', linestyle='dotted')
31
32     # Calculate the midpoint of the line
33     midpoint_index = len(c_values) // 2
34     c_mid = c_values[midpoint_index]
35     f_mid = f_values[midpoint_index]
36
37     # Midpoint as the starting point for the normal vector
38     A_mid = np.array([c_mid, f_mid])
```

Python code to plot graph III

```
39     # Direction vector (slope = 9/5)
40     m = np.array([1, 9/5])
41
42     # Normal vector (slope = -5/9)
43     n = np.array([-9/5, 1])
44
45     # Plot the main line using parametric form (starting at A = [0, 32])
46     main_line_points = line_dir_pt(m, np.array([0, 32]), -50, 50)
47     plt.plot(main_line_points[0, :], main_line_points[1, :], color='blue',
48     ↪     linestyle='dotted')
49
50     # Plot the normal vector emerging from the midpoint
51     normal_points = line_dir_pt(n, A_mid, -20, 20) # Adjust the range of
52     ↪ the normal vector as needed
53     plt.plot(normal_points[0, :], normal_points[1, :], label='Normal
54     ↪     Vector', color='red', linestyle = 'dotted')
55
56     # Set labels
57     plt.xlabel('C (Celsius)')
58     plt.ylabel('F (Fahrenheit)')
```

Python code to plot graph IV

```
56     plt.legend()
57
58     # Set limits
59     plt.xlim(-60, 110)
60     plt.ylim(-70, 220)
61
62     # Add grid
63     plt.grid(True)
64     plt.axhline(0, color='black', linewidth=0.5)
65     plt.axvline(0, color='black', linewidth=0.5)
66
67     # Show the plot
68     plt.gca().set_aspect('equal', adjustable='box')
69     plt.show()
70
71     # Main code execution
72     if __name__ == "__main__":
73         filename = "line_points.txt"
74         c_values, f_values = read_points_from_file(filename)
75         plot_line_and_normal(c_values, f_values)
```

Python code to plot graph V

76