2.5.30

Vivek K Kumar - EE25BTECH11062

September 6, 2025

Question

If the two lines

$$L_1 : x = 5, \frac{y}{3-\alpha} = \frac{z}{-2} \text{ and} \tag{1}$$

$$L_2 : x = 2, \frac{y}{-1} = \frac{z}{2-\alpha} \tag{2}$$

are perpendicular, then the value of α is _____

Variables used

Name	Point
$\mathbf{m_1}$ (Direction vector of L_1)	$\begin{pmatrix} 0 \\ 3-\alpha \\ -2 \end{pmatrix}$
$\mathbf{m_2}$ (Direction vector of L_2)	$\begin{pmatrix} 0 \\ -1 \\ 2-\alpha \end{pmatrix}$

Table: Variables Used

Solution

The lines can be represented as

$$\mathbf{x} = \begin{pmatrix} 5 \\ 0 \\ 0 \end{pmatrix} + \kappa_1 \mathbf{m_1} \tag{3}$$

$$= \begin{pmatrix} 5 \\ 0 \\ 0 \end{pmatrix} + \kappa_1 \begin{pmatrix} 0 \\ 3-\alpha \\ -2 \end{pmatrix} \tag{4}$$

and

$$\mathbf{x} = \begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} + \kappa_2 \mathbf{m_2} \tag{5}$$

$$= \begin{pmatrix} 2 \\ 0 \\ 0 \end{pmatrix} + \kappa_2 \begin{pmatrix} 0 \\ -1 \\ 2-\alpha \end{pmatrix} \tag{6}$$

Solution

As the given lines are perpendicular, their direction vectors follow the relation:

$$\mathbf{m_1}^T\mathbf{m_2} = 0 \tag{7}$$

$$\begin{pmatrix} 0 & 3-\alpha & -2 \end{pmatrix} \begin{pmatrix} 0 \\ -1 \\ 2-\alpha \end{pmatrix} = 0 \tag{8}$$

$$3\alpha - 7 = 0 \tag{9}$$

$$\text{which gives } \alpha = \frac{7}{3} \tag{10}$$

$$\text{and } \mathbf{m_1} = \begin{pmatrix} 0 \\ \frac{2}{3} \\ -2 \end{pmatrix}, \mathbf{m_2} = \begin{pmatrix} 0 \\ -1 \\ \frac{-1}{3} \end{pmatrix} \tag{11}$$

Python - Importing libraries and checking system

```
import sys
import numpy as np
import numpy.linalg as LA
import matplotlib.pyplot as plt
import matplotlib.image as mpimg

from libs.line.funcs import *
from libs.triangle.funcs import *
from libs.conics.funcs import circ_gen

import subprocess
import shlex

print('Using termux?(y/n)')
y = input()
```

Python - Checking if 2 lines are perpendicular

```
m1 = np.array([0, 2/3, -2]).reshape(-1, 1)
r1 = np.array([5, 0, 0]).reshape(-1, 1)
m2 = np.array([0, -1, -1/3]).reshape(-1,1)
r2 = np.array([2, 0, 0]).reshape(-1, 1)
O = np.zeros(3).reshape(-1, 1)
if(m1.T@m2 == 0):
    print('The two lines are perpendicular')
else:
    print('The two lines are not perpendicular')
```

Python - Generating points and plotting

```
p_m1 = line_gen(r1-8*m1, r1+8*m1)
p_m2 = line_gen(r2-8*m2, r2+8*m2)

fig = plt.figure()
ax = fig.add_subplot(111, projection = '3d')

ax.plot(p_m1[0, :], p_m1[1, :], p_m1[2, :], label = 'Line L1')
ax.plot(p_m2[0, :], p_m2[1, :], p_m2[2, :], label = 'Line L2')
```

Python - Labelling points

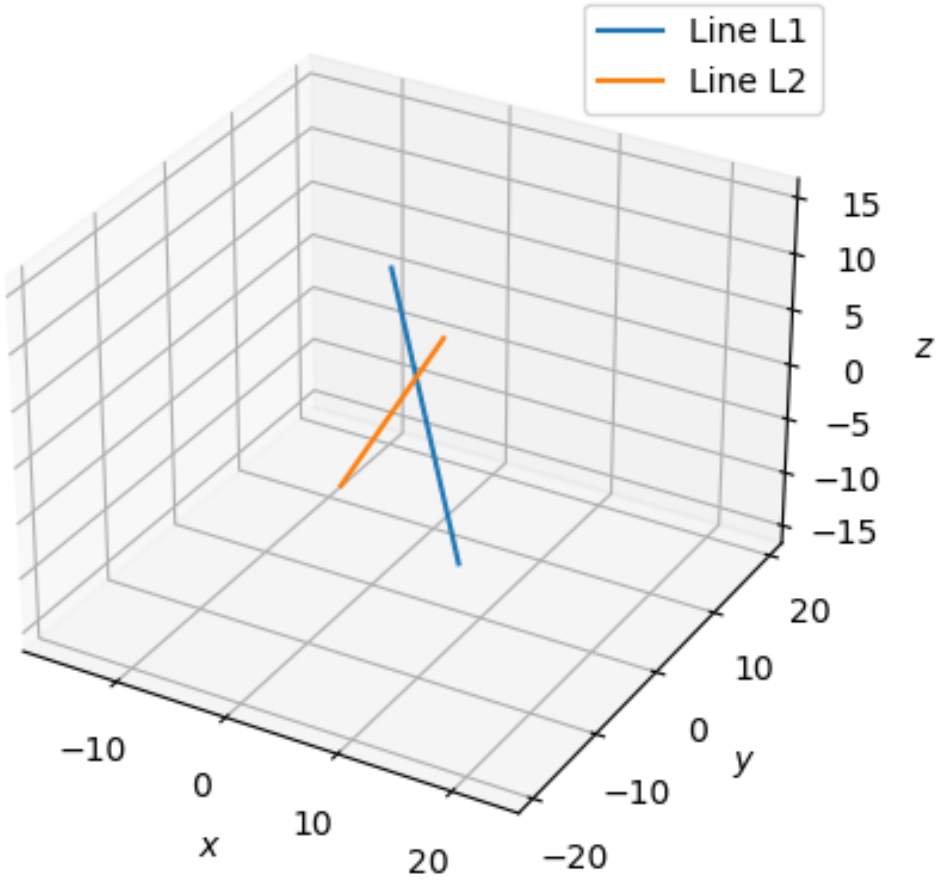
```
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_zlabel('$z$')
ax.legend(loc='best')
ax.grid(True)
ax.axis('equal')
```

Python - Saving figure and opening it

```
fig.savefig('../figs/fig.png')
print('Saved figure to ../figs/fig.png')

if(y == 'y'):
    subprocess.run(shlex.split('termux-open ../figs/fig.png'))
else:
    subprocess.run(["open", "../figs/fig.png"])
```

Plot-Using only Python



C Code (0) - Importing libraries

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <unistd.h>
#include "libs/matfun.h"
#include "libs/geofun.h"
```

C Code (1) - Function to Generate Points on a Line

```c
void point_gen(FILE *p_file, double **A, double **B, int rows,
    int cols, int npts){
    for(int i = 0; i <= npts; i++){
     double **output = Matadd(A, Matscale(Matsub(B, A, rows, cols
         ), rows, cols, (double)i/npts), rows, cols);
     fprintf(p_file, "%lf, %lf, %lf\n", output[0][0], output
         [1][0], output[2][0]);
     freeMat(output, rows);
    }
}
```

C Code (2) - Function to write points b/w given point and origin to a file

```
int check_perpendicularity(double **p1, double **p2, int m, int n
    );

int write_points(double x1, double y1, double z1, double x2,
    double y2, double z2, double x3, double y3, double z3, double
     x4, double y4, double z4, int npts){
    int m = 3;
    int n = 1;

    double **R = createMat(m, n);
    double **O = createMat(m, n);
    double **T = createMat(m, n);
    double **S = createMat(m, n);

    R[0][0] = x2;
    R[1][0] = y2;
    R[2][0] = z2;
```

C Code (2) - Function to write points b/w given point and origin to a file

```
    O[0][0] = x1;
    O[1][0] = y1;
    O[2][0] = z1;

    T[0][0] = x4;
    T[1][0] = y4;
    T[2][0] = z4;

    S[0][0] = x3;
    S[1][0] = y3;
    S[2][0] = z3;
```

C Code (2) - Function to write points b/w given point and origin to a file

```
    FILE *p_file;
    FILE *p_file_2;
    p_file = fopen("plot.dat", "w");
    p_file_2 = fopen("plot2.dat", "w");
    if(p_file == NULL || p_file_2 == NULL){
        printf("Error opening one of the data files\n");
    }
    point_gen(p_file, O, R, m, n, npts);
    point_gen(p_file_2, S, T, m, n, npts);
    int k = check_perpendicularity(Matsub(R, O, m, n), Matsub(T,
        S, m, n), m, n);
    freeMat(R, m);
    freeMat(O, m);
    freeMat(T, m);
    freeMat(S, m);
    fclose(p_file);
    fclose(p_file_2);
```

C Code (3) - Checking Perpendicularity

```c
int check_perpendicularity(double **p1, double **p2, int m, int n
    ){
    return Matmul(transposeMat(p1, m, n), p2, n, m, n)[0][0];
}
```

Python Code (0) - Importing libraries and checking system

```
import numpy as np
import matplotlib.pyplot as plt
import ctypes
import os
import sys
import subprocess

print('Using termux? (y/n)')
termux = input()
```

Python Code (1) - Using Shared Object

```
lib_path = os.path.join(os.path.dirname(__file__), 'plot.so')
my_lib = ctypes.CDLL(lib_path)
my_lib.write_points.argtypes = [ctypes.c_double, ctypes.c_double,
     ctypes.c_double, ctypes.c_double, ctypes.c_double, ctypes.
    c_double, ctypes.c_double, ctypes.c_double, ctypes.c_double,
    ctypes.c_double, ctypes.c_double, ctypes.c_double, ctypes.
    c_int]
my_lib.write_points.restype = ctypes.c_int
r1 = np.array([5, 0, 0])
r2 = np.array([2, 0, 0])
m1 = np.array([0, 2/3, -2])
m2 = np.array([0, -1, -1/3])
p1 = r1 - 8*m1
p2 = r1 + 8*m1
p3 = r2 - 8*m2
p4 = r2 + 8*m2
k = my_lib.write_points(p1[0], p1[1], p1[2], p2[0], p2[1], p2[2],
     p3[0], p3[1], p3[2], p4[0], p4[1], p4[2], 20000)
```

Python Code (2) - Loading points and checking perpendicularity

```
if k == 0:
    print('The given lines are perpendicular')
else:
    print('The given lines are not perpendicular')

points = np.loadtxt('plot.dat', delimiter=',', usecols = (0,1, 2)
    )
points2 = np.loadtxt('plot2.dat', delimiter=',', usecols = (0,1,
    2))

x = points[:, 0]
y = points[:, 1]
z = points[:, 2]

x2 = points2[:, 0]
y2 = points2[:, 1]
z2 = points2[:, 2]
```

Python Code (3) - Plotting points

```
fig = plt.figure()
ax = fig.add_subplot(111, projection = '3d')
ax.plot(x, y, z, label = 'Line L1')
ax.plot(x2, y2, z2, label = 'Line L2')

ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_zlabel('$z$')
ax.legend(loc='best')
ax.grid()
ax.axis('equal')

fig.savefig('../figs/fig2.png')
print('Saved figure to ../figs/fig2.png')
```

Python Code (4) - Saving plot and opening it

```
fig.savefig('../figs/fig2.png')
print('Saved figure to ../figs/fig2.png')

if(termux == 'y'):
    subprocess.run(shlex.split('termux-open ../figs/fig2.png'))
else:
    subprocess.run(["open", "../figs/fig2.png"])
```

Plot-Using Both C and Python