2.10.54

Vishwambhar - EE25BTECH11025

5th september, 2025

Question

let $\mathbf{a}, \mathbf{b}, \mathbf{c}$ be unit vectors such that $\mathbf{a} + \mathbf{b} + \mathbf{c} = \mathbf{0}$. Which of the following are correct?

1. $\mathbf{a} \times \mathbf{b} = \mathbf{b} \times \mathbf{c} = \mathbf{c} \times \mathbf{a} = \mathbf{0}$
2. $\mathbf{a} \times \mathbf{b} = \mathbf{b} \times \mathbf{c} = \mathbf{c} \times \mathbf{a} \neq \mathbf{0}$
3. $\mathbf{a} \times \mathbf{b} = \mathbf{b} \times \mathbf{c} = \mathbf{a} \times \mathbf{c} \neq \mathbf{0}$
4. $\mathbf{a} \times \mathbf{b}, \mathbf{b} \times \mathbf{c}, \mathbf{c} \times \mathbf{a}$ are mutually perpendicular.

Given:

$$\mathbf{a} + \mathbf{b} + \mathbf{c} = 0 \tag{1}$$

$$\mathbf{c} = \begin{pmatrix} \mathbf{a} & \mathbf{b} \end{pmatrix} \begin{pmatrix} -1 \\ -1 \end{pmatrix} \tag{2}$$

$$\tag{3}$$

Assuming 2D space

This $\mathbf{c}$ lies in span of $\mathbf{a}, \mathbf{b}$.

Since $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are all in 2D space, if all three are non-zero unit vectors satisfying this relation, they must be linearly dependent.

Therefore, the 2×2 matrix $\begin{pmatrix} \mathbf{a} & \mathbf{b} \end{pmatrix}$ cannot be invertible.

$$\left| \begin{pmatrix} \mathbf{a} & \mathbf{b} \end{pmatrix} \right| = 0 \tag{4}$$

Singular matrix

So the matrix is singular.
In 2D, norm is defined by the determinant:

$$\|\mathbf{a} \times \mathbf{b}\| = \left|\begin{pmatrix} \mathbf{a} & \mathbf{b} \end{pmatrix}\right| \tag{5}$$

So if $\left|\begin{pmatrix} \mathbf{a} & \mathbf{b} \end{pmatrix}\right| = 0$, then

$$\mathbf{a} \times \mathbf{b} = 0 \tag{6}$$

Similarly, we can show the same for the vectors **a** and **b**.
Thus, the correct option is (1):

$$\mathbf{a} \times \mathbf{b} = \mathbf{b} \times \mathbf{c} = \mathbf{c} \times \mathbf{a} = \mathbf{0} \tag{7}$$

C Code

```
#include <stdio.h>
#include <math.h>
typedef struct {double x, y, z;} Vector;
Vector cross(Vector a, Vector b) {
    Vector result;
    result.x = a.y * b.z - a.z * b.y;
    result.y = a.z * b.x - a.x * b.z;
    result.z = a.x * b.y - a.y * b.x;
    return result;}
double dot(Vector a, Vector b) {
    return a.x * b.x + a.y * b.y + a.z * b.z;}
void check_conditions(Vector a, Vector b, Vector c, int *results)
     {
    Vector ab = cross(a, b);
    Vector bc = cross(b, c);
    Vector ca = cross(c, a);
    results[0] = (ab.x==0 && ab.y==0 && ab.z==0 &&
                  bc.x==0 && bc.y==0 && bc.z==0 &&
                  ca.x==0 && ca.y==0 && ca.z==0);
```

C Code

```
    results[1] = ((ab.x==bc.x && ab.y==bc.y && ab.z==bc.z) &&
                 (bc.x==ca.x && bc.y==ca.y && bc.z==ca.z) &&
                 !(ab.x==0 && ab.y==0 && ab.z==0));
    Vector ac = cross(a, c);
    results[2] = ((ab.x==bc.x && ab.y==bc.y && ab.z==bc.z) &&
                 (bc.x==ac.x && bc.y==ac.y && bc.z==ac.z) &&
                 !(ab.x==0 && ab.y==0 && ab.z==0));
    results[3] = (fabs(dot(ab, bc)) < 1e-9 &&
                 fabs(dot(bc, ca)) < 1e-9 &&
                 fabs(dot(ca, ab)) < 1e-9);}
void out_data(double *points){
    double A[3] = {1, 0, 0};
    double B[3] = {-0.5, sqrt(3)/2, 0};
    double C[3] = {-0.5, -sqrt(3)/2, 0};
    points[0] = A[0];points[1] = A[1];points[2] = A[2];
    points[3] = B[0];points[4] = B[1];points[5] = B[2];
    points[6] = C[0];points[7] = C[1];points[8] = C[2];}
```

Python Code 1

```
import ctypes as ct
import math
lib = ct.CDLL("./problem.so")
class Vector(ct.Structure):
    _fields_ = [("x", ct.c_double), ("y", ct.c_double), ("z", ct.
        c_double)]
lib.check_conditions.argtypes = [Vector, Vector, Vector, ct.
    POINTER(ct.c_double)]
lib.check_conditions.restype = None
points = ct.c_double*9
lib.out_data.argtypes = [ct.POINTER(ct.c_double)]
data = points()
lib.out_data(data)
```

Python Code 1

```python
a = Vector(data[0], data[1], data[2])
b = Vector(data[3], data[4], data[5])
c = Vector(data[6], data[7], data[8])
results = (ct.c_double * 4)()
lib.check_conditions(a, b, c, results)
options = ['a', 'b', 'c', 'd']
for i, res in enumerate(results):
    print(f"Option {options[i]}: {'True' if res else 'False'}")
def send_data():
    return data[0], data[1], data[3], data[4], data[6], data[7]
```

Python Code 2

```
import numpy as np
import matplotlib.pyplot as plt
from call import send_data
Ax, Ay, Bx, By, Cx, Cy = send_data()
a = np.array([Ax, Ay])
b = np.array([Bx, By])
c = np.array([Cx, Cy])
plt.figure()
xs = [a[0], b[0], c[0], a[0]]
ys = [a[1], b[1], c[1], a[1]]
plt.plot(xs, ys, 'k-', label='Triangle (a,b,c)')
O = np.array([0, 0])
plt.plot([O[0], a[0]], [O[1], a[1]], 'r-', label='a')
plt.plot([O[0], b[0]], [O[1], b[1]], 'g-', label='b')
```

Python Code 2

```python
plt.plot([O[0], c[0]], [O[1], c[1]], 'b-', label='c')
plt.scatter([a[0], b[0], c[0]], [a[1], b[1], c[1]], c=['r','g','b'])
plt.text(a[0], a[1], 'a', fontsize=12)
plt.text(b[0], b[1], 'b', fontsize=12)
plt.text(c[0], c[1], 'c', fontsize=12)
plt.axis('equal')
plt.grid(True)
plt.legend()
plt.title("Triangle of unit vectors (a+b+c=0)")
plt.savefig("../figs/plot.png")
plt.show()
```

Plot

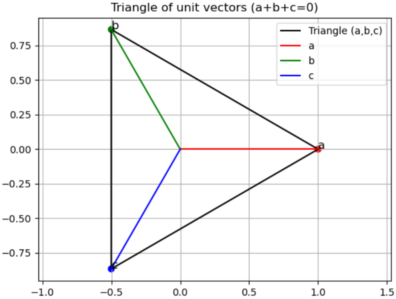

Figure: Plot of vectors $\mathbf{a}, \mathbf{b}$ and $\mathbf{c}$