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

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);
```

C Code

```c
    // Option (a): all cross products = 0
    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);
    // Option (b): all equal and nonzero
    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));
    // Option (c): ab = bc = a*c != 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));
    // Option (d): ab, bc, ca mutually perpendicular (dot = 0)
    results[3] = (fabs(dot(ab, bc)) < 1e-9 &&
                  fabs(dot(bc, ca)) < 1e-9 &&
                  fabs(dot(ca, ab)) < 1e-9);
}
```

Python Code 1

```
import ctypes
from ctypes import c_double, c_int, POINTER
import math

# Load the shared object
lib = ctypes.CDLL("./libvectors.so")

# Define Vector struct
class Vector(ctypes.Structure):
    _fields_ = [("x", c_double), ("y", c_double), ("z", c_double)
        ]

# Define function signature
lib.check_conditions.argtypes = [Vector, Vector, Vector, POINTER(
    c_int)]
lib.check_conditions.restype = None
```

Python Code 1

```python
# Example unit vectors (satisfying a+b+c=0)
a = Vector(1, 0, 0)
b = Vector(-0.5, math.sqrt(3)/2, 0)
c = Vector(-0.5, -math.sqrt(3)/2, 0)

# Results array
results = (c_int * 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'}")
```

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

# Add local path for custom geometry functions
sys.path.insert(0, '/home/ganachari-vishwmabhar/Downloads/codes/CoordGeo')

# Import the given helper functions
from line.funcs import *
from triangle.funcs import *

# Define the vectors a, b, c (unit vectors with a+b+c=0)
a = np.array([1, 0]) # Along x-axis
b = np.array([-0.5, np.sqrt(3)/2]) # 120Â° rotated
c = np.array([-0.5, -np.sqrt(3)/2]) # 240Â° rotated
```

Python Code 2

```
# Plot the triangle formed by a, b, c
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)')

# Mark the vectors from origin
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')
plt.plot([O[0], c[0]], [O[1], c[1]], 'b-', label='c')
```

Python Code 2

```python
# Mark points
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

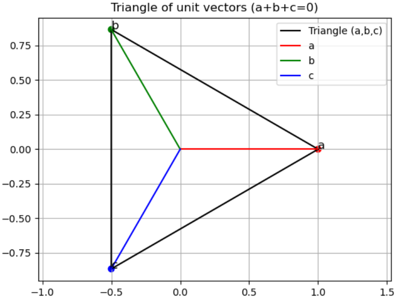

Figure: Plot of vectors $\mathbf{a}, \mathbf{b}$ and $\mathbf{c}$