2.10.42

BEERAM MADHURI - EE25BTECH11012

September 2025

Question

If $\mathbf{a}, \mathbf{b}$ and $\mathbf{c}$ are unit coplanar vectors, then the scalar triple product

$$\begin{bmatrix} 2\mathbf{a} - \mathbf{b} & 2\mathbf{b} - \mathbf{c} & 2\mathbf{c} - \mathbf{a} \end{bmatrix} =$$

finding scalar triple product of

$$\begin{bmatrix}2\mathbf{a}-\mathbf{b} & 2\mathbf{b}-\mathbf{c} & 2\mathbf{c}-\mathbf{a}\end{bmatrix}$$

$$\mathbf{B} = \begin{pmatrix}2\mathbf{a}-\mathbf{b} & 2\mathbf{b}-\mathbf{c} & 2\mathbf{c}-\mathbf{a}\end{pmatrix} = \begin{pmatrix}\mathbf{a} & \mathbf{b} & \mathbf{c}\end{pmatrix}\begin{pmatrix}2 & 0 & -1\\ -1 & 2 & 0\\ 0 & -1 & 2\end{pmatrix} \tag{1}$$

Since **a**, **b**, **c** are coplanar,

$$\begin{vmatrix}\mathbf{a} & \mathbf{b} & \mathbf{c}\end{vmatrix} = 0 \tag{2}$$

$$\begin{vmatrix}2\mathbf{a}-\mathbf{b} & 2\mathbf{b}-\mathbf{c} & 2\mathbf{c}-\mathbf{a}\end{vmatrix} = \begin{vmatrix}\mathbf{a} & \mathbf{b} & \mathbf{c}\end{vmatrix}\begin{vmatrix}2 & 0 & -1\\ -1 & 2 & 0\\ 0 & -1 & 2\end{vmatrix} = 0 \tag{3}$$

Hence, the value of $\begin{bmatrix}2\mathbf{a}-\mathbf{b} & 2\mathbf{b}-\mathbf{c} & 2\mathbf{c}-\mathbf{a}\end{bmatrix} = 0.$

Proof of $\begin{bmatrix} \mathbf{a} & \mathbf{b} & \mathbf{c} \end{bmatrix}$ is singular:

Given $\mathbf{a}$,$\mathbf{b}$,$\mathbf{c}$ are coplanar
plane equation of the plane through $\mathbf{a}$,$\mathbf{b}$,$\mathbf{c}$ be

$$\mathbf{n}^\top \mathbf{r} = 0 \tag{4}$$

where $\mathbf{n}$ is normal to plane

$$\mathbf{n}^\top \mathbf{a} = 0 \tag{5}$$

$$\mathbf{n}^\top \mathbf{b} = 0 \tag{6}$$

$$\mathbf{n}^\top \mathbf{c} = 0 \tag{7}$$

let $M = \begin{bmatrix} \mathbf{a} & \mathbf{b} & \mathbf{c} \end{bmatrix}$

$$\mathbf{n}^\top M = 0^\top \tag{8}$$

For a homogeneous linear system

$$\mathbf{n}^\top M = 0^\top, \quad \mathbf{n} \neq 0 \tag{9}$$

M must be singular.

$$\therefore \begin{bmatrix} \mathbf{a} & \mathbf{b} & \mathbf{c} \end{bmatrix} \text{ is singular}$$

Hence proved

Python Code

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

# Create a 3D plot
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')
```

Python Code

```
# --- 1. Define three unit coplanar vectors (a, b, c) ---
# For simplicity, we'll place them on the x-y plane (z=0).
# These are just example vectors that satisfy the conditions.
a = np.array([1, 0, 0])
b = np.array([np.cos(np.pi/3), np.sin(np.pi/3), 0]) # 60 degrees
    from a
c = np.array([np.cos(2*np.pi/3), np.sin(2*np.pi/3), 0]) # 120
    degrees from a
v1 = 2 * a - b
v2 = 2 * b - c
v3 = 2 * c - a
```

```
# --- 3. Plot the vectors ---
origin = np.array([0, 0, 0])
ax.quiver(*origin, *a, color='r', label='a', arrow_length_ratio
    =0.1)
ax.quiver(*origin, *b, color='g', label='b', arrow_length_ratio
    =0.1)
ax.quiver(*origin, *c, color='b', label='c', arrow_length_ratio
    =0.1)
ax.quiver(*origin, *v1, color='orange', label='v1 = 2a - b',
    arrow_length_ratio=0.1, linestyle='--')
ax.quiver(*origin, *v2, color='purple', label='v2 = 2b - c',
    arrow_length_ratio=0.1, linestyle='--')
ax.quiver(*origin, *v3, color='cyan', label='v3 = 2c - a',
    arrow_length_ratio=0.1, linestyle='--')
```

Python Code

```python
# --- 4. Add labels for each vector ---
ax.text(*(a*1.1), 'a', color='r', fontsize=12)
ax.text(*(b*1.1), 'b', color='g', fontsize=12)
ax.text(*(c*1.1), 'c', color='b', fontsize=12)
ax.text(*(v1*1.05), 'v1', color='orange', fontsize=12)
ax.text(*(v2*1.05), 'v2', color='purple', fontsize=12)
ax.text(*(v3*1.05), 'v3', color='cyan', fontsize=12)
```

Python Code

```python
# --- 5. Visualize the plane ---
# Create a grid for the plane at z=0
xx, yy = np.meshgrid(np.linspace(-3, 3, 2), np.linspace(-3, 3, 2)
    )
zz = np.zeros_like(xx)
ax.plot_surface(xx, yy, zz, alpha=0.1, color='gray')
```

```python
# --- 6. Set plot aesthetics ---
ax.set_xlim([-3, 3])
ax.set_ylim([-3, 3])
ax.set_zlim([-3, 3])
ax.set_xlabel('X axis')
ax.set_ylabel('Y axis')
ax.set_zlabel('Z axis')
ax.set_title('Visualization of Coplanar Vectors')
ax.legend()
ax.grid(True)
# Set a viewing angle for better perspective
ax.view_init(elev=25, azim=30)
plt.show()
```

C Code

```c
#include <stdio.h>

typedef struct {
    double x, y, z;
} Vector;
Vector createVector(double x, double y, double z) {
    Vector v = {x, y, z};
    return v;
}
Vector subtract(Vector u, Vector v) {
    return createVector(u.x - v.x, u.y - v.y, u.z - v.z);
}
Vector scale(Vector u, double k) {
    return createVector(k*u.x, k*u.y, k*u.z);
}
```

C Code

```
   Vector cross(Vector u, Vector v) {
    return createVector(
        u.y*v.z - u.z*v.y,
        u.z*v.x - u.x*v.z,
        u.x*v.y - u.y*v.x
    );
}
double dot(Vector u, Vector v) {
    return u.x*v.x + u.y*v.y + u.z*v.z;
}
double triple(Vector u, Vector v, Vector w) {
    return dot(u, cross(v, w));
}
```

C Code

```c
Vector twominus(Vector a, Vector b) {
    return subtract(scale(a, 2), b); }
double computeX(Vector a, Vector b, Vector c) {
    Vector v1 = twominus(a, b);
    Vector v2 = twominus(b, c);
    Vector v3 = twominus(c, a);
    return triple(v1, v2, v3);}
__attribute__((visibility("default")))
double computeX_py(double ax, double ay, double az,
                   double bx, double by, double bz,
                   double cx, double cy, double cz) {
    Vector a = createVector(ax, ay, az);
    Vector b = createVector(bx, by, bz);
    Vector c = createVector(cx, cy, cz);
    return computeX(a, b, c);}
```

Python and C Code

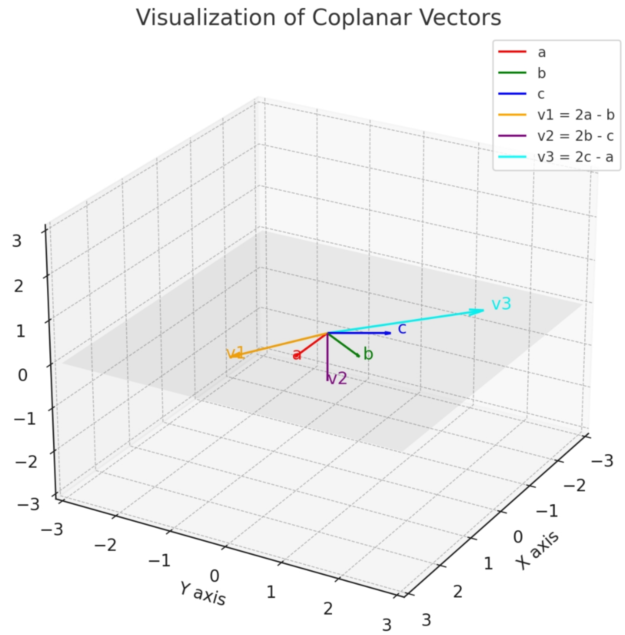
```
import ctypes

lib = ctypes.CDLL("./libscalartp.so")

lib.computeX_py.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]
lib.computeX_py.restype = ctypes.c_double

a = (1.0, 0.0, 0.0)
b = (0.0, 1.0, 0.0)
c = (0.6,0.8,0)

X = lib.computeX_py(*a, *b, *c)
print("X =", X)
```



Visualization of Coplanar Vectors
a
b
c
v1 = 2a - b
v2 = 2b - c
v3 = 2c - a
v3
c
b
a
v1
v2
3
2
1
0
−1
−2
−3
Y axis
X axis