

## 2.9.16

EE25BTECH11065-Yoshita J

September 12,2025

# Question

Prove that three points A, B, and C with position vectors  $\mathbf{a}$ ,  $\mathbf{b}$ , and  $\mathbf{c}$  respectively are collinear if and only if  $(\mathbf{b} \times \mathbf{c}) + (\mathbf{c} \times \mathbf{a}) + (\mathbf{a} \times \mathbf{b}) = \mathbf{0}$ .

# Theoretical Solution

The three points A, B, and C are collinear if and only if the vectors **AB** and **AC** are parallel. The position vectors for these are:

$$\mathbf{AB} = \mathbf{b} - \mathbf{a}$$

$$\mathbf{AC} = \mathbf{c} - \mathbf{a}$$

If two vectors are collinear,

$$\mathbf{AB} \times \mathbf{AC} = \mathbf{0} \quad (1)$$

$$(\mathbf{b} - \mathbf{a}) \times (\mathbf{c} - \mathbf{a}) = \mathbf{0} \quad (2)$$

# Theoretical Solution

Expanding the expression,

$$(\mathbf{b} \times \mathbf{c}) - (\mathbf{b} \times \mathbf{a}) - (\mathbf{a} \times \mathbf{c}) + (\mathbf{a} \times \mathbf{a}) = \mathbf{0} \quad (3)$$

Rearranging the equation we get,

$$(\mathbf{a} \times \mathbf{b}) + (\mathbf{b} \times \mathbf{c}) + (\mathbf{c} \times \mathbf{a}) = \mathbf{0} \quad (4)$$

Hence we proved that that three points A, B, and C with position vectors  $\mathbf{a}$ ,  $\mathbf{b}$ , and  $\mathbf{c}$  respectively are collinear if and only if

$$(\mathbf{b} \times \mathbf{c}) + (\mathbf{c} \times \mathbf{a}) + (\mathbf{a} \times \mathbf{b}) = \mathbf{0}$$

```
#include <stdio.h>
#include <math.h>

typedef struct {
    double x, y, z;
} Vector;

Vector cross_product(Vector v1, Vector v2);
Vector add_vectors(Vector v1, Vector v2);
void print_vector(const char* name, Vector v);
void test_collinearity(const char* case_name, Vector a, Vector b,
    Vector c);
```

# C Code

```
void test_collinearity(const char* case_name, Vector a, Vector b,
    Vector c) {

    Vector a_cross_b = cross_product(a, b);
    Vector b_cross_c = cross_product(b, c);
    Vector c_cross_a = cross_product(c, a);
    Vector temp_sum = add_vectors(a_cross_b, b_cross_c);
    Vector final_sum = add_vectors(temp_sum, c_cross_a);
    print_vector(Result of (a x b) + (b x c) + (c x a), final_sum
        );

    if (is_zero_vector(final_sum)) {
        printf(Result is the zero vector. The points are
            COLLINEAR.\n);
    } else {
        printf(Result is not the zero vector. The points are NOT
            collinear.\n);
    }
}
```

```
Vector cross_product(Vector v1, Vector v2) {  
    Vector result;  
    result.x = v1.y * v2.z - v1.z * v2.y;  
    result.y = v1.z * v2.x - v1.x * v2.z;  
    result.z = v1.x * v2.y - v1.y * v2.x;  
    return result;  
}  
  
Vector add_vectors(Vector v1, Vector v2) {  
    Vector result;  
    result.x = v1.x + v2.x;  
    result.y = v1.y + v2.y;  
    result.z = v1.z + v2.z;  
    return result;  
}
```

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

def plot_vectors(ax, points, title):

    colors = ['r', 'g', 'b']

    a, b, c = points[0], points[1], points[2]

    axb = np.cross(a, b)
    bxc = np.cross(b, c)
    cxa = np.cross(c, a)

    sum_of_cross_products = axb + bxc + cxa

    ax.scatter(0, 0, 0, color='black', s=50, label='Origin')
```



```
for i, (point, label) in enumerate(zip(points, ['A', 'B', 'C']
)):
    ax.scatter(point[0], point[1], point[2], color=colors[i],
               s=50, label=f'Point {label}')
    # Draw the position vector from the origin to the point
    ax.quiver(0, 0, 0, point[0], point[1], point[2], color=
              colors[i], arrow_length_ratio=0.1, linewidth=1.5)

all_points = np.array(points + [points[0]]) # Add first point
to the end to close the shape
ax.plot(all_points[:,0], all_points[:,1], all_points[:,2],
        color='grey', linestyle='--')

ax.set_xlabel('X axis')
ax.set_ylabel('Y axis')
ax.set_zlabel('Z axis')
ax.legend()
```

```
result_str = np.array2string(sum_of_cross_products, formatter
                              ={'float_kind':lambda x: "%.1f % x"})
ax.set_title(f'{title}\nResult of (ab)+(bc)+(ca) = {
              result_str}', fontsize=10)

collinear_points = [
    np.array([2, 3, 4]),
    np.array([4, 6, 8]), # This is 2 * the first point
    np.array([6, 9, 12]) # This is 3 * the first point
]

non_collinear_points = [
    np.array([4, 1, 5]),
    np.array([1, 5, 2]),
    np.array([-2, 2, 7])
]
```

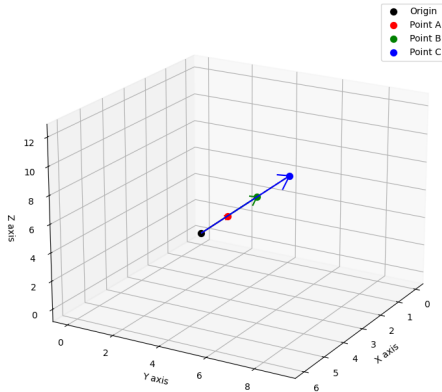
```
fig = plt.figure(figsize=(14, 7))

ax1 = fig.add_subplot(1, 2, 1, projection='3d')
plot_vectors(ax1, collinear_points, 'Case 1: Collinear Points')
ax1.view_init(elev=20, azimuth=30) # Adjust viewing angle

ax2 = fig.add_subplot(1, 2, 2, projection='3d')
plot_vectors(ax2, non_collinear_points, 'Case 2: Non-Collinear
Points')
ax2.view_init(elev=20, azimuth=30) # Adjust viewing angle

plt.tight_layout()
plt.show()
```

Case 2: Non-convex Points  
Result of  $(a \times b) + (b \times c) + (c \times a) = [0 \ 0 \ 0]$



Case 2: Non-convex Points  
Result of  $(a \times b) + (b \times c) + (c \times a) = [11 \ 24 \ 21]$

