2.9.8

Rushil Shanmukha Srinivas
EE25BTECH11057
Electrical Enggineering ,
IIT Hyderabad.

September 7, 2025

Problem Statement

Question : $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}$ are four non-zero vectors such that $\mathbf{a} \times \mathbf{b} = \mathbf{c} \times \mathbf{d}$ and $\mathbf{a} \times \mathbf{c} = 4\,\mathbf{b} \times \mathbf{d}$ then show that $(\mathbf{a} - 2\mathbf{d})$ is parallel to $(2\mathbf{b} - \mathbf{c})$ where $\mathbf{a} \neq 2\mathbf{d}$, $\mathbf{c} \neq 2\mathbf{b}$.

Cross Product of vectors

Solution : We are given nonzero vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}$ such that

$$\mathbf{a} \times \mathbf{b} = \mathbf{c} \times \mathbf{d}, \mathbf{a} \times \mathbf{c} = 4\,\mathbf{b} \times \mathbf{d}, \tag{3.1}$$

with $\mathbf{a} \neq 2\mathbf{d}$ and $\mathbf{c} \neq 2\mathbf{b}$. We need to show $(\mathbf{a} - 2\mathbf{d})$ is parallel to $(2\mathbf{b} - \mathbf{c})$, i.e.

$$(\mathbf{a} - 2\mathbf{d}) \times (2\mathbf{b} - \mathbf{c}) = \mathbf{0}. \tag{3.2}$$

By bilinearity:

$$(\mathbf{a} - 2\mathbf{d}) \times (2\mathbf{b} - \mathbf{c}) = 2(\mathbf{a} \times \mathbf{b}) - (\mathbf{a} \times \mathbf{c}) - 4(\mathbf{d} \times \mathbf{b}) + 2(\mathbf{d} \times \mathbf{c}). \tag{3.3}$$

Also, $\mathbf{d} \times \mathbf{b} = -\mathbf{b} \times \mathbf{d}$ and $\mathbf{d} \times \mathbf{c} = -\mathbf{c} \times \mathbf{d}$.
Substitute these:

$$\begin{aligned} &2(\mathbf{c} \times \mathbf{d}) - 4(\mathbf{b} \times \mathbf{d}) - 4(-\mathbf{b} \times \mathbf{d}) + 2(-\mathbf{c} \times \mathbf{d}) && (3.4) \\ &= 2(\mathbf{c} \times \mathbf{d}) - 4(\mathbf{b} \times \mathbf{d}) + 4(\mathbf{b} \times \mathbf{d}) - 2(\mathbf{c} \times \mathbf{d}) = \mathbf{0}. && (3.5) \end{aligned}$$

Matrix

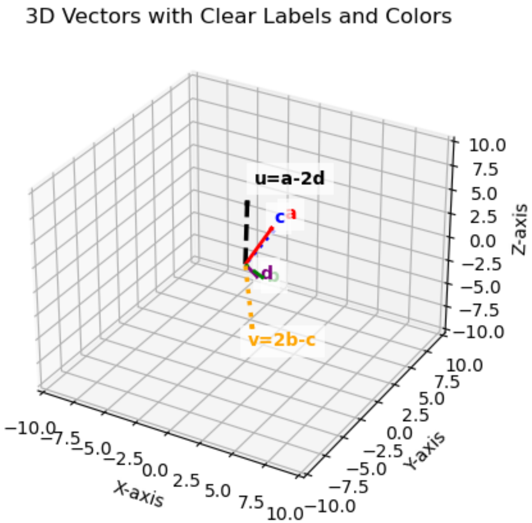
Let $\mathbf{u} = \mathbf{a} - 2\mathbf{d}$ and $\mathbf{v} = 2\mathbf{b} - \mathbf{c}$. Since $\mathbf{u} \times \mathbf{v} = \mathbf{0}$, they are linearly dependent.
Equivalently, the matrix

$$M = [\,\mathbf{u} \;\; \mathbf{v}\,] \tag{3.6}$$

has $\operatorname{rank}(M) = 1$. This is exactly the criterion for $\mathbf{u}$ and $\mathbf{v}$ to be parallel.
Therefore,

$$\boxed{\mathbf{a} - 2\mathbf{d} \parallel 2\mathbf{b} - \mathbf{c}}. \tag{3.7}$$

Plots


3D Vectors with Clear Labels and Colors
u=a-2d
c
a
d
b
v=2b-c
X-axis
Y-axis
Z-axis
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

C Code

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

// Function to check if (a-2d) is parallel to (2b-c)
int check_parallel(double a[3], double b[3], double c[3], double d[3]) {
    double u[3], v[3], cross[3];

    // u = a - 2d
    for (int i = 0; i < 3; i++) {
        u[i] = a[i] - 2.0 * d[i];
    }
    // v = 2b - c
    for (int i = 0; i < 3; i++) {
        v[i] = 2.0 * b[i] - c[i];
    }
    // cross product u x v
    cross[0] = u[1]*v[2] - u[2]*v[1];
    cross[1] = u[2]*v[0] - u[0]*v[2];
    cross[2] = u[0]*v[1] - u[1]*v[0];

    // Check if cross product is (almost) zero
    double eps = 1e-9;
    if (fabs(cross[0]) < eps && fabs(cross[1]) < eps && fabs(cross[2]) <
        eps)
        return 1; // parallel
    else
        return 0; // not parallel
}
```

Python : call_c.py

```
import ctypes

# Load the shared library (make sure parallel.so is in the same folder)
lib = ctypes.CDLL("./parallel.so")

# Define argument and return types
lib.check_parallel.argtypes = [
    ctypes.POINTER(ctypes.c_double), # a
    ctypes.POINTER(ctypes.c_double), # b
    ctypes.POINTER(ctypes.c_double), # c
    ctypes.POINTER(ctypes.c_double) # d
]
lib.check_parallel.restype = ctypes.c_int

def check_parallel(a, b, c, d):
    """
    Calls the C function check_parallel(a, b, c, d).
    Each of a, b, c, d must be length-3 lists or tuples.
    Returns True if (a-2d) || (2b-c), False otherwise.
    """
    # Convert Python lists to C arrays
    A = (ctypes.c_double * 3)(*a)
    B = (ctypes.c_double * 3)(*b)
    C = (ctypes.c_double * 3)(*c)
    D = (ctypes.c_double * 3)(*d)

    # Call the C function
    result = lib.check_parallel(A, B, C, D)
    return bool(result)

# ----------------- Example usage
    -----------------
if __name__ == "__main__":
    a = [1.0, 2.0, 3.0]
    b = [2.0, -1.0, 0.0]
    c = [0.0, 4.0, 1.0]
    d = [1.0, 0.0, −1.0]

if check_parallel(a, b, c, d):
        print("(a − 2d) is parallel to (2b − c)")
    else:
        print("(a − 2d) is NOT parallel to (2b − c)")
```

Python Code for Plotting

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

def check_parallel(a, b, c, d, tol=1e-9):
    a, b, c, d = map(np.array, (a, b, c, d))
    u = a - 2*d
    v = 2*b - c
    cross = np.cross(u, v)
    return np.linalg.norm(cross) < tol, u, v

def plot_vectors(a, b, c, d, u, v):
    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')
    # Helper function to draw vector + label
    def draw_vector(vec, name, color, style="-", scale=1.2, offset
        =(0,0,0), lw=2.0):
```

```
        ax.quiver(0, 0, 0, vec[0], vec[1], vec[2],
                  color=color, linestyle=style,
                  arrow_length_ratio=0.08, linewidth=lw)
        ax.text(vec[0]*scale + offset[0],
                vec[1]*scale + offset[1],
                vec[2]*scale + offset[2],
                name, fontsize=10, weight="bold", color=color,
                bbox=dict(facecolor="white", alpha=0.7, edgecolor="
                    none"))

    # Base vectors (unique colors + offsets to avoid overlap)
    draw_vector(a, "a", "red", style="-", offset=(0.4, 0.2, 0))
    draw_vector(c, "c", "blue", style=":", offset=(-0.4, -0.2, 0))
    draw_vector(b, "b", "green", style="--", offset=(0.3, -0.5, 0.3)) #
        shifted away
    draw_vector(d, "d", "purple", style="-.", offset=(-0.3, 0.5, -0.3))
        # opposite shift
```

```
# Special vectors u and v (thicker lines)
draw_vector(u, "u=a-2d", "black", style="--", lw=2.5, scale
    =1.25, offset=(0.3,0.3,0))
draw_vector(v, "v=2b-c", "orange", style=":", lw=2.5, scale=1.25,
    offset=(-0.3,-0.3,0))

# Axis scaling
all_vecs = np.array([a, b, c, d, u, v])
max_range = np.max(np.abs(all_vecs)) * 1.5 + 1
ax.set_xlim([-max_range, max_range])
ax.set_ylim([-max_range, max_range])
ax.set_zlim([-max_range, max_range])

ax.set_xlabel("X-axis")
ax.set_ylabel("Y-axis")
ax.set_zlabel("Z-axis")
ax.set_title("3D Vectors with Clear Labels and Colors")
```

```
    plt.show()

if __name__ == "__main__":
    a = np.array([1, 2, 3])
    b = np.array([2, −1, 0])
    c = np.array([0, 4, 1])
    d = np.array([1, 0, −1])

    is_parallel, u, v = check_parallel(a, b, c, d)
    print("Parallel?", is_parallel)

    plot_vectors(a, b, c, d, u, v)
```