2.5.4

Hema Havil - EE25BTECH11050

October 1, 2025

Question

If $\mathbf{a} = 2\hat{i} + y\hat{j} + \hat{k}$ and $\mathbf{b} = \hat{i} + 2\hat{j} + 3\hat{k}$ are two vectors for which the vector $(\mathbf{a} + \mathbf{b})$ is perpendicular to the vector $(\mathbf{a} - \mathbf{b})$, then find all the possible values of y.

Theoretical Solution

Let the given vectors be :

$$\mathbf{a} = \begin{pmatrix} 2 \\ y \\ 1 \end{pmatrix} \mathbf{b} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \tag{1}$$

Given that $\mathbf{a} + \mathbf{b}$ and $\mathbf{a} - \mathbf{b}$ are perpendicular, then

$$(\mathbf{a} + \mathbf{b})^T (\mathbf{a} - \mathbf{b}) = 0 \tag{2}$$

$$\mathbf{a}^T\mathbf{a} - \mathbf{b}^T\mathbf{b} = 0 \tag{3}$$

$$\mathbf{a}^T\mathbf{a} = \mathbf{b}^T\mathbf{b} \tag{4}$$

Theoretical Solution

The values of $\mathbf{a}^T\mathbf{a}$ and $\mathbf{b}^T\mathbf{b}$ can be calculated by,

$$\mathbf{a}^T\mathbf{a} = \begin{pmatrix} 2 & y & 1 \end{pmatrix} \begin{pmatrix} 2 \\ y \\ 1 \end{pmatrix} = 4 + y^2 + 1 = 5 + y^2 \tag{5}$$

$$\mathbf{b}^T\mathbf{b} = \begin{pmatrix} 1 & 2 & 3 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} = 1 + 4 + 9 = 14 \tag{6}$$

From equation 0.4,

$$5 + y^2 = 14 \tag{7}$$

$$y^2 = 9 \tag{8}$$

$$y = \pm 3 \tag{9}$$

Therefore the values of y are 3 and -3

C Code- Ploting the given vectors

```c

#include <stdio.h>
#include <math.h>
double matrix_multiply_transpose(const double u[3], const double
    v[3]) {
    double result = 0.0;
    for (int i = 0; i < 3; ++i)
        result += u[i] * v[i]; // Equivalent to u^T * v
    return result;
}

int solve_y_matrix_form(double* y1, double* y2) {
    double u1 = 2 + 1;
    double u3 = 1 + 3;
    double v1 = 2 - 1;
    double v3 = 1 - 3;
```

C Code- Ploting the given vectors

```c
    double A = 1.0;
    double B = 0.0;
    double C = -9.0;

    double discriminant = B*B - 4*A*C;
    if (discriminant < 0)
        return -1;

    double sqrt_disc = sqrt(discriminant);
    *y1 = (-B + sqrt_disc) / (2*A);
    *y2 = (-B - sqrt_disc) / (2*A);
    return 0;
}
```

Python Code using shared output

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

# Load C shared library
lib = ctypes.CDLL('./2.5.4.so') # Change to .dll on Windows

# Define return and arg types
lib.solve_y_matrix_form.argtypes = [ctypes.POINTER(ctypes.
    c_double), ctypes.POINTER(ctypes.c_double)]
lib.solve_y_matrix_form.restype = ctypes.c_int

# Prepare output variables
y1 = ctypes.c_double()
y2 = ctypes.c_double()

# Call C function
res = lib.solve_y_matrix_form(ctypes.byref(y1), ctypes.byref(y2))
```

Python Code using shared output

```
     if res != 0:
    print(No real roots for y found.)
    exit()

y_values = [y1.value, y2.value]
print(fValues of y computed using matrix multiplication in C: {y_values})

# Plotting function
def plot_vectors_for_y(y_val):
    a = np.array([2, y_val, 1])
    b = np.array([1, 2, 3])
    a_plus_b = a + b
    a_minus_b = a - b
    origin = np.array([0, 0, 0])

    fig = plt.figure()
    ax = fig.add_subplot(111, projection='3d')
```

Python Code using shared output

```
def draw(vec, color, label):
       ax.quiver(*origin, *vec, color=color, label=label,
           arrow_length_ratio=0.1)

    draw(a, 'red', 'a')
    draw(b, 'blue', 'b')
    draw(a_plus_b, 'green', 'a + b')
    draw(a_minus_b, 'purple', 'a - b')

    ax.set_xlim([-1, 6])
    ax.set_ylim([-4, 6])
    ax.set_zlim([-4, 6])
    ax.set_xlabel('X')
    ax.set_ylabel('Y')
    ax.set_zlabel('Z')
```

Python Code using shared output

```python
    ax.set_title(f'Vectors for y = {y_val}')
    ax.legend()
    plt.grid(True)
    plt.show()
for y in y_values:
    plot_vectors_for_y(y)
```

Plot by python using shared output from c - 1

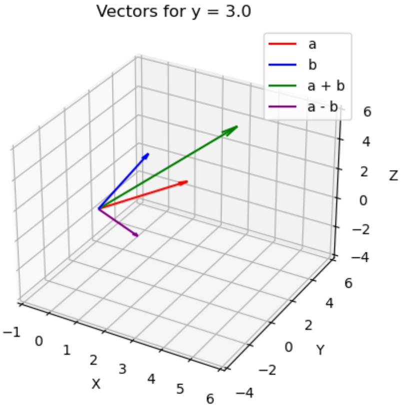

Figure: Plot of the vectors when y=3

Plot by python using shared output from c - 2

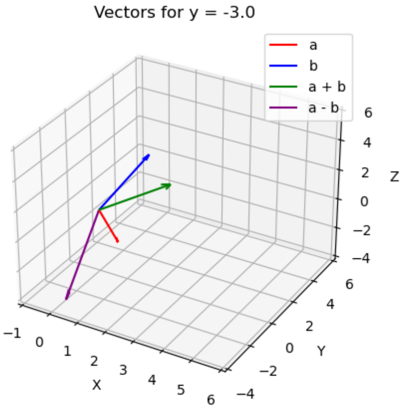

Figure: Plot of the vectors when y=-3

Python code for the plot

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

# Define vector b
b = np.array([1, 2, 3])

# Define a function for the condition (a+b)  (a-b) = 0
def find_y():
    # a = [2, y, 1]
    # Condition: (a + b)  (a - b) = 0
    # (2+1, y+2, 1+3)  (2-1, y-2, 1-3) = 0
    # Simplify: [3, y+2, 4]  [1, y-2, -2] = 0
    # Dot product: 3*1 + (y+2)*(y-2) + 4*(-2) = 0
    # 3 + (y^2 - 4) - 8 = 0 => y^2 - 9 = 0 => y = 3
    y1 = 3
    y2 = -3
    return [y1, y2]
```

Python code for the plot

```
# Get possible y values
y_solutions = find_y()
print(Possible values of y:, y_solutions)

# Choose one y to plot
y = y_solutions[0] # you can also try y_solutions[1]

# Define vectors a, a+b, a-b
a = np.array([2, y, 1])
a_plus_b = a + b
a_minus_b = a - b

# Plot vectors
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
origin = np.array([0, 0, 0])
```

Python code for plot

```
     ax.quiver(*origin, *a_plus_b, color='r', label='a + b')
ax.quiver(*origin, *a_minus_b, color='b', label='a - b')

ax.set_xlim([0, 5])
ax.set_ylim([0, 5])
ax.set_zlim([0, 5])
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.legend()
plt.title(f'Vectors (a+b) and (a-b) for y={y}')
plt.show()
```

Plot of the vectors using python - 1

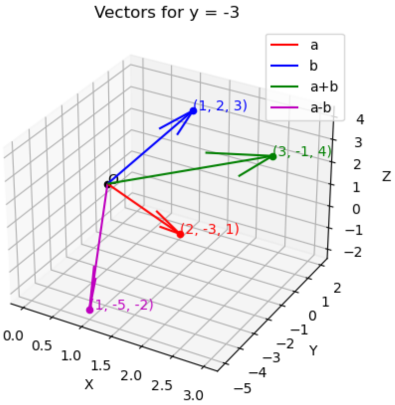

Figure: Plot of the vectors when y=-3

Plot of the vectors using python - 2

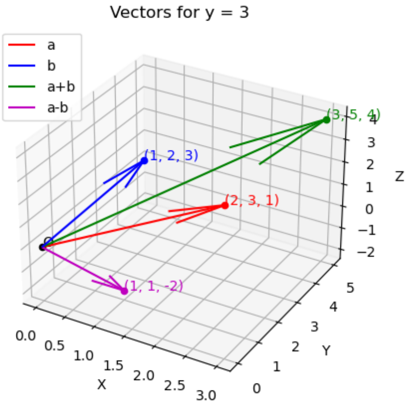

Figure: Plot of the vectors when y=3