2.9.10

Vaishnavi - EE25BTECH11059

September 12, 2025

Question

Let $\mathbf{a}$ and $\mathbf{b}$ be two vectors such that $\|\mathbf{a}+\mathbf{b}\| = \|\mathbf{b}\|$. Prove that $\mathbf{a}+2\mathbf{b}$ is perpendicular to $\mathbf{a}$.

Solution

Variable	Value
a	vector **a**
b	vector **b**

Table: Variables Used

Solution

$$(\mathbf{a}+\mathbf{b})^T(\mathbf{a}+\mathbf{b}) = \mathbf{b}^\mathbf{T}\mathbf{b}. \tag{1}$$

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

Since dot product is symmetric

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

$$\mathbf{a}^\mathbf{T}\mathbf{a} + 2\,\mathbf{a}^\mathbf{T}\mathbf{b} + \mathbf{b}^\mathbf{T}\mathbf{b} = \mathbf{b}^\mathbf{T}\mathbf{b}. \tag{4}$$

$$\mathbf{a}^\mathbf{T}\mathbf{a} + 2\,\mathbf{a}^\mathbf{T}\mathbf{b} = 0. \tag{5}$$

solution

We want to show $(\mathbf{a} + 2\mathbf{b})$ is perpendicular to $\mathbf{a}$.

$$\text{To prove: } \mathbf{a}^{\mathbf{T}}(\mathbf{a} + 2\mathbf{b}) = 0 \tag{6}$$

$$\mathbf{a}^{\mathbf{T}}(\mathbf{a} + 2\mathbf{b}) = \mathbf{a}^{\mathbf{T}}\mathbf{a} + 2\,\mathbf{a}^{\mathbf{T}}\mathbf{b} \tag{7}$$

By eq 5 and 7

$$\mathbf{a}^{\mathbf{T}}(\mathbf{a} + 2\mathbf{b}) = 0 \tag{8}$$

Hence proved

Graph

Refer to Figure

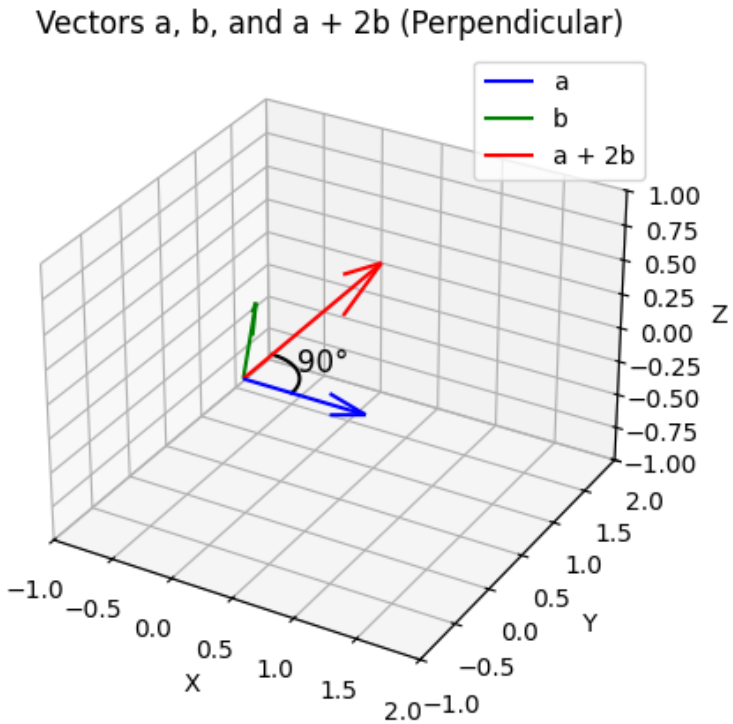

Figure:

Python Code

```
# Plot origin
origin = np.zeros(3)

# Plot vectors
ax.quiver(*origin, *a, color='blue', label='a')
ax.quiver(*origin, *b, color='green', label='b')
ax.quiver(*origin, *a_plus_2b, color='red', label='a +
    2b')

# ---- Add 90-degree arc ----
# Normalize vectors
a_unit = a / np.linalg.norm(a)
a2b_unit = a_plus_2b / np.linalg.norm(a_plus_2b)

# Create arc between a and a+2b
theta = np.linspace(0, np.pi / 2, 30)
arc_radius = 0.4
arc_points = np.array([arc_radius * (np.cos(t) *
    a_unit + np.sin(t) * a2b_unit) for t in theta])
```

Python Code

```
# Plot the arc
ax.plot(arc_points[:, 0], arc_points[:, 1], arc_points
    [:, 2], color='black')

# Label the angle
angle_label_pos = arc_radius * (np.cos(np.pi / 4) *
    a_unit + np.sin(np.pi / 4) * a2b_unit)
ax.text(*angle_label_pos,  90    , fontsize=12, color='
    black')

# Axis settings
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.set_xlim([-1, 2])
ax.set_ylim([-1, 2])
ax.set_zlim([-1, 1])
ax.legend()
```

Python Code

```
# Save figure
plt.savefig( graph4.png )
print( Saved as graph4.png )

# Optional: Show the plot
# plt.show()
```

C Code

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

#define EPS 1e-6

// Compute dot product of two 2D vectors stored as 1
    2 matrices
// a: array double a[2]; b: array double b[2]
double dot2(const double a[2], const double b[2]) {
    return a[0]*b[0] + a[1]*b[1];
}

// Solve the question using matrix-like vector
   operations
// returns 1 if (a + 2b) is perpendicular to a under
   the condition ||a + b|| == ||b||, else 0
```

C Code

```c
int solve_matrix_vectors(double a0, double a1,
    double b0, double b1) {
  double a_vec[2] = { a0, a1 };
  double b_vec[2] = { b0, b1 };

  double ab_vec[2] = { a0 + b0, a1 + b1 };      // a
     + b
  double b_norm2 = dot2(b_vec, b_vec);          // ||
     b||^2
  double ab_norm2 = dot2(ab_vec, ab_vec);        //
     ||a + b||^2

  if (fabs(ab_norm2 - b_norm2) > EPS) {
      // Condition ||a + b|| == ||b|| fails
      return 0;
  }
```

C Code

```
    double a2b_vec[2] = { a0 + 2.0 * b0, a1 + 2.0 * b1 };
          // a + 2b

        double dp = dot2(a_vec, a2b_vec);     // a     (a +
            2b)

        if (fabs(dp) < EPS) {
            // Perpendicular
            return 1;
        } else {
            return 0;
        }
}
```

Python and C Code

```
import ctypes
import os

# locate shared library
_dir = os.path.dirname(__file__)
lib_path = os.path.join(_dir, 'libmatrix_vectors.so')

lib = ctypes.CDLL(lib_path)

# declare the argument types and return type
lib.solve_matrix_vectors.argtypes = [
    ctypes.c_double,  # a0 (ax)
    ctypes.c_double,  # a1 (ay)
    ctypes.c_double,  # b0 (bx)
    ctypes.c_double   # b1 (by)
]
```

Python and C Code

```
lib.solve_matrix_vectors.restype = ctypes.c_int

def solve_matrix_vectors(a0, a1, b0, b1):
    """Wrapper function returning True / False."""
    res = lib.solve_matrix_vectors(ctypes.c_double(a0)
        ,
                                   ctypes.c_double(a1)
                                       ,
                                   ctypes.c_double(b0)
                                       ,
                                   ctypes.c_double(b1)
                                       )
    return bool(res)

if __name__ == "__main__":
    # Examples
    a0, a1 = 2.0, 1.0
    b0, b1 = 1.0, 2.0
```