Question

Let $\mathbf{a} = 4\hat{\imath} + 5\hat{\jmath} - \hat{k}$, $\mathbf{b} = \hat{\imath} - 4\hat{\jmath} + 5\hat{k}$, $\mathbf{c} = 3\hat{\imath} + \hat{\jmath} - \hat{k}$. Find $\mathbf{d}$ perpendicular to both $\mathbf{b}$ and $\mathbf{c}$ and satisfying $\mathbf{d} \cdot \mathbf{a} = 21$.

Theoritical solution

Write vectors as column matrices:

$$\mathbf{a} = \begin{pmatrix} 4 \\ 5 \\ -1 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 1 \\ -4 \\ 5 \end{pmatrix}, \quad \mathbf{c} = \begin{pmatrix} 3 \\ 1 \\ -1 \end{pmatrix}.$$

Since $\mathbf{d}$ is perpendicular to both $\mathbf{b}$ and $\mathbf{c}$,

$$\mathbf{d} = \lambda(\mathbf{b} \times \mathbf{c}).$$

Compute the cross product:

$$\mathbf{b} \times \mathbf{c} = \begin{pmatrix} (-4)(-1) - 5(1) \\ -(1(-1) - 5(3)) \\ 1(1) - (-4)(3) \end{pmatrix} = \begin{pmatrix} -1 \\ 16 \\ 13 \end{pmatrix}.$$

Thus

$$\mathbf{d} = \lambda \begin{pmatrix} -1 \\ 16 \\ 13 \end{pmatrix}.$$

Now apply the condition $\mathbf{d} \cdot \mathbf{a} = 21$:

$$\mathbf{d} \cdot \mathbf{a} = \lambda \begin{pmatrix} -1 & 16 & 13 \end{pmatrix} \begin{pmatrix} 4 \\ 5 \\ -1 \end{pmatrix}.$$

$$= \lambda(-4 + 80 - 13) = \lambda(63).$$

So

$$\lambda(63) = 21 \quad \Rightarrow \quad \lambda = \tfrac{1}{3}.$$

Hence

$$\mathbf{d} = \frac{1}{3} \begin{pmatrix} -1 \\ 16 \\ 13 \end{pmatrix} = -\frac{1}{3}\hat{\imath} + \frac{16}{3}\hat{\jmath} + \frac{13}{3}\hat{k}.$$

$$\boxed{\mathbf{d} = -\frac{1}{3}\hat{\imath} + \frac{16}{3}\hat{\jmath} + \frac{13}{3}\hat{k}}$$

C code

```
#include <stdio.h>
// Function to compute cross product
void crossProduct(int a[3], int b[3], int c[3], int K) {
    c[0] = K * (a[1]*b[2] - a[2]*b[1]); // i component
    c[1] = K * (a[2]*b[0] - a[0]*b[2]); // j component
    c[2] = K * (a[0]*b[1] - a[1]*b[0]); // k component
}int main() {
    int a[3], b[3], c[3], K;
  // Input vectors a and b
    printf("Enter vector a (ax ay az): ");
    scanf("%d %d %d", &a[0], &a[1], &a[2]);
printf("Enter vector b (bx by bz): ");
    scanf("%d %d %d", &b[0], &b[1], &b[2]);
  printf("Enter scalar K: ");
    scanf("%d", &K);
// Compute c = K(a *b)
    crossProduct(a, b, c, K);
  // Print result
    printf("Vector c = %di + %dj + %dk\n", c[0], c[1], c[2]);
```

Python Plotting Code - Part 1

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
# Define the vectors
a = np.array([4, 5, -1])
b = np.array([1, -4, 5])
c = np.array([3, 1, -1])
# Find a direction for d (perpendicular to both c and b)
d_dir = np.cross(c, b)
# Find k such that d*a = 21
k = 21 / np.dot(d_dir, a)
d = k * d_dir
# Origin for all vectors
origin = np.zeros(3)
# Set up 3D plot
fig = plt.figure(figsize=(7, 7))
ax = fig.add_subplot(111, projection='3d')
```

Python plotting code - part 2

```
ax.quiver(*origin, *a, color='r', label='a', length=np.linalg.
    norm(a), arrow_length_ratio=0.1)
ax.quiver(*origin, *b, color='g', label='b', length=np.linalg.
    norm(b), arrow_length_ratio=0.1)
ax.quiver(*origin, *c, color='b', label='c', length=np.linalg.
    norm(c), arrow_length_ratio=0.1)
ax.quiver(*origin, *d, color='k', label='d', length=np.linalg.
    norm(d), arrow_length_ratio=0.15)
# Styling and labels
ax.set_xlim([0, 8])
ax.set_ylim([0, 8])
ax.set_zlim([-2, 8])
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.set_title('3D Plot of Vectors a, b, c, and d')
ax.legend()
plt.tight_layout()
plt.show()
```

3D Plot of Vectors a, b, c, and d

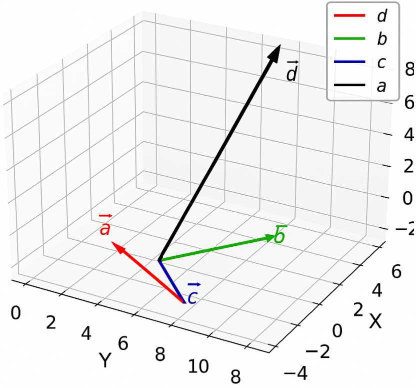

Figure: plot