2.8.10

Jnanesh Sathisha karmar - EE25BTECH11029

September 6,2025

Question

If with reference to the right handed system of mutually perpendicular unit vectors $\hat{i}, \hat{j}$ and $\hat{k}, \alpha = 3\hat{i} - \hat{j}, \beta = 2\hat{i} + \hat{j} - 3\hat{k}$, then express β in the form $\beta = \beta_1 + \beta_2$ where β_1 is parallel to α and β_2 is perpendicular to α

Equation

Given details:

$$\alpha = 3\hat{i} - \hat{j} = \begin{pmatrix} 3 \\ -1 \\ 0 \end{pmatrix} \tag{1}$$

$$\beta = 2\hat{i} + \hat{j} - 3\hat{k} = \begin{pmatrix} 2 \\ 1 \\ -3 \end{pmatrix} \tag{2}$$

Theoretical Solution

β_1 is a projection of β on α
The projection formula for projection is:

$$\beta_1 = \frac{\beta^{\mathbf{T}}\alpha}{\|\alpha^{\mathbf{2}}\|}\alpha \tag{3}$$

Theoretical Solution

$$= \frac{\begin{pmatrix} 2 & 1 & -3 \end{pmatrix} \begin{pmatrix} 3 \\ -1 \\ 0 \end{pmatrix}}{(3)^2 + (-1)^2 + (0)^2} \begin{pmatrix} 3 \\ -1 \\ 0 \end{pmatrix} \tag{4}$$

$$= \frac{5}{10} \begin{pmatrix} 3 \\ -1 \\ 0 \end{pmatrix} \tag{5}$$

$$= \begin{pmatrix} \frac{3}{2} \\ \frac{-1}{2} \\ 0 \end{pmatrix} \tag{6}$$

Theoretical Solution

Now according to the given equation :

$$\beta = \beta_1 + \beta_2 \tag{7}$$

$$\beta_2 = \beta - \beta_1 \tag{8}$$

$$\beta_2 = \begin{pmatrix} 2 \\ 1 \\ -3 \end{pmatrix} - \begin{pmatrix} \frac{3}{2} \\ \frac{-1}{2} \\ 0 \end{pmatrix} \tag{9}$$

$$\beta_2 = \begin{pmatrix} \frac{1}{2} \\ \frac{3}{2} \\ -3 \end{pmatrix} \tag{10}$$

Theoretical Solution

Lets verify wheter β_2 is perpendicular to α
For that:

$$\alpha^T.\beta_2 = 0 \tag{11}$$

$$\begin{pmatrix} 3 & -1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{2} \\ \frac{3}{2} \\ -3 \end{pmatrix} = \begin{pmatrix} 0 \end{pmatrix} \tag{12}$$

Therefore β_2 is perpendicular to α

Theoretical Solution

Therefore β is:

$$\beta = \beta_1 + \beta_2 \tag{13}$$

$$\beta = \begin{pmatrix} \frac{3}{2} \\ \frac{-1}{2} \\ 0 \end{pmatrix} + \begin{pmatrix} \frac{1}{2} \\ \frac{3}{2} \\ -3 \end{pmatrix} \tag{14}$$

Theoretical Solution

Performing QR decomposition on the matrix $\begin{pmatrix} \alpha & \beta \end{pmatrix}$

$$\mathbf{A} = \begin{pmatrix} \alpha & \beta \end{pmatrix} = \begin{pmatrix} 3 & 2 \\ -1 & 1 \\ 0 & -3 \end{pmatrix} \tag{15}$$

Finding $\mathbf{q_1}$ (normalized α)

$$\|\alpha\| = \sqrt{\begin{pmatrix} \alpha \end{pmatrix} \begin{pmatrix} \alpha \end{pmatrix}^T} = \sqrt{9+1} = \sqrt{10} \tag{16}$$

$$\mathbf{q_1} = \frac{\alpha}{\|\alpha\|} = \begin{pmatrix} \frac{3}{\sqrt{10}} & \frac{-1}{\sqrt{10}} & 0 \end{pmatrix} \tag{17}$$

Theoretical Solution

The projection of β on $\mathbf{q_1}$
projection coefficient:

$$\mathbf{r_{12}} = \mathbf{q_1^T}\beta = \frac{3}{\sqrt{10}}.2 + \frac{-1}{\sqrt{10}} + 0 = \frac{5}{\sqrt{10}} \tag{18}$$

projection:

$$proj\,(\beta)_{q_1} = \mathbf{r_{12}}.\mathbf{q_1} = \frac{5}{\sqrt{10}}.\begin{pmatrix} \frac{3}{\sqrt{10}} & \frac{-1}{\sqrt{10}} & 0 \end{pmatrix} = \begin{pmatrix} 1.5 & -0.5 & -3 \end{pmatrix} \tag{19}$$

Theoretical Solution

subtract projection from β:

$$\mathbf{u_1} = \beta - proj\left(\beta\right)_{q_1} = \begin{pmatrix} 0.5 & 1.5 & -3 \end{pmatrix} \tag{20}$$

we normalize $\mathbf{u_2}$ to get $\mathbf{q_2}$
compute $\left\|\mathbf{u_2}\right\|$

$$\left\|u_2\right\| = \sqrt{\begin{pmatrix} u_2 \end{pmatrix} \begin{pmatrix} u_2 \end{pmatrix}^T} = \sqrt{11.5} \tag{21}$$

$$\mathbf{q_2} = \frac{\mathbf{u_2}}{\left\|u_2\right\|} = \begin{pmatrix} \frac{0.5}{\sqrt{11.5}} & \frac{1.5}{\sqrt{11.5}} & \frac{-3}{\sqrt{11.5}} \end{pmatrix} \tag{22}$$

Theoretical Solution

$\mathbf{Q}$'s columns are $\mathbf{q_1}$ and $\mathbf{q_2}$

$$\mathbf{Q} = \begin{pmatrix} 3 & 0.5 \\ -1 & 1.5 \\ 0 & -3 \end{pmatrix} \tag{23}$$

$\mathbf{R}$ is:

$$\mathbf{R} = \begin{pmatrix} \sqrt{10} & \frac{5}{\sqrt{10}} \\ 0 & \sqrt{11.5} \end{pmatrix} \tag{24}$$

C Code (1) - Function to store the points

```c
#include <stdio.h>

double start_points[4][3]={
        {0.0,0.0,0.0},
        {0.0,0.0,0.0},
        {0.0,0.0,0.0},
        {0.0,0.0,0.0}
};
double end_points[4][3]={
        {3.0,-1.0,0.0},
        {2.0,1.0,-3.0},
        {3/2,-1/2,0.0},
        {1/2,3/2,-3.0}
```

C Code (1) - Function to store the points

```c
};
void get_start_points(double *arr){
        for (int i=0;i<4;i++){
                arr[i*3+0]=start_points[i][0];
                arr[i*3+1]=start_points[i][1];
                arr[i*3+2]=start_points[i][2];
        }
}
void get_end_points(double *arr){
        for (int i=0;i<4;i++){
                arr[i*3+0]=end_points[i][0];
                arr[i*3+1]=end_points[i][1];
                arr[i*3+2]=end_points[i][2];
        }
}
```

Python Code - Using Shared Object

```
import ctypes
import numpy as np
import matplotlib.pyplot as plt
import subprocess
y=input("are you using termux?(y/n)=")


lib = ctypes.CDLL('./vectors.so')

lib.get_start_points.argtypes = [ctypes.POINTER(ctypes.c_double)]
lib.get_end_points.argtypes = [ctypes.POINTER(ctypes.c_double)]
```

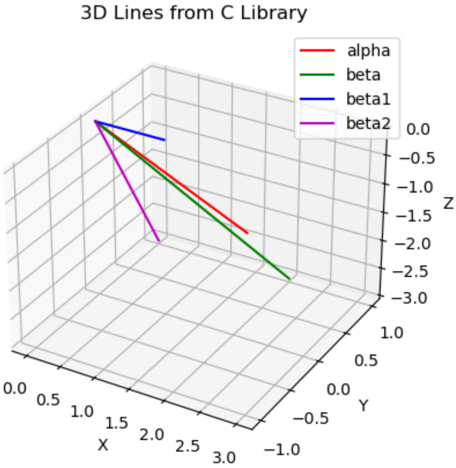
Python Code - Using Shared Object

```
n_lines = 4

start_points = np.zeros((n_lines,3), dtype=np.float64)
end_points = np.zeros((n_lines,3), dtype=np.float64)
lib.get_start_points(start_points.ctypes.data_as(ctypes.POINTER(
    ctypes.c_double)))
lib.get_end_points(end_points.ctypes.data_as(ctypes.POINTER(
    ctypes.c_double)))


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

colors = ['r','g','b','m']
label=['alpha','beta','beta1','beta2']
```

Python Code - Using Shared Object

```
for i in range(n_lines):
    xs = [start_points[i,0], end_points[i,0]]
    ys = [start_points[i,1], end_points[i,1]]
    zs = [start_points[i,2], end_points[i,2]]
    ax.plot(xs, ys, zs, color=colors[i], label=label[i])

ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.legend()
plt.title("3D Lines from C Library")
fig.savefig('../figs/fig2.png')
if (y=='y'):
    subprocess.run(shlex.split('termux-open ../figs/fig.png'))
else:
    subprocess.run(["open", "../figs/fig.png"])
plt.show()
```

Plot-Using Both C and Python



Python Code

```
import numpy as np
import matplotlib.pyplot as plt
import sys
import subprocess
print('Using termux?(y/n)')
y = input()
alpha=np.array([3,-1,0])
beta=np.array([2,1,-3])
beta1=np.array([3/2,-1/2,0])
beta2=np.array([1/2,3/2,-3])
```

Python Code

```
fig=plt.figure()
ax=fig.add_subplot(111,projection='3d')
ax.plot([0,alpha[0]],[0,alpha[1]],[0,alpha[2]],'b-',label='alpha'
    )
ax.plot([0,beta[0]],[0,beta[1]],[0,beta[2]],'g-',label='beta')
ax.plot([0,beta1[0]],[0,beta1[1]],[0,beta1[2]],'r-',label='beta1'
    )
ax.plot([0,beta2[0]],[0,beta2[1]],[0,beta2[2]],color='pink',label
    ='beta')
```

Python Code

```
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_zlabel('$z$')
ax.legend(loc='best')
ax.grid(True)
ax.axis('equal')
fig.savefig('../figs/fig.png')
print('Saved figure to ../figs/fig.png')

if(y == 'y'):
    subprocess.run(shlex.split('termux-open ../figs/fig.png'))
else:
    subprocess.run(["open", "../figs/fig.png"])

plt.show()
```

Plot-Using only Python