Matgeo Presentation - Problem 12.492

ee25btech11063 - Vejith

October 3, 2025

Question

Direction cosines of the vector $3\hat{\mathbf{i}} + -2\hat{\mathbf{j}} + 6\hat{\mathbf{k}}$ are

a) $[3/7, -2/7, 6/7]$

b) $[-3/7, 2/7, -6/7]$

c) $[-7/3, 7/2, -7/6]$

d) $[7/3, -7/2, 7/6]$

Solution

let

$$\mathbf{r} = \begin{pmatrix} 3 \\ -2 \\ 6 \end{pmatrix} \tag{0.1}$$

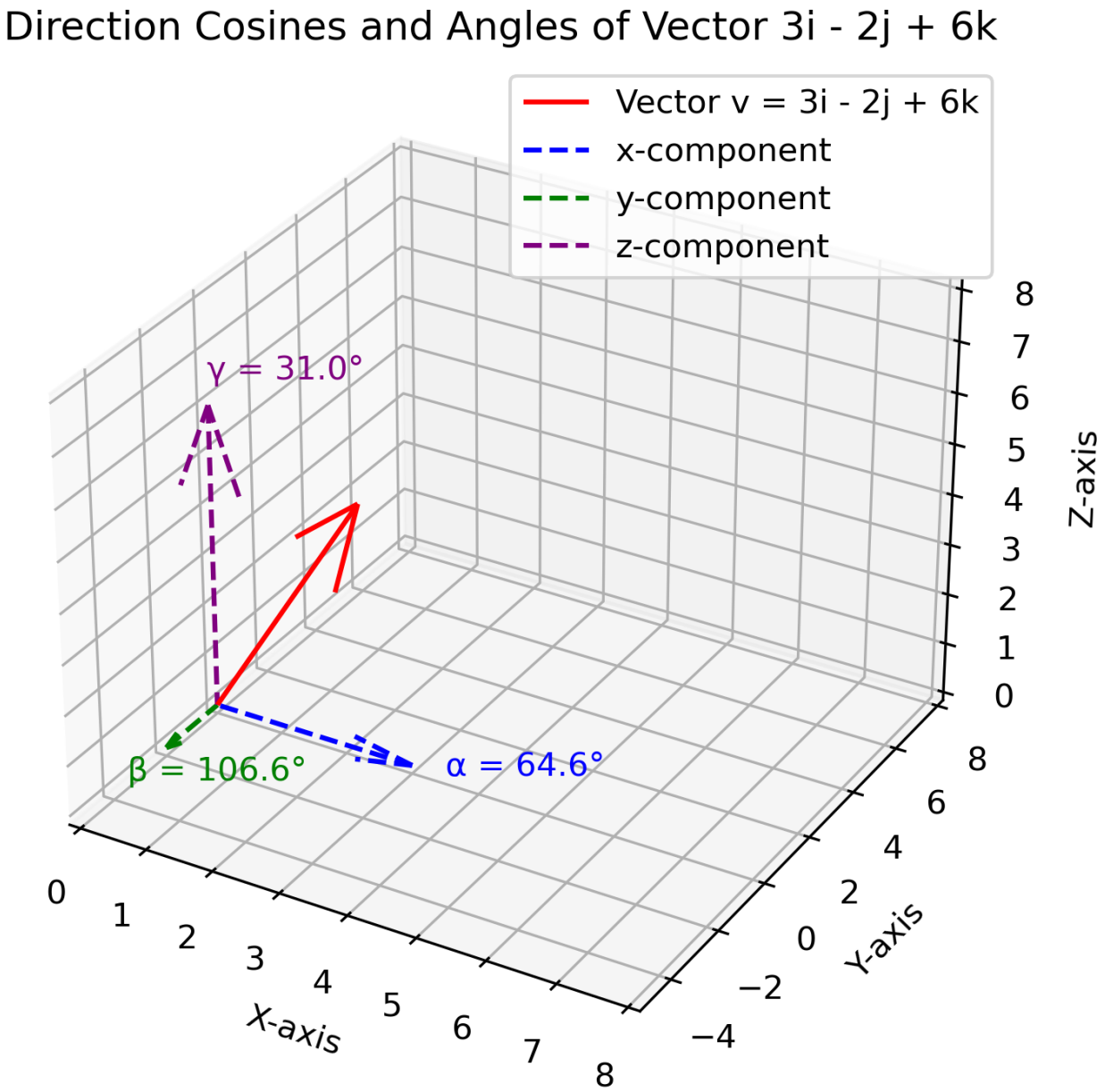
$$\|\mathbf{r}\| = \sqrt{9 + 4 + 36} \tag{0.2}$$

$$\implies \|\mathbf{r}\| = 7 \tag{0.3}$$

The unit vector in the direction of $\mathbf{r}$ is

$$\frac{\mathbf{r}}{\|\mathbf{r}\|} = \frac{1}{7}\begin{pmatrix} 3 \\ -2 \\ 6 \end{pmatrix} = \begin{pmatrix} \frac{3}{7} \\ \frac{-2}{7} \\ \frac{6}{7} \end{pmatrix} \tag{0.4}$$

Plot


Direction Cosines and Angles of Vector 3i - 2j + 6k
Vector v = 3i - 2j + 6k
x-component
y-component
z-component
γ = 31.0°
β = 106.6°
α = 64.6°
X-axis
Y-axis
Z-axis

C Code: dc.c

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

int main() {
    // Vector components
    int x = 3, y = -2, z = 6;
    double magnitude, l, m, n;

    // Calculate magnitude
    magnitude = sqrt(x*x + y*y + z*z);

    // Direction cosines
    l = x / magnitude;
    m = y / magnitude;
    n = z / magnitude;

    // Open file for writing
    FILE *fp = fopen("dc.dat", "w");
    if (fp == NULL) {
        printf("Error opening file!\n");
        return 1;
    }

    fprintf(fp, "Direction cosines of vector (3, -2, 6):\n");
    fprintf(fp, "l = %.4f\n", l);
    fprintf(fp, "m = %.4f\n", m);
    fprintf(fp, "n = %.4f\n", n);

    fclose(fp);
    printf("Output written to dc.dat successfully.\n");

    return 0;
}
```

Python: plot.py

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

# Vector components
a, b, c = 3, -2, 6
v = np.array([a, b, c])

# Magnitude of vector
magnitude = np.linalg.norm(v)

# Direction cosines
cos_alpha = a / magnitude
cos_beta = b / magnitude
cos_gamma = c / magnitude

# Direction angles in degrees
alpha = np.degrees(np.arccos(cos_alpha))
beta = np.degrees(np.arccos(cos_beta))
gamma = np.degrees(np.arccos(cos_gamma))

# Print direction cosines and angles
print(f"Direction␣cosines:␣({cos_alpha:.2f},␣{cos_beta:.2f},␣{cos_gamma:.2f})")
print(f"Direction␣angles␣(degrees):␣␣=␣{alpha:.2f},␣␣=␣{beta:.2f},␣␣=␣{gamma:.2f}")

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

# Plot the vector
ax.quiver(0, 0, 0, a, b, c, color='r', label='Vector␣v␣=␣3i␣-␣2j␣+␣6k')
```

Python: plot.py

```
# Plot projections on axes
ax.quiver(0, 0, 0, a, 0, 0, color='blue', linestyle='dashed', label='x-component')
ax.quiver(0, 0, 0, 0, b, 0, color='green', linestyle='dashed', label='y-component')
ax.quiver(0, 0, 0, 0, 0, c, color='purple', linestyle='dashed', label='z-component')

# Annotate angles with adjusted positions to avoid overlap
ax.text(a + 0.5, 0, 0, f'␣=␣{alpha:.1f}', color='blue', fontsize=10)
ax.text(0, b - 1.5, 0, f'␣=␣{beta:.1f}', color='green', fontsize=10)
ax.text(0, 0, c + 0.5, f'␣=␣{gamma:.1f}', color='purple', fontsize=10)

# Axes limits
max_val = max(abs(a), abs(b), abs(c)) + 2
ax.set_xlim([0, max_val])
ax.set_ylim([min(0, b) - 3, max_val])
ax.set_zlim([0, max_val])

# Labels
ax.set_xlabel('X-axis')
ax.set_ylabel('Y-axis')
ax.set_zlabel('Z-axis')
ax.set_title('Direction␣Cosines␣and␣Angles␣of␣Vector␣3i␣-␣2j␣+␣6k')
ax.legend()

plt.tight_layout()

# Save the figure
plt.savefig('direction_cosines_vector.png', dpi=300)
print("Plot␣saved␣as␣'direction_cosines_vector.png'")

plt.show()
```