4.5.11

EE25BTECH11009 - Anshu Kumar Ram

October 7, 2025

Question

Find the cartesian equation of the line which passes through the point $(-2, 4, -5)$ and parallel to the line

$$\frac{x+3}{3} = \frac{y-4}{5} = \frac{z+8}{6} \tag{1}$$

Solution - Vector Form

From the given line, the direction vector is

$$\mathbf{m} = \begin{pmatrix} 3 \\ 5 \\ 6 \end{pmatrix} \tag{2}$$

The required line passes through

$$\mathbf{A} = \begin{pmatrix} -2 \\ 4 \\ -5 \end{pmatrix} \tag{3}$$

So, the vector equation is

$$\mathbf{r} = \mathbf{A} + \lambda \mathbf{m}, \quad \lambda \in \mathbb{R} \tag{4}$$

In matrix form,

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} -2 \\ 4 \\ -5 \end{pmatrix} + \lambda \begin{pmatrix} 3 \\ 5 \\ 6 \end{pmatrix} \tag{5}$$

Solution - Cartesian Form

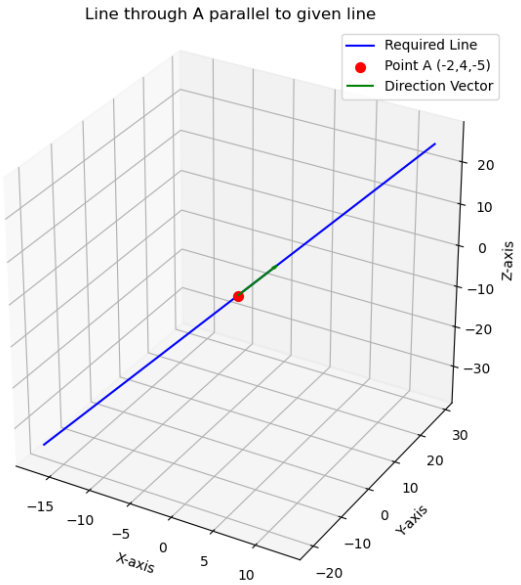
Eliminating λ,

$$\frac{x+2}{3} = \frac{y-4}{5} = \frac{z+5}{6} \tag{6}$$

Hence, the required cartesian equation of the line is

$$\boxed{\frac{x+2}{3} = \frac{y-4}{5} = \frac{z+5}{6}} \tag{7}$$

Figure


Line through A parallel to given line
Required Line
Point A (-2,4,-5)
Direction Vector
X-axis
Y-axis
Z-axis
−15
−10
−5
0
5
10
−20
−10
0
10
20
30
−30
−20
−10
0
10
20

C Code - Part 1

```c
#include <stdio.h>

// Function to fill point A
void get_point(int *A) {
    A[0] = -2; A[1] = 4; A[2] = -5;
}
```

C Code - Part 2

```c
// Function to fill direction vector
void get_direction(int *m) {
    m[0] = 3; m[1] = 5; m[2] = 6;
}
```

Python + C Code - Setup

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

# Load the shared library
lib = ctypes.CDLL("./func.so")

# Create arrays for results
A = (ctypes.c_int * 3)()
m = (ctypes.c_int * 3)()

# Call C functions
lib.get_point(A)
lib.get_direction(m)

# Convert to numpy
point_A = np.array([A[i] for i in range(3)])
direction_vector = np.array([m[i] for i in range(3)])

print("Point A:", point_A)
print("Direction vector:", direction_vector)
```

Python + C Code - Parametric Line

```
# --- Generate line points using parametric form ---
lam = np.linspace(-5, 5, 100)
x = point_A[0] + lam * direction_vector[0]
y = point_A[1] + lam * direction_vector[1]
z = point_A[2] + lam * direction_vector[2]
```

Python + C Code - Plotting

```python
fig = plt.figure(figsize=(8, 8))
ax = fig.add_subplot(111, projection='3d')

ax.plot(x, y, z, label="Required Line", color="blue")
ax.scatter(point_A[0], point_A[1], point_A[2],
           color="red", s=50, label="Point A (-2,4,-5)")
ax.quiver(point_A[0], point_A[1], point_A[2],
          direction_vector[0], direction_vector[1], direction_vector[2],
          color="green", label="Direction Vector", arrow_length_ratio=0.1)

ax.set_title("Line through A parallel to given line")
ax.set_xlabel("X-axis")
ax.set_ylabel("Y-axis")
ax.set_zlabel("Z-axis")
ax.legend()
ax.set_box_aspect([1,1,1])
plt.savefig("../figs/Figure_2.png")
plt.show()
```

Pure Python Code - Setup

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

# Required line: passes through A(-2,4,-5)
point_A = np.array([-2, 4, -5])

# Given line: passes through P(-3,4,-8)
point_P = np.array([-3, 4, -8])

# Direction vector (same for both lines)
direction_vector = np.array([3, 5, 6])

print(f"Point A (Required Line): {point_A}")
print(f"Point P (Given Line): {point_P}")
print(f"Direction Vector (m): {direction_vector}")


```

Pure Python Code - Plotting

```python
# --- Parametric equations ---
lam = np.linspace(-5, 5, 100)
x_A = point_A[0] + lam * direction_vector[0]
y_A = point_A[1] + lam * direction_vector[1]
z_A = point_A[2] + lam * direction_vector[2]
x_P = point_P[0] + lam * direction_vector[0]
y_P = point_P[1] + lam * direction_vector[1]
z_P = point_P[2] + lam * direction_vector[2]
fig = plt.figure(figsize=(8, 8))

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

# Plot the required line (Blue)
ax.plot(x_A, y_A, z_A, label='Required Line (through A)', color='blue')
ax.scatter(point_A[0], point_A[1], point_A[2],
           color='purple', s=50, label='Point A (-2,4,-5)')
ax.quiver(point_A[0], point_A[1], point_A[2],
          direction_vector[0], direction_vector[1], direction_vector[2],
          color='green', arrow_length_ratio=0.1, label='Direction Vector
          ↪  at A')
```

Pure Python Code - Plotting

```python
# Plot the given line (Red dashed)
ax.plot(x_P, y_P, z_P, '--', label='Given Line (through P)', color='red')
ax.scatter(point_P[0], point_P[1], point_P[2],
           color='orange', s=50, label='Point P (-3,4,-8)')
ax.quiver(point_P[0], point_P[1], point_P[2],
          direction_vector[0], direction_vector[1], direction_vector[2],
          color='black', arrow_length_ratio=0.1, label='Direction Vector
          ↪  at P')

ax.set_title('Required Line and Given Line (Parallel)')
ax.set_xlabel('X-axis')
ax.set_ylabel('Y-axis')
ax.set_zlabel('Z-axis')
ax.legend()
ax.set_box_aspect([1,1,1])
plt.savefig("../figs/Figure_2.png")
plt.show()
```