4.10.21

EE25BTECH11019 – Darji Vivek M.

Question:

Prove that the line through $A(0,-1,-1)$ and $B(4,5,1)$ intersects the line through $C(3,9,4)$ and $D(-4,4,4)$.

Solution

Matrix Method:

$$\mathbf{A} = \begin{pmatrix} 0 \\ -1 \\ -1 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} 4 \\ 5 \\ 1 \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} 3 \\ 9 \\ 4 \end{pmatrix}, \quad \mathbf{D} = \begin{pmatrix} -4 \\ 4 \\ 4 \end{pmatrix}, \tag{1}$$

$$\mathbf{d}_1 = \mathbf{B} - \mathbf{A} = \begin{pmatrix} 4 \\ 6 \\ 2 \end{pmatrix}, \quad \mathbf{d}_2 = \mathbf{D} - \mathbf{C} = \begin{pmatrix} -7 \\ -5 \\ 0 \end{pmatrix}, \tag{2}$$

$$\mathbf{P}(\lambda) = \mathbf{A} + \lambda \mathbf{d}_1, \quad \mathbf{Q}(\mu) = \mathbf{C} + \mu \mathbf{d}_2, \tag{3}$$

$$\mathbf{P}(\lambda) = \mathbf{Q}(\mu) \implies \lambda \mathbf{d}_1 - \mu \mathbf{d}_2 = \mathbf{C} - \mathbf{A}, \tag{4}$$

$$\mathbf{C} - \mathbf{A} = \begin{pmatrix} 3 \\ 10 \\ 5 \end{pmatrix}. \tag{5}$$

Solution

Formulating as a Matrix Equation:

$$\begin{pmatrix} \mathbf{d}_1 & -\mathbf{d}_2 \end{pmatrix} \begin{pmatrix} \lambda \\ \mu \end{pmatrix} = \mathbf{C} - \mathbf{A}. \tag{6}$$

Substituting the values,

$$\begin{pmatrix} 4 & 7 \\ 6 & 5 \\ 2 & 0 \end{pmatrix} \begin{pmatrix} \lambda \\ \mu \end{pmatrix} = \begin{pmatrix} 3 \\ 10 \\ 5 \end{pmatrix}. \tag{7}$$

Solution

Augmented matrix and row-reduction:

$$\left(\begin{array}{cc|c} 4 & 7 & 3 \\ 6 & 5 & 10 \\ 2 & 0 & 5 \end{array}\right) \xrightarrow{R_1 \leftrightarrow R_3,\ R_3 \to R_3 - 2R_1} \left(\begin{array}{cc|c} 2 & 0 & 5 \\ 6 & 5 & 10 \\ 4 & 7 & 3 \end{array}\right) \xrightarrow{R_1 \to \frac{1}{2}R_1,\ R_2 \to R_2 - 6R_1,\ R_3 \to R_3 - 4R_1} \left(\begin{array}{cc|c} 1 & 0 & \frac{5}{2} \\ 0 & 5 & -5 \\ 0 & 7 & -7 \end{array}\right) \xrightarrow{R_2 \to \frac{1}{5}R_2,\ R_3 \to R_3 - \frac{7}{5}R_2} \left(\begin{array}{cc|c} 1 & 0 & \frac{5}{2} \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{array}\right)$$

(8)

From the reduced system we read off

$$\lambda = \frac{5}{2}, \tag{9}$$

$$\mu = -1. \tag{10}$$

Intersection point:

$$\mathbf{P}\left(\tfrac{5}{2}\right) = \begin{pmatrix} 0 \\ -1 \\ -1 \end{pmatrix} + \frac{5}{2}\begin{pmatrix} 4 \\ 6 \\ 2 \end{pmatrix} = \begin{pmatrix} 10 \\ 14 \\ 4 \end{pmatrix}, \tag{11}$$

$$\mathbf{Q}(-1) = \begin{pmatrix} 3 \\ 9 \\ 4 \end{pmatrix} + (-1)\begin{pmatrix} -7 \\ -5 \\ 0 \end{pmatrix} = \begin{pmatrix} 10 \\ 14 \\ 4 \end{pmatrix}. \tag{12}$$

Therefore, the lines intersect at $\boxed{\begin{pmatrix} 10 \\ 14 \\ 4 \end{pmatrix}}$.

C Code

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

// Function: check intersection of line AB and line CD
// Returns 1 if they intersect, else 0
int line_intersection(double A[3], double B[3], double
    C[3], double D[3], double P[3]) {
    double d1[3], d2[3], rhs[3];
    double a11, a12, a21, a22, b1, b2, det;
    double lambda, mu;

    // direction vectors
    d1[0] = B[0] - A[0];
    d1[1] = B[1] - A[1];
    d1[2] = B[2] - A[2];

    d2[0] = D[0] - C[0];
    d2[1] = D[1] - C[1];
    d2[2] = D[2] - C[2];
```

C Code

```
  // rhs = C - A
  rhs[0] = C[0] - A[0];
  rhs[1] = C[1] - A[1];
  rhs[2] = C[2] - A[2];
// Build 2x2 system using dot products (Gram matrix)
  a11 = d1[0]*d1[0] + d1[1]*d1[1] + d1[2]*d1[2];
  a12 = d1[0]*d2[0] + d1[1]*d2[1] + d1[2]*d2[2];
  a21 = a12;
  a22 = d2[0]*d2[0] + d2[1]*d2[1] + d2[2]*d2[2];

  b1 = d1[0]*rhs[0] + d1[1]*rhs[1] + d1[2]*rhs[2];
  b2 = d2[0]*rhs[0] + d2[1]*rhs[1] + d2[2]*rhs[2];

  det = a11*a22 - a12*a21;
  if(fabs(det) < 1e-6) return 0; // parallel

  lambda = ( b1*a22 - b2*a12 ) / det;
  mu     = ( a11*b2 - a21*b1 ) / det;
```

C Code

```
  // Intersection point from line AB
    P[0] = A[0] + lambda*d1[0];
    P[1] = A[1] + lambda*d1[1];
    P[2] = A[2] + lambda*d1[2];

    // Point from line CD
    double Q[3];
    Q[0] = C[0] + mu*d2[0];
    Q[1] = C[1] + mu*d2[1];
    Q[2] = C[2] + mu*d2[2];

    // Check if P == Q
    if(fabs(P[0]-Q[0]) < 1e-6 && fabs(P[1]-Q[1]) < 1e
        -6 && fabs(P[2]-Q[2]) < 1e-6)
         return 1;
    return 0;
}
```

Python (Call)

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

# Load shared C library
lib = ctypes.CDLL("./8.so")

# Define argument and return types
lib.line_intersection.argtypes = [ctypes.POINTER(
    ctypes.c_double), ctypes.POINTER(ctypes.c_double),
                                  ctypes.POINTER(
                                      ctypes.c_double),
                                       ctypes.POINTER(
                                      ctypes.c_double),
                                  ctypes.POINTER(
                                      ctypes.c_double)]
lib.line_intersection.restype = ctypes.c_int
```

Python (Call)

```
# Points
A = np.array([0, -1, -1], dtype=np.double)
B = np.array([4,  5,  1], dtype=np.double)
C = np.array([3,  9,  4], dtype=np.double)
D = np.array([-4, 4,  4], dtype=np.double)
P = np.zeros(3, dtype=np.double)

# Call C function
res = lib.line_intersection(A.ctypes.data_as(ctypes.
    POINTER(ctypes.c_double)),
                            B.ctypes.data_as(ctypes.
                                POINTER(ctypes.c_double
                                )),
                            C.ctypes.data_as(ctypes.
                                POINTER(ctypes.c_double
                                )),
                            D.ctypes.data_as(ctypes.
                                POINTER(ctypes.c_double
                                )),
```

Python (Call)

```
                                    P.ctypes.data_as(ctypes.
                                        POINTER(ctypes.
                                        c_double)))

print("Intersect:", bool(res))
if res:
    print("Intersection Point:", P)

# ---- Plot ----
fig = plt.figure()
ax = fig.add_subplot(111, projection="3d")
# Line AB
ax.plot([A[0], B[0]], [A[1], B[1]], [A[2], B[2]], 'r',
    label="Line AB")
```

Python (Plot)

```
ax.text(*A, "A")
ax.text(*B, "B")

# Line CD
ax.plot([C[0], D[0]], [C[1], D[1]], [C[2], D[2]], 'b',
    label="Line CD")
ax.text(*C, "C")
ax.text(*D, "D")

# Intersection point
if res:
    ax.scatter(P[0], P[1], P[2], color='g', s=50,
        label="Intersection P")
    ax.text(*P, "P")

ax.legend()
plt.show()
```

Python Output and Plot

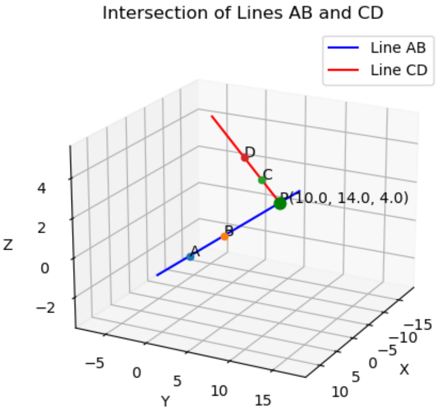

Figure: Given 2 lines are Intersecting