

1.2.16

Dhanush Kumar A - AI25BTECH11010

August 27, 2025

Question

Show that $(-1, 2, 1)$, $(1, -2, 5)$, $(4, -7, 8)$ and $(2, -3, 4)$ are the vertices of a parallelogram.

Variables used

Name	Point
Point A	$\begin{pmatrix} -1 \\ 2 \\ 1 \end{pmatrix}$
Point B	$\begin{pmatrix} 1 \\ -2 \\ 5 \end{pmatrix}$
Point C	$\begin{pmatrix} 4 \\ -7 \\ 8 \end{pmatrix}$
Point D	$\begin{pmatrix} 2 \\ -3 \\ 4 \end{pmatrix}$

Table: Variables Used

Solution

Now,

$$\mathbf{B} - \mathbf{A} = \begin{pmatrix} 1 - (-1) \\ -2 - 2 \\ 5 - 1 \end{pmatrix} = \begin{pmatrix} 2 \\ -4 \\ 4 \end{pmatrix}, \quad (1)$$

$$\mathbf{C} - \mathbf{D} = \begin{pmatrix} 4 - 2 \\ -7 - (-3) \\ 8 - 4 \end{pmatrix} = \begin{pmatrix} 2 \\ -4 \\ 4 \end{pmatrix}. \quad (2)$$

Hence,

$$\mathbf{B} - \mathbf{A} = \mathbf{C} - \mathbf{D} \quad (3)$$

\implies

$$\mathbf{C} - \mathbf{B} = \mathbf{D} - \mathbf{A} \quad (4)$$

Therefore, A, B, C, D are the vertices of a parallelogram.

Python code- Checking whether the points are vertices of parallelogram

```
import numpy as np
import itertools

def is_parallel(v, w, tol=1e-9):
    """Check if vectors v and w are parallel (cross product = 0)
    """
    v = np.array(v, dtype=float)
    w = np.array(w, dtype=float)
    if np.allclose(v, 0, atol=tol) or np.allclose(w, 0, atol=tol):
        return np.allclose(v, w, atol=tol)
    return np.allclose(np.cross(v, w), 0, atol=tol)

def is_parallelogram(points):
    """
    Check if 4 points form a parallelogram using only the parallel
    -sides test.
    Returns True if yes, else False.
    """
```

Python code - Checking whether the points are vertices of parallelogram

```
P = [np.array(p, dtype=float) for p in points]

for perm in itertools.permutations(range(4)):
    A, B, C, D = [P[i] for i in perm]

    AB, BC, CD, DA = B - A, C - B, D - C, A - D

# Check opposite sides are parallel and adjacent sides not
# parallel
if is_parallel(AB, CD) and is_parallel(BC, DA) and not
    is_parallel(AB, BC):
    return True
return False
```

Python code - Checking whether the points are vertices of parallelogram

```
# Example points
A = (-1, 2, 1)
B = ( 1, -2, 5)
C = ( 4, -7, 8)
D = ( 2, -3, 4)

points = [A, B, C, D]

if is_parallelogram(points):
    print("The given points form a parallelogram.")
else:
    print(" The given points do NOT form a parallelogram.")
```

Python code - plotting the points

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d.art3d import Poly3DCollection
import os

# Given points as column vectors (x,y,z)
A = np.array([-1, 2, 1]).reshape(-1,1)
B = np.array([1, -2, 5]).reshape(-1,1)
C = np.array([4, -7, 8]).reshape(-1,1)
D = np.array([2, -3, 4]).reshape(-1,1)

# Stack coordinates
coords = np.block([A,B,C,D])

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


Python code - plotting the points

```
# Scatter points
ax.scatter(coords[0,:], coords[1,:], coords[2,:], color='r', s
           =50)

# Draw parallelogram edges
edges = [(A,B), (B,C), (C,D), (D,A)]
for edge in edges:
    pts = np.hstack(edge)
    ax.plot(pts[0,:], pts[1,:], pts[2,:], color='b')

# Fill parallelogram face
verts = [[A.flatten(), B.flatten(), C.flatten(), D.flatten()]]
ax.add_collection3d(Poly3DCollection(verts, alpha=0.3, facecolor=
    'cyan'))
```

Python code - plotting the points

```
# Labels
ax.text(A[0,0], A[1,0], A[2,0], "A(-1,2,1)")
ax.text(B[0,0], B[1,0], B[2,0], "B(1,-2,5)")
ax.text(C[0,0], C[1,0], C[2,0], "C(4,-7,8)")
ax.text(D[0,0], D[1,0], D[2,0], "D(2,-3,4)")

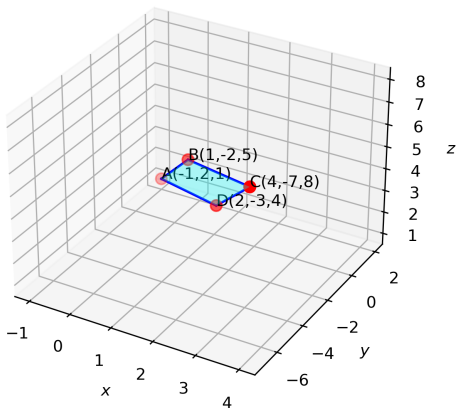
# Axes
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
ax.set_zlabel('$z$')
plt.title("Parallelogram in 3D")

# Save figure
save_path = '../figs/img.png'
os.makedirs(os.path.dirname(save_path), exist_ok=True)
plt.savefig(save_path, dpi=300)

print(f"Image saved at: {save_path}")
```

Plot-Using Python

Parallelogram in 3D



C code- writing the points

```
#include <stdio.h>
typedef struct { double x,y,z; } Point;
void save_points() {
    Point pts[4]={{-1,2,1},{1,-2,5},{4,-7,8},{2,-3,4}};
    FILE *f=fopen("points.dat","w");
    for(int i=0;i<4;i++) fprintf(f,"%f %f %f\n",pts[i].x,pts[i].y
        ,pts[i].z);
    fclose(f);
}
```

Python code -Ploting the points using c function

```
import numpy as np, matplotlib.pyplot as plt, os
from mpl_toolkits.mplot3d.art3d import Poly3DCollection

points=np.loadtxt("points.dat")
fig=plt.figure(); ax=fig.add_subplot(111,projection="3d")
ax.add_collection3d(Poly3DCollection([points],alpha=0.3,facecolor
    ="cyan"))
ax.scatter(points[:,0],points[:,1],points[:,2],color="red",s=50)
for i,(x,y,z) in enumerate(points): ax.text(x,y,z,f"P{i+1}",color
    ="black")
ax.set_xlabel("X"); ax.set_ylabel("Y"); ax.set_zlabel("Z")

save_path=" ../figs/img1.png"
os.makedirs(os.path.dirname(save_path),exist_ok=True)
plt.savefig(save_path,dpi=300)
plt.show()
```

Plot-Using Python and C

