2.10.55

Harsha-EE25BTECH11026

August 26,2025

Question

The edges of a parallelopiped are of unit length and are parallel to non-coplanar unit vectors $\hat{a}, \hat{b}, \hat{c}$ such that $\hat{a}.\hat{b} = \hat{b}.\hat{c} = \hat{c}.\hat{a} = \frac{1}{2}$. Then, the volume of the parallelopiped is

1. $\frac{1}{\sqrt{2}}$
2. $\frac{1}{2\sqrt{2}}$
3. $\frac{\sqrt{3}}{2}$
4. $\frac{1}{\sqrt{3}}$

Theoretical Solution

According to the question,the edges of the parallelopiped are parallel to the unit vectors $\hat{a}, \hat{b}, \hat{c}$ and

$$\hat{a}^T\hat{b} = \hat{b}^T\hat{c} = \hat{c}^T\hat{a} = \frac{1}{2}$$

Equation

As we know that the volume of parallelopiped is given by

$$V = [\mathbf{a}\ \mathbf{b}\ \mathbf{c}]$$

and

$$[\mathbf{a}\ \mathbf{b}\ \mathbf{c}][\mathbf{a}\ \mathbf{b}\ \mathbf{c}]^T = \mathbf{G}$$

where $\mathbf{G}$ is the Gram Matrix.

Theoretical Solution

$$\therefore \mathbf{G} = \begin{pmatrix} \hat{a}^T\hat{a} & \hat{a}^T\hat{b} & \hat{a}^T\hat{c} \\ \hat{b}^T\hat{a} & \hat{b}^T\hat{b} & \hat{b}^T\hat{c} \\ \hat{c}^T\hat{a} & \hat{c}^T\hat{b} & \hat{c}^T\hat{c} \end{pmatrix} = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & 1 \end{pmatrix}$$

For calculating the det($\mathbf{G}$), we can use the concept of eigen values.

Definition

Eigen values are those scalars which satisfies the following condition, For any non-zero eigen-vector $\mathbf{v}$ and coefficient matrix $\mathbf{M}$,

$$\mathbf{Mv} = \lambda\mathbf{v} \text{ , } \textit{where } \lambda \textit{ is an eigen value.}$$

Theoretical solution

$$\mathbf{G} = (1-\rho)\mathbf{I} + \rho\,\mathbf{1}\mathbf{1}^T, \text{ where } \rho = \frac{1}{2} \text{ and } \mathbf{1} = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$$

Let $\mathbf{1}\mathbf{1}^T = \mathbf{J}$. As we could see that the eigen-vector of $\mathbf{J}$ is $\mathbf{1}$ and by the rule,

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 3 \\ 3 \\ 3 \end{pmatrix} = 3\mathbf{1}$$

So, 3 is a eigen value of $\mathbf{J}$. Also we can observe that any vector orthogonal to $\mathbf{J}$ has eigen value 0 and since the eigen vector has only two degrees of freedom,

$$\therefore \textit{ eigen values of } \mathbf{J} \textit{ are } \{3, 0, 0\}$$

Theoretical Solution

Modifying the above equation on **G**,

$$\therefore \mathbf{Gv} = \frac{1}{2}\mathbf{Iv} + \frac{1}{2}\mathbf{Jv}$$

$$\implies G\mathbf{v} = \frac{(1+\mu)}{2}\mathbf{v}$$

where μ is the eigen value of **J**. Here the eigen value of **G** is $\frac{1+\mu}{2}$ and substituting the obtained eigen values of **J** in this equation, we get the eigen values of **G** to be $\{2, \frac{1}{2}, \frac{1}{2}\}$

Theoretical Solution

As we know that for eigen values of $\mathbf{G}$ being $\{\mu_1, \mu_2, \mu_3\}$

$$det(\mathbf{G}) = \mu_1\mu_2\mu_3$$

$$\therefore det(\mathbf{G}) = 2 \times \frac{1}{2} \times \frac{1}{2} = \frac{1}{2}$$

$$\implies V = \sqrt{det(\mathbf{G})} = \frac{1}{\sqrt{2}} \; units$$

C Code - Volume of parallelopiped

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

// Jacobi eigenvalue algorithm for 3x3 symmetric matrix
// Finds eigenvalues of G
void jacobi_eigenvalues(double G[3][3], double eigenvalues[3]) {
    double A[3][3];
    for(int i=0;i<3;i++) {
        for(int j=0;j<3;j++) A[i][j] = G[i][j];
    }
```

C Code - Volume of parallelopiped

```
double eps = 1e-12;
for(int iter=0; iter<100; iter++) {
    // find largest off-diagonal element
    int p=0,q=1;
    double max = fabs(A[0][1]);
    for(int i=0;i<3;i++) {
        for(int j=i+1;j<3;j++) {
            if(fabs(A[i][j]) > max) { max=fabs(A[i][j]); p=i;
                q=j; }
        }
    }
```

C Code - Volume of parallelopiped

```
if(max < eps) break;

double theta = 0.5 * atan2(2*A[p][q], A[q][q]-A[p][p]);
double c = cos(theta), s = sin(theta);

// rotate
double app = c*c*A[p][p] - 2*s*c*A[p][q] + s*s*A[q][q];
double aqq = s*s*A[p][p] + 2*s*c*A[p][q] + c*c*A[q][q];
A[p][q] = A[q][p] = 0.0;
A[p][p] = app;
A[q][q] = aqq;
```

C Code - Volume of parallelopiped

```
        for(int k=0;k<3;k++) {
            if(k!=p && k!=q) {
                double akp = c*A[k][p] - s*A[k][q];
                double akq = s*A[k][p] + c*A[k][q];
                A[k][p] = A[p][k] = akp;
                A[k][q] = A[q][k] = akq;
            }
        }
    }

    // diagonal entries are eigenvalues
    for(int i=0;i<3;i++) eigenvalues[i] = A[i][i];
}
```

C Code - Volume of parallelopiped

```c
// Volume from eigenvalues
double parallelepiped_volume(double G[3][3]) {
    double eigenvalues[3];
    jacobi_eigenvalues(G, eigenvalues);

    double det = eigenvalues[0] * eigenvalues[1] * eigenvalues
        [2];
    if(det < 0) det = -det;
    return sqrt(det);
}
```

Python+C Code

```
import ctypes
import matplotlib as mp
mp.use('TkAgg')
from mpl_toolkits.mplot3d.art3d import Poly3DCollection
import numpy as np
import matplotlib.pyplot as plt

lib = ctypes.CDLL("./libvolume.so")
lib.parallelepiped_volume.restype = ctypes.c_double
lib.parallelepiped_volume.argtypes = [ctypes.c_double * 3 * 3]

G = np.array([[1,0.5,0.5],
              [0.5,1,0.5],
              [0.5,0.5,1]], dtype=np.float64)

G_ctypes = (ctypes.c_double * 3 * 3)(*[(ctypes.c_double * 3)(*row
    ) for row in G])
```

Python+C Code

```
volume = lib.parallelepiped_volume(G_ctypes)
print("Volume =", round(volume,4),"units")

# Define three unit vectors with given dot products = 0.5
a = np.array([1, 0, 0])
b = np.array([0.5, np.sqrt(3)/2, 0])
c = np.array([0.5, 1/2*np.sqrt(3), np.sqrt(2/3)])
```

Python+C Code

```
# Vertices
O = np.array([0,0,0])
A = a
B = b
C = c
AB = a+b
AC = a+c
BC = b+c
ABC = a+b+c

vertices = [O, A, B, C, AB, AC, BC, ABC]
labels = ["O","A","B","C","A+B","A+C","B+C","A+B+C"]
```

Python+C Code

```
# Define faces
faces = [
    [O, A, AB, B],
    [O, A, AC, C],
    [O, B, BC, C],
    [A, AB, ABC, AC],
    [B, AB, ABC, BC],
    [C, AC, ABC, BC]
]

# Plot
fig = plt.figure(figsize=(9,7))
ax = fig.add_subplot(111, projection='3d')
```

Python+C Code

```
# Draw faces
poly3d = [[list(v) for v in face] for face in faces]
ax.add_collection3d(Poly3DCollection(poly3d, alpha=0.3, facecolor
    ='cyan'))

# Draw vectors
ax.quiver(0,0,0, a[0],a[1],a[2], color='r', linewidth=2, label='a
    ')
ax.quiver(0,0,0, b[0],b[1],b[2], color='g', linewidth=2, label='b
    ')
ax.quiver(0,0,0, c[0],c[1],c[2], color='b', linewidth=2, label='c
    ')

# Label vertices
for i, v in enumerate(vertices):
    ax.text(v[0], v[1], v[2], labels[i], fontsize=10, color='
        black')
```

Python+C Code

```
# Labels
ax.set_xlabel("X-axis")
ax.set_ylabel("Y-axis")
ax.set_zlabel("Z-axis")
ax.set_title("Parallelepiped formed by unit vectors a, b, c")

ax.set_box_aspect([1,1,1]) # equal aspect
ax.legend()
plt.savefig("/home/user/Matrix/Matgeo_assignments/2.10.55/figs/
    Figure-1.png")
plt.show()
```

Python Code

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as mp
mp.use('TkAgg')
from mpl_toolkits.mplot3d.art3d import Poly3DCollection

# Define three unit vectors with given dot products = 0.5
a = np.array([1, 0, 0])
b = np.array([0.5, np.sqrt(3)/2, 0])
c = np.array([0.5, 1/2*np.sqrt(3), np.sqrt(2/3)])
# Gram matrix
G = np.array([[np.dot(a,a), np.dot(a,b), np.dot(a,c)],
              [np.dot(b,a), np.dot(b,b), np.dot(b,c)],
              [np.dot(c,a), np.dot(c,b), np.dot(c,c)]])
```

Python Code

```
# Volume of parallelepiped
volume = np.sqrt(np.linalg.det(G))
print("Volume of parallelepiped =", round(volume,4))

# Vertices
O = np.array([0,0,0])
A = a
B = b
C = c
AB = a+b
AC = a+c
BC = b+c
ABC = a+b+c

vertices = [O, A, B, C, AB, AC, BC, ABC]
labels = ["O","A","B","C","A+B","A+C","B+C","A+B+C"]
```

Python Code

```
# Define faces
faces = [
    [O, A, AB, B],
    [O, A, AC, C],
    [O, B, BC, C],
    [A, AB, ABC, AC],
    [B, AB, ABC, BC],
    [C, AC, ABC, BC]
]

# Plot
fig = plt.figure(figsize=(9,7))
ax = fig.add_subplot(111, projection='3d')

# Draw faces
poly3d = [[list(v) for v in face] for face in faces]
ax.add_collection3d(Poly3DCollection(poly3d, alpha=0.3, facecolor
    ='cyan'))
```

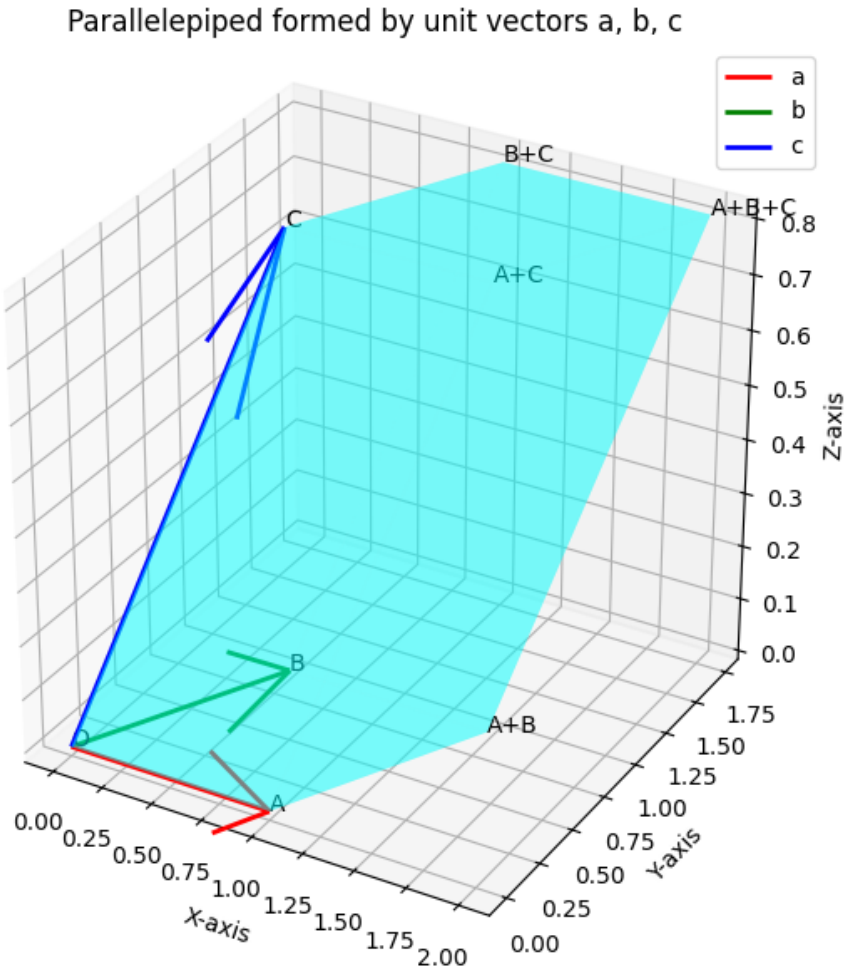
Python Code

```
# Draw vectors
ax.quiver(0,0,0, a[0],a[1],a[2], color='r', linewidth=2, label='a
    ')
ax.quiver(0,0,0, b[0],b[1],b[2], color='g', linewidth=2, label='b
    ')
ax.quiver(0,0,0, c[0],c[1],c[2], color='b', linewidth=2, label='c
    ')

# Label vertices
for i, v in enumerate(vertices):
    ax.text(v[0], v[1], v[2], labels[i], fontsize=10, color='
        black')
```

Python Code

```
# Labels
ax.set_xlabel("X-axis")
ax.set_ylabel("Y-axis")
ax.set_zlabel("Z-axis")
ax.set_title("Parallelepiped formed by unit vectors a, b, c")

ax.set_box_aspect([1,1,1]) # equal aspect
ax.legend()
plt.savefig("/home/user/Matrix/Matgeo_assignments/2.10.55/figs/
    Figure-1.png")
plt.show()
```

Plot


Parallelepiped formed by unit vectors a, b, c
a
b
c
B+C
A+B+C
C
A+C
B
A+B
O
A
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
X-axis
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
Y-axis
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
Z-axis