1.10.9

Hema Havil - EE25BTECH11050

September 16, 2025

Question

Find the unit vector in the direction of the vector PQ, where P and Q are the points (1, 2, 3) and (4, 5, 6), respectively.

Theoretical Solution

Given,
The points:

$$\mathbf{P} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \mathbf{Q} = \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix} \tag{1}$$

Let the required unit vector be $\mathbf{x}$, then
The formula for unit vector along a line joining two points

$$\mathbf{x} = \frac{\mathbf{X}}{\|\mathbf{X}\|} \tag{2}$$

The vector along $\mathbf{P}$ and $\mathbf{Q}$ is

$$\mathbf{X} = \mathbf{Q} - \mathbf{P} \tag{3}$$

Theoretical Solution

$$\mathbf{X} = \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix} - \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \tag{4}$$

$$\mathbf{X} = \begin{pmatrix} 4-1 \\ 5-2 \\ 6-3 \end{pmatrix} \tag{5}$$

$$\mathbf{X} = \begin{pmatrix} 3 \\ 3 \\ 3 \end{pmatrix} \tag{6}$$

Magnitude of the vector $\mathbf{X}$ is

$$\|\mathbf{X}\| = \sqrt{X^T X} \tag{7}$$

Theoretical Solution

$$\|\mathbf{X}\| = \sqrt{\begin{pmatrix} 3, 3, 3 \end{pmatrix} \begin{pmatrix} 3 \\ 3 \\ 3 \end{pmatrix}} \tag{8}$$

$$\|\mathbf{X}\| = \sqrt{(3)^2 + (3)^2 + (3)^2} \tag{9}$$

$$\|\mathbf{X}\| = \sqrt{3\,(3)^2} \tag{10}$$

$$\|\mathbf{X}\| = 3\sqrt{3} \tag{11}$$

Then the unit vector,

$$\mathbf{x} = \frac{1}{3\sqrt{3}}(\mathbf{X}) = \frac{1}{3\sqrt{3}}\begin{pmatrix}3\\3\\3\end{pmatrix} \tag{12}$$

$$\mathbf{x} = \frac{3}{3\sqrt{3}}\begin{pmatrix}1\\1\\1\end{pmatrix} \tag{13}$$

$$\mathbf{x} = \left(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\right) \tag{14}$$

Theoretical Solution

Therefore the required unit vector is

$$\mathbf{x} = \left(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\right)$$

C Code- Computing the unit vector

```c
// file: unitvec3d.c
#include <math.h>

#ifdef _WIN32
#define API __declspec(dllexport)
#else
#define API
#endif

// Compute unit vector from P to Q in 3D
// Inputs: P[3], Q[3]
// Output: unit[3]
// Returns: 0 on success, -1 if P=Q
```

C Code - Computing the unit vector

```
    API int unit_vector_3d(const double* P, const double* Q,
        double* unit) {
    double dx = Q[0] - P[0];
    double dy = Q[1] - P[1];
    double dz = Q[2] - P[2];
    double norm = sqrt(dx*dx + dy*dy + dz*dz);
    if (norm == 0.0) return -1;
    unit[0] = dx / norm;
    unit[1] = dy / norm;
    unit[2] = dz / norm;
    return 0;
}
```

Python Code using shared output

```
                import ctypes, os, math
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

# Load library
if os.name == nt:
    libname = unitvec3d.dll
else:
    libname = ./libunitvec3d.so
lib = ctypes.CDLL(libname)

# Function signature
lib.unit_vector_3d.argtypes = [
    ctypes.POINTER(ctypes.c_double),
    ctypes.POINTER(ctypes.c_double),
    ctypes.POINTER(ctypes.c_double),
]
```

Python Code using shared output

```
    lib.unit_vector_3d.restype = ctypes.c_int

def unit_vector_from_c(P, Q):
    P_arr = (ctypes.c_double * 3)(*P)
    Q_arr = (ctypes.c_double * 3)(*Q)
    U_arr = (ctypes.c_double * 3)()
    ret = lib.unit_vector_3d(P_arr, Q_arr, U_arr)
    if ret != 0:
        raise ValueError(P and Q coincide.)
    return [U_arr[0], U_arr[1], U_arr[2]]

# Example points
P = (1.0, 2.0, 3.0)
Q = (4.0, 5.0, 6.0)
```

Python Code using shared output

```
# Get unit vector
u = unit_vector_from_c(P, Q)
PQ = [Q[i]-P[i] for i in range(3)]
print(Unit vector:, u)

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

# Plot points
ax.scatter(*P, color=red, s=60, label=P)
ax.scatter(*Q, color=blue, s=60, label=Q)

# Vector PQ
ax.quiver(P[0], P[1], P[2], PQ[0], PQ[1], PQ[2], color=green,
    label=PQ)
```

Python Code using shared output

```
            # Unit vector (length 1)
ax.quiver(P[0], P[1], P[2], u[0], u[1], u[2], color=orange, label
    =Unit vector)

ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.legend()
plt.show()
```

Plot by python using shared output from c

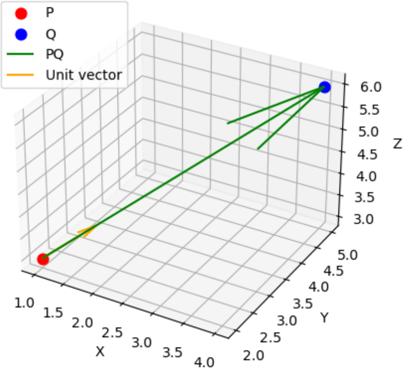

Figure: Plot of the unit vector along PQ

```
    import math
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

def unit_vector(P, Q):
    Compute unit vector from P to Q in 3D.
    dx, dy, dz = Q[0]-P[0], Q[1]-P[1], Q[2]-P[2]
    norm = math.sqrt(dx*dx + dy*dy + dz*dz)
    if norm == 0:
        raise ValueError(P and Q are the same point, unit vector
            undefined.)
    return (dx/norm, dy/norm, dz/norm)

# Example points
P = (1, 2, 3)
Q = (4, 5, 6)
```

Python code for the plot

```
# Compute vector PQ and unit vector
PQ = (Q[0]-P[0], Q[1]-P[1], Q[2]-P[2])
u = unit_vector(P, Q)

print(Vector PQ =, PQ)
print(Unit vector =, u)

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

# Plot points
ax.scatter(*P, color=red, s=60, label=P)
ax.scatter(*Q, color=blue, s=60, label=Q)

# Vector PQ (green arrow)
ax.quiver(P[0], P[1], P[2], PQ[0], PQ[1], PQ[2],
         color=green, label=PQ, arrow_length_ratio=0.1)
```

Python code for plot

```
    # Unit vector (orange arrow, length 1)
ax.quiver(P[0], P[1], P[2], u[0], u[1], u[2],
          color=orange, label=Unit vector, arrow_length_ratio=0.2)

# Labels and aesthetics
ax.set_xlabel(X)
ax.set_ylabel(Y)
ax.set_zlabel(Z)
ax.legend()
ax.set_title(Vector PQ and Unit Vector from P)
ax.grid(True)

plt.show()
```

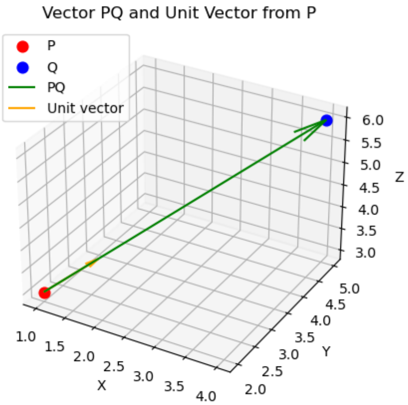

Figure: Plot for the unit vector along PQ