Problem 12.765

ee25btech11023-Venkata Sai

October 10, 2025

Problem

Let $\mathbf{v_1} = \begin{pmatrix} 1 \\ 2 \\ 0 \end{pmatrix}$ and $\mathbf{v_2} = \begin{pmatrix} 2 \\ 1 \\ 3 \end{pmatrix}$ be two vectors. The value of the coefficient α in the expression $\mathbf{v_1} = \alpha\mathbf{v_2} + \mathbf{e}$, which minimizes the length of the error vector $\mathbf{e}$, is

Formula

Given expression

$$\mathbf{v_1} = \alpha \mathbf{v_2} + \mathbf{e} \tag{3.1}$$

where $\mathbf{e}$ is the error vector
For any linear system $\mathbf{Ax} = \mathbf{B}$, the least squares solution formula is given by

$$\left(\mathbf{A}^\top \mathbf{A}\right) \mathbf{x} = \mathbf{A}^\top \mathbf{B} \tag{3.2}$$

$$\mathbf{x} = \left(\mathbf{A}^\top \mathbf{A}\right)^{-1} \mathbf{A}^\top \mathbf{B} \tag{3.3}$$

On writing the given expression as a linear system

$$\mathbf{v_2} \alpha = \mathbf{v_1} \tag{3.4}$$

where α being an 1×1 vector

Conclusion

$$\mathbf{A} = \mathbf{v_2}, \mathbf{B} = \mathbf{v_1} \tag{3.5}$$

$$\begin{aligned}
\alpha &= \left(\mathbf{v_2}^\top \mathbf{v_2}\right)^{-1} \mathbf{v_2}^\top \mathbf{v_1} && (3.6)\\
&= \left(\begin{pmatrix}2\\1\\3\end{pmatrix}^\top \begin{pmatrix}2\\1\\3\end{pmatrix}\right)^{-1} \begin{pmatrix}2\\1\\3\end{pmatrix}^\top \begin{pmatrix}1\\2\\0\end{pmatrix} && (3.7)\\
&= \left(\begin{pmatrix}2 & 1 & 3\end{pmatrix} \begin{pmatrix}2\\1\\3\end{pmatrix}\right)^{-1} \begin{pmatrix}2 & 1 & 3\end{pmatrix} \begin{pmatrix}1\\2\\0\end{pmatrix} && (3.8)\\
&= (4+1+9)^{-1}(2+2+0) && (3.9)\\
&= \frac{1}{14}(4) && (3.10)\\
&= \frac{2}{7} && (3.11)
\end{aligned}$$

C code

```c
void get_vectors(double* data) {
    data[0] = 1.0;
    data[1] = 2.0;
    data[2] = 0.0;
    data[3] = 2.0;
    data[4] = 1.0;
    data[5] = 3.0;
}
```

Python code for calling

```python
import ctypes
import numpy as np

def solve_least_squares():
    lib = ctypes.CDLL('./code.so')

    out_data = (ctypes.c_double * 6)()
    lib.get_vectors.argtypes = [ctypes.POINTER(ctypes.c_double)]
    lib.get_vectors(out_data)
    data = np.array(list(out_data))
    v1 = data[0:3]
    v2 = data[3:6]
    v2_dot_v2 = np.dot(v2, v2)
    v2_dot_v1 = np.dot(v2, v1)
    alpha = v2_dot_v1 / v2_dot_v2
    error_vec = v1 - (alpha * v2)
    return v1, v2, error_vec, alpha
```

Python code for plotting

```
import matplotlib.pyplot as plt
import numpy as np
from call import solve_least_squares

v1, v2, e, alpha = solve_least_squares()

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

ax.quiver(0, 0, 0, v1[0], v1[1], v1[2], color='r', label='Vector
    $\\vec{v_1}$')
ax.text(v1[0], v1[1], v1[2], ' $\\vec{v_1}$')

line_v2 = np.array([np.zeros(3), 1.5 * v2])
ax.plot(line_v2[:, 0], line_v2[:, 1], line_v2[:, 2], 'g--', alpha
    =0.5, label='Line of $\\vec{v_2}$')
ax.quiver(0, 0, 0, v2[0], v2[1], v2[2], color='g')
ax.text(v2[0], v2[1], v2[2], ' $\\vec{v_2}$')
projection_point = alpha * v2
```

Python code for plotting

```python
ax.quiver(projection_point[0], projection_point[1], projection_point[2],
        e[0], e[1], e[2],
        color='k', linestyle=':', label='Error Vector $\\vec{e}$')
ax.text(projection_point[0]+0.8, projection_point[1]-0.5, projection_point[2]+0.5, '$\\vec{e}$')
ax.set_title('Least Squares Error Vector')
ax.set_xlabel('X'); ax.set_ylabel('Y'); ax.set_zlabel('Z')
ax.legend()
ax.grid(True)
ax.axis('equal')
plt.show()
```