5.4.13

EE25BTECH11018 - Darisy Sreetej

October 4, 2025

Question

Using elementary transformations, find the inverse of the following matrix.

$$\begin{pmatrix} 1 & 3 \\ 2 & 7 \end{pmatrix}$$

Solution

Given

$$\mathbf{M} = \begin{pmatrix} 1 & 3 \\ 2 & 7 \end{pmatrix} \tag{1}$$

Let $\mathbf{M}^{-1}$ be the inverse of $\mathbf{M}$.Then

$$\mathbf{M}\mathbf{M}^{-1} = \mathbf{I} \tag{2}$$

Augmented matrix of $\left(\mathbf{M} \mid \mathbf{I}\right)$ is given by

$$\left(\begin{array}{cc|cc} 1 & 3 & 1 & 0 \\ 2 & 7 & 0 & 1 \end{array}\right) \xrightarrow{R_2 \to R_2 - 2R_1} \left(\begin{array}{cc|cc} 1 & 3 & 1 & 0 \\ 0 & 1 & -2 & 1 \end{array}\right) \xrightarrow{R_1 \to R_1 - 3R_2} \left(\begin{array}{cc|cc} 1 & 0 & 7 & -3 \\ 0 & 1 & -2 & 1 \end{array}\right) \tag{3}$$

Hence the inverse of the matrix $\begin{pmatrix} 1 & 3 \\ 2 & 7 \end{pmatrix}$ is $\begin{pmatrix} 7 & -3 \\ -2 & 1 \end{pmatrix}$

C code

```c
#include <stdio.h>

#define N 2 // Matrix size

void inverse(double A[N][N], double inv[N][N]) {
    double aug[N][2*N];

    // Forming augmented matrix [A | I]
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            aug[i][j] = A[i][j];
            aug[i][j + N] = (i == j) ? 1 : 0;
        }
    }
```

C Code

```c
    // Applying GaussJordan elimination
for (int i = 0; i < N; i++) {
    double pivot = aug[i][i];
    for (int j = 0; j < 2*N; j++) {
        aug[i][j] /= pivot; // Make pivot element = 1
    }

    for (int k = 0; k < N; k++) {
        if (k != i) {
            double factor = aug[k][i];
            for (int j = 0; j < 2*N; j++) {
                aug[k][j] -= factor * aug[i][j];
            }
        }
    }
}
```

```c
    // Obtaining inverse matrix
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            inv[i][j] = aug[i][j + N];
        }
    }
}
```

Python + C code

```
import ctypes
import numpy as np
import sympy as sp

# Load the compiled shared library (.so file)
lib = ctypes.CDLL("./inverse_matrix.so")

# Define the argument types for the inverse function
lib.inverse.argtypes = [
    ctypes.POINTER((ctypes.c_double * 2) * 2),
    ctypes.POINTER((ctypes.c_double * 2) * 2)
]
# Define the 2x2 input matrix A
A = np.array([[1, 3],
              [2, 7]], dtype=np.double)
```

```python

# Prepare an empty matrix for the inverse
inv = np.zeros((2, 2), dtype=np.double)

# Call the C function: inverse(A, inv)
lib.inverse(
    A.ctypes.data_as(ctypes.POINTER((ctypes.c_double * 2) * 2)),
    inv.ctypes.data_as(ctypes.POINTER((ctypes.c_double * 2) * 2))
)

# Convert the result to a Sympy Matrix for clean display
inverse = sp.Matrix(inv)

print("Inverse of the matrix:")
sp.pprint(inverse)
```

Python code

```python
import sympy as sp

# Define the matrix
A = sp.Matrix([[1, 3],
               [2, 7]])

# Find the inverse using Sympy
A_inv = A.inv()

# Display the result neatly
print("Inverse of the matrix:")
sp.pprint(A_inv)
```