Inverse of a Matrix Using Elementary Transformations

EE25BTECH11008 - Anirudh M Abhilash

October 4, 2025

Problem Statement

Find the inverse of the matrix

$$A = \begin{pmatrix} 2 & 3 \\ 1 & 4 \end{pmatrix}$$

using elementary transformations.

Solution

$$AA^{-1} = I,$$

We write the augmented matrix of A with the identity matrix:

$$[A|I] = \left(\begin{array}{cc|cc} 2 & 3 & 1 & 0 \\ 1 & 4 & 0 & 1 \end{array}\right).$$

Solution (cont..)

Step 1:

$$R_1 \to \frac{1}{2}R_1$$

$$\left(\begin{array}{cc|cc} 1 & 3/2 & 1/2 & 0 \\ 1 & 4 & 0 & 1 \end{array}\right).$$

Step 2:

$$R_2 \to R_2 - R_1$$

$$\left(\begin{array}{cc|cc} 1 & 3/2 & 1/2 & 0 \\ 0 & 5/2 & -1/2 & 1 \end{array}\right).$$

Solution (cont..)

Step 3:

$$R_2 \to \frac{2}{5}R_2$$

$$\left(\begin{array}{cc|cc} 1 & 3/2 & 1/2 & 0 \\ 0 & 1 & -1/5 & 2/5 \end{array}\right).$$

Step 4:

$$R_1 \to R_1 - \frac{3}{2}R_2$$

$$\left(\begin{array}{cc|cc} 1 & 0 & 4/5 & -3/5 \\ 0 & 1 & -1/5 & 2/5 \end{array}\right).$$

Solution (cont..)

Hence, the inverse of A is

$$A^{-1} = \begin{pmatrix} 4/5 & -3/5 \\ -1/5 & 2/5 \end{pmatrix}.$$

C Code (Inverse)

```
#include <stdio.h>
#include <stdlib.h>

void inverse(double *mat, double *inv, int n) {
    int i, j, k;
    double temp;
    double **aug = (double **)malloc(n * sizeof(double *));
    for (i = 0; i < n; i++) {
        aug[i] = (double *)malloc(2 * n * sizeof(double));
        for (j = 0; j < n; j++) {
            aug[i][j] = mat[i*n + j];
            aug[i][j+n] = (i == j) ? 1.0 : 0.0;
        }
    }
```

C Code (Cont..)

```
for (i = 0; i < n; i++) {
    temp = aug[i][i];
    for (j = 0; j < 2*n; j++)
        aug[i][j] /= temp;

    for (k = 0; k < n; k++) {
        if (k != i) {
            temp = aug[k][i];
            for (j = 0; j < 2*n; j++)
                aug[k][j] -= temp * aug[i][j];
        }
    }
}
```

C Code (Cont..)

```
    for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
            inv[i*n + j] = aug[i][j+n];

    for (i = 0; i < n; i++)
        free(aug[i]);
    free(aug);
}
```

Python Code (Using C)

```
import ctypes
import numpy as np

lib = ctypes.CDLL(’./inv.so’)

lib.inverse.argtypes = [ctypes.POINTER(ctypes.c_double),
                        ctypes.POINTER(ctypes.c_double),
                        ctypes.c_int]
lib.inverse.restype = None

A = np.array([[2, 3],
              [1, 4]], dtype=np.float64)
n = A.shape[0]
```

Python Code (Cont..)

```
A_inv = np.zeros((n, n), dtype=np.float64)

lib.inverse(A.ctypes.data_as(ctypes.POINTER(ctypes.c_double)),
            A_inv.ctypes.data_as(ctypes.POINTER(ctypes.
                c_double)),
            n)

print("Original matrix:")
print(A)
print("Inverse matrix:")
print(A_inv)
```