5.5.11

Vaishnavi - EE25BTECH11059

October 2, 2025

Question

Find inverse of the following matrix,using elementary transformation

$$A = \begin{pmatrix} 2 & 0 & -1 \\ 5 & 1 & 0 \\ 0 & 1 & 3 \end{pmatrix}$$

Solution

Construct the augmented matrix of A and I

$$\left(\begin{array}{ccc|ccc} 2 & 0 & -1 & 1 & 0 & 0 \\ 5 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 3 & 0 & 0 & 1 \end{array}\right) \xrightarrow{R_1 \leftarrow \frac{1}{2}R_1} \left(\begin{array}{ccc|ccc} 1 & 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 5 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 3 & 0 & 0 & 1 \end{array}\right) \tag{1}$$

$$\xrightarrow{R_2 \leftarrow R_2 - 5R_1} \left(\begin{array}{ccc|ccc} 1 & 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 1 & \frac{5}{2} & -\frac{5}{2} & 1 & 0 \\ 0 & 1 & 3 & 0 & 0 & 1 \end{array}\right) \tag{2}$$

solution

$$\xrightarrow{R_3 \leftarrow R_3 - R_2} \left(\begin{array}{ccc|ccc} 1 & 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 1 & \frac{5}{2} & -\frac{5}{2} & 1 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{5}{2} & -1 & 1 \end{array}\right) \tag{3}$$

$$\xrightarrow{R_3 \leftarrow 2R_3} \left(\begin{array}{ccc|ccc} 1 & 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 1 & \frac{5}{2} & -\frac{5}{2} & 1 & 0 \\ 0 & 0 & 1 & 5 & -2 & 2 \end{array}\right) \tag{4}$$

Solutions

$$\xrightarrow{R_2 \leftarrow R_2 - \frac{5}{2}R_3} \left(\begin{array}{ccc|ccc} 1 & 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 1 & 0 & -15 & 6 & -5 \\ 0 & 0 & 1 & 5 & -2 & 2 \end{array}\right) \tag{5}$$

$$\xrightarrow{R_1 \leftarrow R_1 + \frac{1}{2}R_3} \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & 3 & -1 & 1 \\ 0 & 1 & 0 & -15 & 6 & -5 \\ 0 & 0 & 1 & 5 & -2 & 2 \end{array}\right) \tag{6}$$

Solutions

$$A^{-1} = \begin{pmatrix} 3 & -1 & 1 \\ -15 & 6 & -5 \\ 5 & -2 & 2 \end{pmatrix} \tag{7}$$

Python Code

```
import numpy as np

# Define the matrix A
A = np.array([[2, 0, -1],
              [5, 1, 0],
              [0, 1, 3]])

# Calculate the inverse of A
A_inv = np.linalg.inv(A)

print( Inverse of the matrix A is: )
print(A_inv)
```

C Code

```c
#include <stdio.h>

void invert_matrix(double A[3][3], double inv[3][3]) {
    int i, j, k;
    double aug[3][6];
    // Build augmented matrix [A | I]
    for (i = 0; i < 3; ++i) {
        for (j = 0; j < 3; ++j) {
            aug[i][j] = A[i][j];
            aug[i][j+3] = (i == j) ? 1.0 : 0.0;
        }
    }
```

C Code

```
    // Forward elimination
      for (i = 0; i < 3; ++i) {
          double f = aug[i][i];
          for (j = 0; j < 6; ++j)
              aug[i][j] /= f;
          for (k = 0; k < 3; ++k) {
              if(k != i) {
                  double f2 = aug[k][i];
                  for (j = 0; j < 6; ++j)
                      aug[k][j] -= f2 * aug[i][j];
              }
          }
      }
      // Extract inverse
      for (i = 0; i < 3; ++i)
          for(j = 0; j < 3; ++j)
              inv[i][j] = aug[i][j+3];
}
```

Python and C Code

```
import ctypes
import numpy as np

# Load shared library
lib = ctypes.CDLL('./code.so')

# Prepare arguments
A = np.array([[2, 0, -1],
              [5, 1, 0],
              [0, 1, 3]], dtype=np.float64)
A_inv = np.zeros((3, 3), dtype=np.float64)

# Set up argtypes for the C function
lib.invert_matrix.argtypes = [ctypes.POINTER(ctypes.
    c_double * 3 * 3), ctypes.POINTER(ctypes.c_double *
     3 * 3)]
```

Python and C Code

```
# Create ctypes pointers
A_ct = (ctypes.c_double * 3 * 3)(*A.flatten())
A_inv_ct = (ctypes.c_double * 3 * 3)()

# Call C function
lib.invert_matrix(ctypes.byref(A_ct), ctypes.byref(
    A_inv_ct))

# Convert result back to numpy
A_inv_result = np.array(A_inv_ct).reshape((3, 3))
print( Inverse matrix from C:\n , A_inv_result)
```