Matgeo Presentation - Problem 5.4.16

ee25btech11021 - Dhanush sagar

October 1, 2025

Problem Statement

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

$$\begin{pmatrix} 2 & 5 \\ 1 & 3 \end{pmatrix}$$

solution

Given

$$\mathbf{A} = \begin{pmatrix} 2 & 5 \\ 1 & 3 \end{pmatrix} \tag{0.1}$$

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

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{I} \tag{0.2}$$

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

$$\left(\begin{array}{cc|cc} 2 & 5 & 1 & 0 \\ 1 & 3 & 0 & 1 \end{array}\right) \tag{0.3}$$

Perform the elementary row operation $R_2 \to 2R_2 - R_1$ to eliminate the first column entry of R_2:

$$\left(\begin{array}{cc|cc} 2 & 5 & 1 & 0 \\ 1 & 3 & 0 & 1 \end{array}\right) \xrightarrow{R_2 \to 2R_2 - R_1} \left(\begin{array}{cc|cc} 2 & 5 & 1 & 0 \\ 0 & 1 & -1 & 2 \end{array}\right) \tag{0.4}$$

solution

Now eliminate the 5 above the (2,2) pivot by $R_1 \to R_1 - 5R_2$:

$$\left(\begin{array}{cc|cc} 2 & 5 & 1 & 0 \\ 0 & 1 & -1 & 2 \end{array}\right) \xrightarrow{R_1 \to R_1 - 5R_2} \left(\begin{array}{cc|cc} 2 & 0 & 6 & -10 \\ 0 & 1 & -1 & 2 \end{array}\right) \tag{0.5}$$

Finally make the leading entry of R_1 unity by $R_1 \to \frac{1}{2}R_1$:

$$\left(\begin{array}{cc|cc} 2 & 0 & 6 & -10 \\ 0 & 1 & -1 & 2 \end{array}\right) \xrightarrow{R_1 \to \frac{1}{2}R_1} \left(\begin{array}{cc|cc} 1 & 0 & 3 & -5 \\ 0 & 1 & -1 & 2 \end{array}\right) \tag{0.6}$$

Hence the inverse of the matrix $\begin{pmatrix} 2 & 5 \\ 1 & 3 \end{pmatrix}$ is

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

C Source Code:matrix gen.c

```
#include <stdio.h>

void get_matrix(double* mat) {
    mat[0] = 2;  mat[1] = 5;
    mat[2] = 1;  mat[3] = 3;
}
```

Python Script:inverse matrix.py

```
mport ctypes
import numpy as np
# Load the shared library
lib = ctypes.CDLL("./libmatrix.so")
# Define function signature: get_matrix(double* mat)
lib.get_matrix.argtypes = [ctypes.POINTER(ctypes.c_double)]
lib.get_matrix.restype = None
# Prepare a NumPy array (2x2) for the matrix
A = np.zeros((2, 2), dtype=np.double)
# Pass pointer to C function (as 1D flattened array)
lib.get_matrix(A.ctypes.data_as(ctypes.POINTER(ctypes.c_double
print("Matrix A =\n", A)
try:
    A_inv = np.linalg.inv(A)
    print("\nComputed A^{-1} =\n", A_inv)
except np.linalg.LinAlgError:
    print("Matrix is singular, no inverse exists.")
```