# 5.4.24

Jnanesh Sathisha karmar - EE25BTECH11029

October 10,2025

# Question:

Using elementary transformations,find the inverse of each of the following matrices

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

# Theoretical Solution

To find the inverse of a matrix $\mathbf{A}$, we use the Gauss-Jordan elimination method. We begin by creating an augmented matrix by placing the identity matrix $\mathbf{I}$ to the right of matrix $\mathbf{A}$, forming $(\mathbf{A}|\mathbf{I})$.

The augmented matrix for $\mathbf{A} = \begin{pmatrix} 2 & 1 \\ 4 & 2 \end{pmatrix}$ is:

$$\left(\mathbf{A} \mid \mathbf{I}\right) = \left(\begin{array}{cc|cc} 2 & 1 & 1 & 0 \\ 4 & 2 & 0 & 1 \end{array}\right) \tag{1}$$

# Theoretical Solution

The goal is to use elementary row operations to transform the left side of the augmented matrix into the identity matrix. The right side will then become the inverse, $\mathbf{A}^{-1}$. We perform the operation $R_2 \to R_2 - 2R_1$:

$$\left(\begin{array}{cc|cc} 2 & 1 & 1 & 0 \\ 4-2(2) & 2-2(1) & 0-2(1) & 1-2(0) \end{array}\right) \tag{2}$$

After performing the operation, the matrix becomes:

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

Because a row of zeros has appeared on the left-hand side, it is impossible to continue the process to form the identity matrix. This indicates that the original matrix $\mathbf{A}$ is singular (its determinant is zero). Therefore, the inverse of the matrix does not exist.

# C Code

```c
#include <stdio.h>
#include <math.h>
#define MAX 10
void print_matrix(int n,double mat[MAX][MAX]){
    for(int i=0;i<n;i++){
        for (int j=0;j<n;j++){
            printf("%lf",mat[i][j]);
        }
        printf("\n");
    }
}
int inverse_matrix(int n,double mat[MAX][MAX],double inverse[MAX
    ][MAX]){
    double augvec[MAX][MAX*2];
    for(int i=0;i<n;i++){
        for(int j=0;j<n;j++){
            augvec[i][j]=mat[i][j];
        }
```

# C Code

```
        for(int j=0;j<n;j++){
            augvec[i][j+n]=(i==j)?1:0;
        }
    }
    for(int i=0;i<n;i++){
        double pivot=augvec[i][i];
        if(pivot==0){
int max_row = i;
for (int k = i + 1; k < n; k++) {
    if (fabs(augvec[k][i]) > fabs(augvec[max_row][i])) {
        max_row = k;
    }
}
if (max_row != i) {
    for (int j = 0; j < 2 * n; j++) {
        double temp = augvec[i][j];
        augvec[i][j] = augvec[max_row][j];
        augvec[max_row][j] = temp;
    }
```

# C Code

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

    }
    for(int i=0;i<n;i++){
            for(int j=0;j<n;j++){
                inverse[i][j]=augvec[i][j+n];
            }
        }
    }
```

# C Code

```c
int main(){
    int n;
    printf("enter the size of the matrix=");
    scanf("%d",&n);
    printf("\n");
    double mat[MAX][MAX];
    for (int i=0;i<n;i++){
        for(int j=0;j<n;j++){
            scanf("%lf",&mat[i][j]);
        }
    }
    double inv[MAX][MAX];

    inverse_matrix(n,mat,inv);
    if(inverse_matrix(n,mat,inv)==1){
        print_matrix(n,inv);}
    if(inverse_matrix(n,mat,inv)==0){
        printf("The matrix is singular");
    }
```