

Linear Algebra

***Matrix:
Matrix Inverse***

Automotive Intelligence Lab.



Contents

- The matrix inverse
- Types of inverses and conditions for invertibility
- Computing the inverse
- The inverse is unique
- Moore-Penrose pseudoinverse
- Numerical stability of the inverse
- Geometric interpretation of the inverse
- Summary
- Code exercises

The matrix inverse

Inverse of Matrix A

■ Multiply A to produce identity matrix.

- ▶ Which is written as A^{-1} .
- ▶ Cancel a matrix into identity matrix.

■ Linearly transform a matrix into identity matrix.

- ▶ Matrix inverse
 - Contains linear transformation.
- ▶ Matrix multiplication
 - Mechanism of applying transformation to the matrix.

$$A^{-1}A = I$$

Inverse of matrix

Why We Need to Invert Matrices?

■ In the form $Ax = b$

- ▶ Already know A and b .
- ▶ Need to cancel matrix to solve x .

$$\begin{aligned}Ax &= b \\ A^{-1}Ax &= A^{-1}b \\ Ix &= A^{-1}b \\ x &= A^{-1}b\end{aligned}$$

General form to solve x

Types of inverses and conditions for invertibility

Type 1: Full Inverse

■ Means $A^{-1}A = AA^{-1} = \boxed{I}$

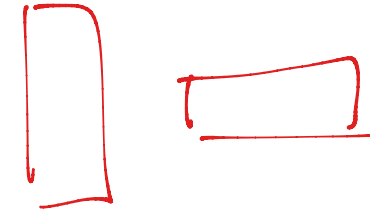
■ Two conditions

- ▶ Square
- ▶ Full-rank

Type 2: One Sided Inverse

■ Inverse of non-square matrix

- ▶ Can transform a rectangular matrix into identity matrix.
 - Works only for one multiplication order.
- ▶ A tall matrix T can have a left-inverse.
 - $LT = I$ but $TL \neq I$.
- ▶ A wide matrix W can have a right-inverse.
 - $WR = I$ but $RW \neq I$.

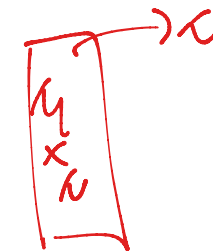


$$L \times T = I$$

■ Conditions of one-sided inverse

- ▶ Only if it has maximum possible rank.
- ▶ Non-square matrix size: $M \times N$
 - Tall matrix's rank should be N .
 - Wide matrix's rank should be M

Full rank 이어야 함.



Type 3: Pseudoinverse

■ Inverse for every matrix

- ▶ Every matrix has a pseudoinverse.
 - Regardless of its shape and rank.
- ▶ Square full rank matrix
 - Its pseudoinverse equals its full inverse.
- ▶ Nonsquare and its maximum possible rank
 - Tall matrix: pseudoinverse equals its left-inverse.
 - Wide matrix: pseudoinverse equals its right-inverse.
- ▶ Reduced-rank matrix
 - Has a pseudoinverse matrix.
 - Pseudoinverse transforms the singular into another matrix.
 - Close but not equal to the identity matrix.

■ Matrices that do not have a full or one-sided inverse are called **singular** or **noninvertible**.

- ▶ Same thing as labeling a matrix **reduced-rank** or rank-deficient.

Computing the inverse

Inverse of a 2×2 Matrix

How to invert 2×2 matrix?

- ▶ 1. Swap the diagonal elements.
- ▶ 2. Multiply the off-diagonal elements by -1.

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \xrightarrow{\text{Step 1}} \begin{bmatrix} d & b \\ c & a \end{bmatrix} \xrightarrow{\text{Step 2}} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

- ▶ 3. Divide by the determinant.

$$A^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$\begin{aligned} AA^{-1} &= \begin{bmatrix} a & b \\ c & d \end{bmatrix} \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} \\ &= \frac{1}{ad - bc} \begin{bmatrix} ad - bc & 0 \\ 0 & ad - bc \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \end{aligned}$$

Sims

Proof of inverse matrix

$$\begin{aligned} &\begin{bmatrix} 1 & 4 \\ 2 & 7 \end{bmatrix} \begin{bmatrix} 7 & -4 \\ -2 & 1 \end{bmatrix} \frac{1}{7 - 8} \\ &= \begin{bmatrix} (7 - 8) & (-4 + 4) \\ (14 - 14) & (-8 + 7) \end{bmatrix} \frac{1}{-1} \\ &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \end{aligned}$$

Numerical example

Code to Inverse of a 2×2 Matrix

■ Computing the inverse in MATLAB.

- ▶ $A * A_{inv}$ gives the identity matrix.

```
% Clear workspace, command window, and close all figures
clc; clear; close all;

% Create a 2x2 matrix
A = [[1 4]; [2 7]];

% Check if the matrix is invertible by determining if the
determinant is non-zero
if det(A) == 0
    error('The original matrix is singular and does not have an
inverse.');
```

✓

```
end

% Compute the inverse of the matrix
Ainv = inv(A);

% Compute the product of the original matrix and its inverse
(should be the identity matrix)
AAinv = A * Ainv;

% Visualize the original matrix with its values
visualizeMatrixWithValues(A, 'A');

% Visualize the inverse matrix with its values
```

A^{-1} , $1/A$, $\text{inv}(A)$

```
visualizeMatrixWithValues(Ainv, 'Ainv');

% Visualize the product of the original matrix and its inverse
with its values
visualizeMatrixWithValues(AAinv, 'A*Ainv');

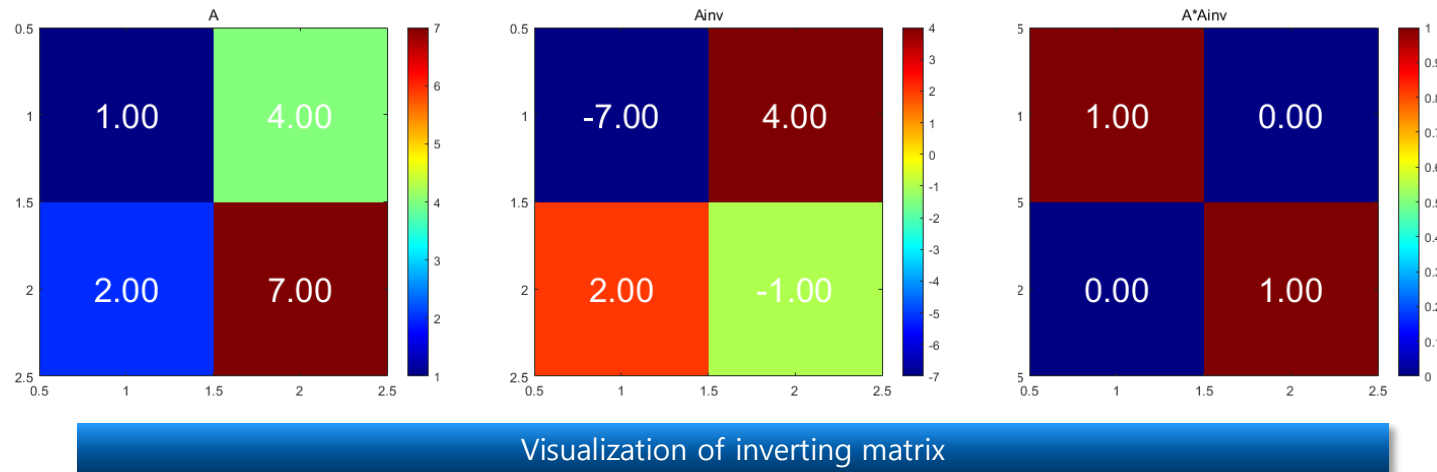
% Function definition : visualize matrix and value
function visualizeMatrixWithValues(matrix, titleStr)
    figure;
    imagesc(matrix);
    colorbar;
    title(titleStr);
    colormap jet;
    axis square;
    % Add text annotations for each element
    for i = 1:size(matrix, 1)
        for j = 1:size(matrix, 2)
            text(j, i, num2str(matrix(i, j)), '%.2f'), ...
                'HorizontalAlignment', 'center', ...
                'Color', 'white', 'FontSize', 25);
        end
    end
end
```

Code of inverting 2×2 matrix

Result of Inverse of a 2×2 Matrix

■ Result of computing the inverse matrix in MATLAB.

- ▶ $A * A_{inv}$ gives the identity matrix.




Another Example of Inverse of a 2×2 Matrix

■ Severe problems with this example.

▶ Matrix multiplication gives us 0 instead of ΔI .

- Determinant is zero.
- Cannot divide by zero.

$$\begin{bmatrix} 1 & 4 \\ 2 & 8 \end{bmatrix} \begin{bmatrix} 8 & -4 \\ -2 & 1 \end{bmatrix} \frac{1}{0} = \begin{bmatrix} (8-8) & (-4+4) \\ (16-16) & (-8+8) \end{bmatrix} \frac{1}{0} = ???$$


Another example of inverse of 2×2 matrix

■ What's different about the second example?

- ▶ It is a **reduced-rank matrix** ($rank = \boxed{1}$).
- ▶ Shows numerical example that reduced-rank matrices are not invertible.

MATLAB about Second Example

■ MATLAB won't even try to calculate the result like we did.

- ▶ It gives an error with following message.
- ▶ Reduced-rank matrices **do not have an inverse**.
 - Programs like MATLAB won't even try to calculate it.

```
% Create a 2x2 matrix
A = [[1 4]; [2 8]];

% Check if the matrix is invertible by determining if the determinant is non-zero
if det(A) == 0
    error('The original matrix is singular and does not have an inverse.');
```

MATLAB code of second case

다음 사용 중 오류가 발생함: AILAB_LA_figure_inverse_matrix_error (9번 라인)
The original matrix is singular and does not have an inverse.

MATLAB code error

Inverse of a Diagonal Matrix

■ Product of two diagonal matrices is **scalar multiplied** diagonal elements.

- ▶ Simply invert each diagonal element.
 - Ignoring off-diagonal zeros.
- ▶ Diagonal matrix with at least one zero on the diagonal has **no inverse**.
 - You'll have $1/0$.
 - Diagonal matrix is full-rank only if all diagonal elements are Non-zero

$$\begin{bmatrix} 1/b & 0 & 0 \\ 0 & 1/c & 0 \\ 0 & 0 & 1/d \end{bmatrix} \begin{bmatrix} b & 0 & 0 \\ 0 & c & 0 \\ 0 & 0 & d \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Example of inverting a diagonal matrix

Algorithms to Compute Inverse

■ Involves four intermediate matrices.

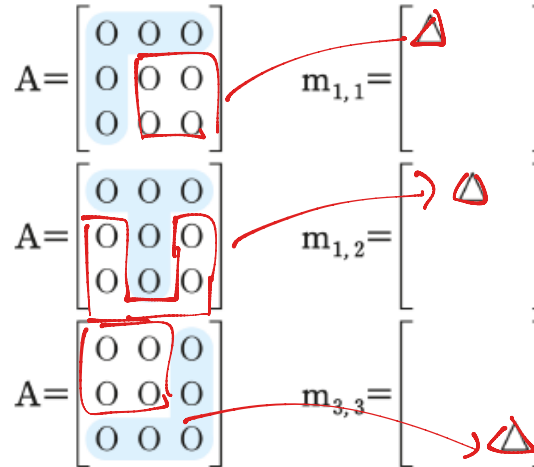
■ Minors matrix

- ▶ Comprise determinants of submatrices.
- ▶ Element $m_{i,j}$ of the minors matrix
 - Determinant of submatrix
 - Created by excluding i th row and j th column.

The **adjugate** of a matrix \mathbf{A} can be used to find the inverse of \mathbf{A} as follows:

If \mathbf{A} is an invertible matrix, then

$$\mathbf{A}^{-1} = \frac{1}{\det(\mathbf{A})} \text{adj}(\mathbf{A}).$$



Overview of procedure for 3×3 matrix

Other Algorithms to Compute Inverse

■ Grid matrix

- ▶ Checkerboard of alternating +1s and -1s.

$$g_{i,j} = -1^{i+j}$$

Formula of grid matrix

■ Cofactors matrix

- ▶ Hadamard multiplication of the minors matrix with grid matrix.

*LU decomposition
Inverse?*

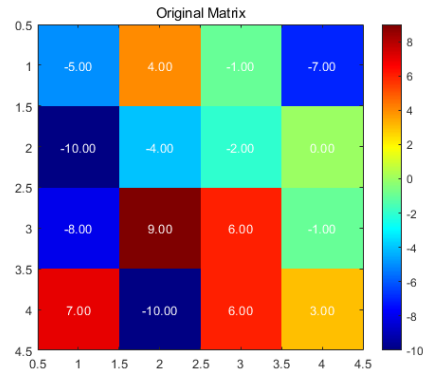
■ Adjugate matrix

- ▶ Transpose of cofactors matrix.
- ▶ Scalar multiplied by the inverse of the determinant of the original matrix.
- ▶ Adjugate matrix is inverse of the original matrix.

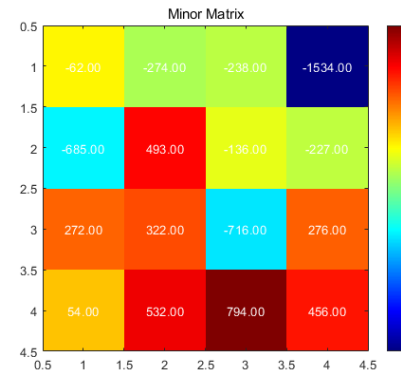
$$A^{-1} = \frac{1}{\det(A)} \text{adj}(A)$$

Visualization Result of Computing Various Matrices

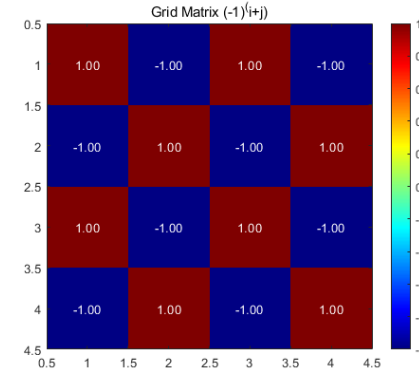
■ Implements of these matrices are Homework!



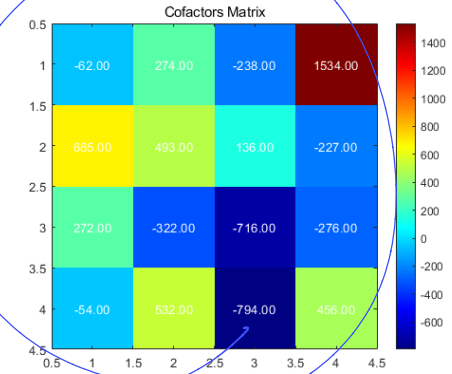
Original matrix



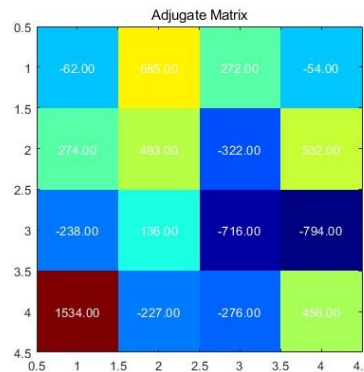
Minors matrix



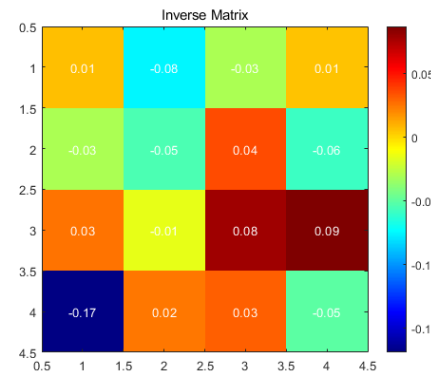
Grid matrix



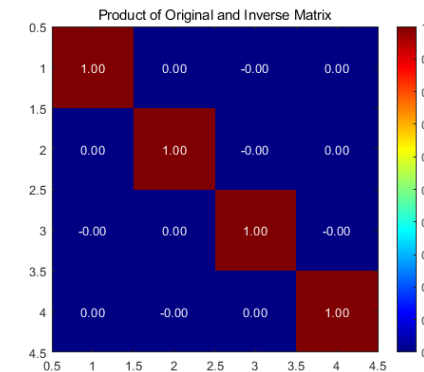
Cofactors matrix



Adjugate matrix



Inverse matrix



Original * Inverse

One-Sided Inverse

■ Tall matrix T of size $M > N$ doesn't have a full inverse.

- ▶ No tall matrix T^{-1} such that $TT^{-1} = T^{-1}T = I$.
- ▶ There is matrix L such that $LT = I$.

■ Goal is to find matrix L .

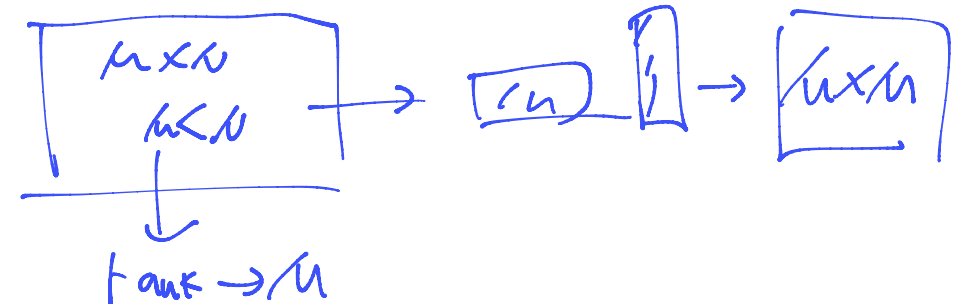
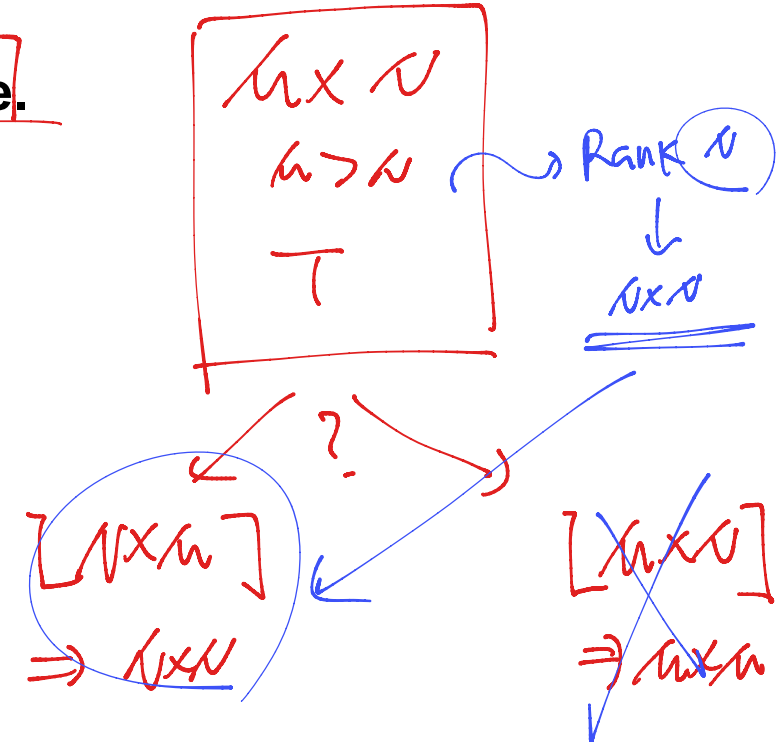
- ▶ Making matrix T square.
 - Multiplying it by its transpose.
 - Question: $T^T T$ ($N \times N$) or TT^T ($M \times M$)?
 - Both are square.
 - But $T^T T$ is **full rank** if T is full column-rank.
 - All square full-rank matrices have an inverse.
- ▶ Look for a matrix that left-multiplies T to produce the identity matrix.

$$(T^T T)^{-1} (T^T T) = I$$

$$L = (T^T T)^{-1} T^T$$

$$LT = I$$

- ▶ Matrix L is the left-inverse of matrix T .



Code Exercise of Calculating Left Inverse

Code Exercise (08_01)

- ▶ Get the left inverse matrix.
- ▶ Check whether left multiplying the original tall matrix(TL) produces the identity matrix.
- ▶ Check the result of post multiplying(LT).

```
% Clear workspace, command window, and close all figures
clc; clear; close all;

% Generate a random 40x4 matrix with integer elements from -10 to 10
T = randi([-10, 10], 40, 4);

% Compute the product of T transpose and T
TtT = T' * T;

% Compute the inverse of the product
TtT_inv = inv(TtT);

% Compute the Left inverse matrix
L = TtT_inv * T';

% Multiply the inverse with the original product to get the identity matrix
LT = L * T;
TL = T * L;

% Visualize the matrices
visualizeOriginalMatrix(T, 'Original Matrix T');
visualizeLeftInverseMatrix(L, 'L');
visualizeMatrix(LT, 'LT ');
visualizeMatrix(TL, 'TL ');

% Function definition : visualize matrix
function visualizeMatrix(matrix, titleStr)
```

```
figure;
imagesc(matrix); % Display the matrix as a color image
colorbar;        % Show a color scale
title(titleStr);
axis square;     % Make axes square
end

% Function definition : visualize original matrix
function visualizeOriginalMatrix(matrix, titleStr)
figure;
imagesc(matrix); % Display the matrix as a color image
colorbar;        % Show a color scale
title(titleStr);
axis([0.5 40 0 40]); % Make axes square
end

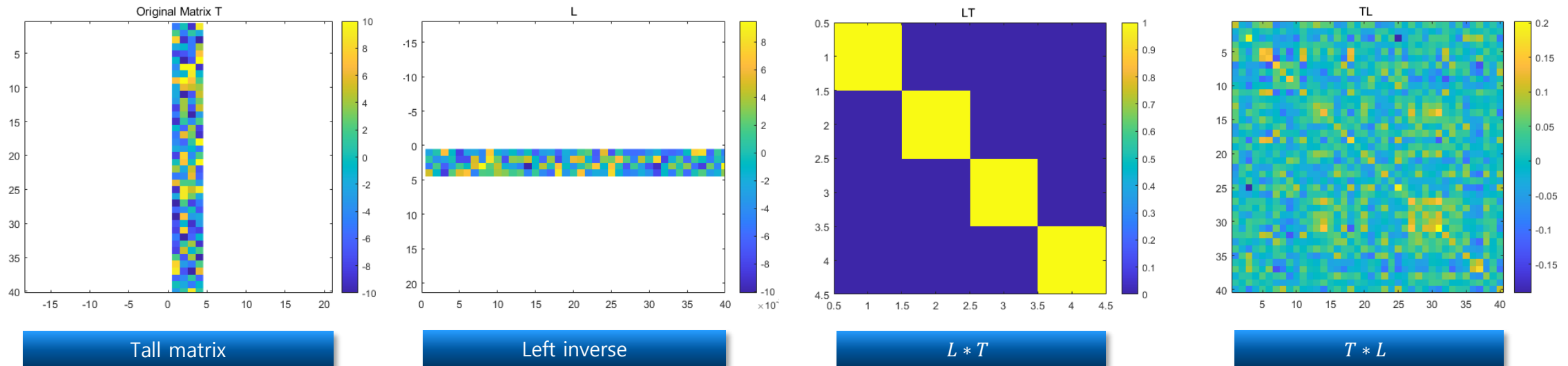
% Function definition : visualize left inverse matrix
function visualizeLeftInverseMatrix(matrix, titleStr)
figure;
imagesc(matrix); % Display the matrix as a color image
colorbar;        % Show a color scale
title(titleStr);
axis([0 40 0.5 40]); % Make axes square
end
```

MATLAB code example of calculating left inverse

Code to Calculate Left Inverse

■ Following figure illustrates a tall matrix, its left inverse, and two ways of multiplying left inverse by the matrix.

- ▶ Left matrix is not a square matrix
- ▶ Left multiplying by the left inverse is the identity matrix.
- ▶ Right multiplying by the left inverse is not the identity matrix.



About Left Inverse of matrix T

■ Left inverse is a one-sided inverse.

- ▶ LT is identity matrix. But TL is not.

■ Left inverse is defined only for tall matrices that have full column-rank.

- ▶ A matrix size $M > N$ with rank $r < N$ doesn't have a left-inverse.
- ▶ $T^T T$ is reduced-rank.
- ▶ Thus cannot be inverted.

■ How about right inverse?

- ▶ Try yourself.

The inverse is unique

Proof of Uniqueness of Inverse

■ Cannot be $AB = I$ and $AC = I$ while $B \neq C$.
??

► Several proofs of this claim.

- One is proof by negation.

- Try but fail to prove a false claim.
- Thereby proving the correct claim.

► Proof by negation: start with three assumptions.

- 1. Matrix A is invertible
- 2. Matrices B and C are inverses of A .
- 3. Matrices B and C are distinct.
 - Meaning $B \neq C$.
- Follow each expression from left to right.

$$\underline{C} = \underline{CI} = \underline{CAB} = \underline{IB} = \underline{B}$$

Proof by negation

$$C = B$$

- Assumption of $B \neq C$ is false.

► Conclusion: invertible matrix has **exactly one inverse**.

Moore-Penrose pseudoinverse

Pseudoinverse for Singular Matrices

■ **Reduced-rank matrices** do not have a full of a one-sided inverse.

- ▶ Impossible to transform a reduced-rank matrix into the identity matrix.
 - By matrix multiplication.

■ But singular matrices do have **pseudoinverses**.

■ **Pseudoinverses?**

- ▶ Transformation matrices that bring a matrix close to the identity matrix.
- ▶ Plural pseudoinverses was not a typo.
 - Reduced-rank matrix has an infinite number of pseudoinverse.

■ **Best pseudoinverse: Moore-Penrose pseudoinverse**

- ▶ Sometimes abbreviated as the MP pseudoinverse.

Moore-Penrose Pseudoinverse

■ Best pseudoinverse: Moore-Penrose pseudoinverse

- ▶ Can always assume that pseudoinverse refers to it.
- ▶ Following matrix is pseudoinverse of the singular matrix.
 - First line: pseudoinverse of the matrix.
 - Second line: product of the matrix and its pseudoinverse.
 - Scaling for 85: to facilitate visual inspection of the matrix.
 - Indicating pseudoinverse using dagger, plus sign, or asterisk.
 - $A^\dagger, A^+,$ or A^* .

$$\begin{bmatrix} 1 & 4 \\ 2 & 8 \end{bmatrix}^\dagger = \frac{1}{85} \begin{bmatrix} 1 & 2 \\ 4 & 8 \end{bmatrix}$$

$$\frac{1}{85} \begin{bmatrix} 1 & 4 \\ 2 & 8 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 4 & 8 \end{bmatrix} = \begin{bmatrix} .2 & .4 \\ .4 & .8 \end{bmatrix}$$

Example of pseudoinverse matrix

Code: Moore-Penrose Pseudoinverse

- Use function `pinv()` in MATLAB.

```
A = [[1 4]; [2 8]];
Apinv = pinv(A);
AApinv = A*Apinv;

disp("Apinv");
disp(Apinv);
disp("AApinv");
disp(AApinv);
```

MATLAB code to pseudoinverse

■ How is the pseudoinverse computed?

- ▶ Take `SVD` of a matrix.
- ▶ Invert nonzero singular values without changing singular vectors.
- ▶ Reconstruct matrix by multiplying $U\Sigma^+V^T$.
- ▶ If you don't understand it, don't worry.
 - It will be intuitive by the end of Lecture 14.

Numerical stability of the inverse

Complexity of Computing Matrix Inverse

■ Computing matrix inverse involves a lot of FLOPs (float-point operations)

- ▶ Matrix inverse includes many determinants.
- ▶ Computing many determinants can lead to numerical inaccuracies.
 - Accumulate and cause significant problems when working with large matrices.
- ▶ Low level libraries generally strive to avoid explicitly inverting matrices.
 - Or decompose matrices into the product of other matrices that are more numerically stable.

float → 32-bit
double → 64-bit

■ Matrices that have numerical values in roughly the same range tend to be more stable.

- ▶ Why random-numbers matrices are easy to work with.
- ▶ Matrices with a large range of numerical values.
 - Have a high risk of numerical instability.
- ▶ Range of numerical values?
 - Formally captured as the condition number of a matrix.
 - Condition number
 - Ratio of the largest to smallest singular value.
 - A measure of the spread of numerical values in a matrix.

Hilbert Matrix

■ Numerically unstable matrix

▶ Hilbert matrix is defined by the following formula.

- i and j are row and column indices.

$$h_{i,j} = \frac{1}{i + j - 1}$$

Formula to create a Hilbert matrix

$$\begin{bmatrix} 1 & 1/2 & 1/3 \\ 1/2 & 1/3 & 1/4 \\ 1/3 & 1/4 & 1/5 \end{bmatrix}$$

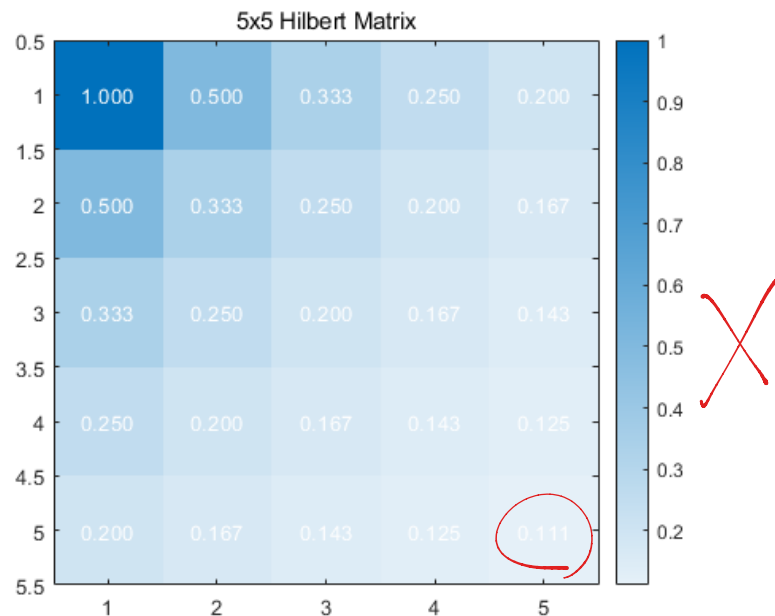
Example of a 3×3 Hilbert matrix

- ▶ As the matrix gets larger, ranges of numerical values increases.
- Computer calculated Hilbert matrix quickly becomes rank-deficient.

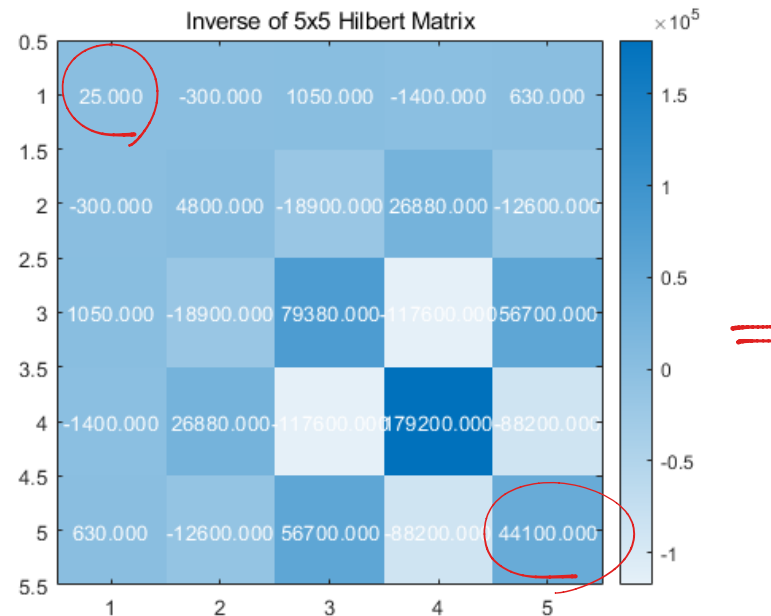
Inverse of Hilbert Matrix

■ Full-rank Hilbert matrices have inverse in a very different numerical range.

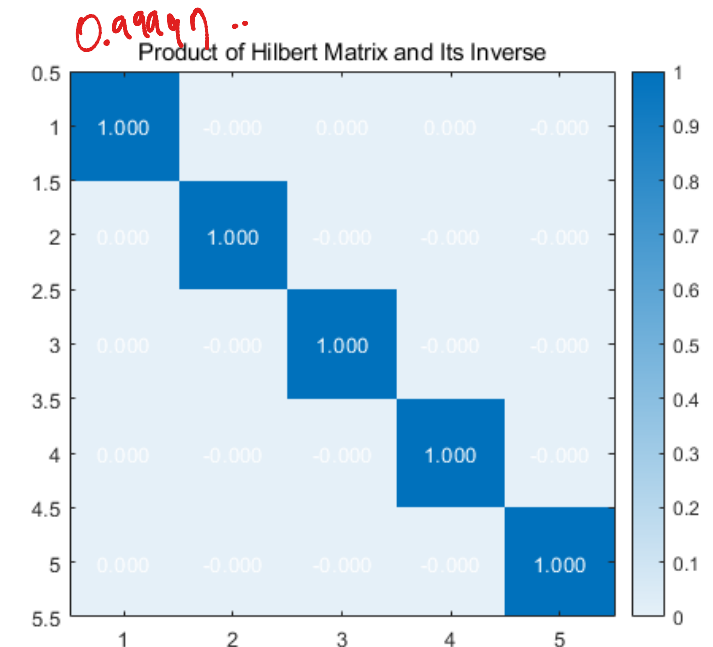
- ▶ Illustrated them below.
- ▶ Product matrix certainly looks like the identity matrix.
 - In reality, rounding error increases rapidly.
 - With the size of the matrix.



Hilbert matrix



Inv(Hilbert)



Their product

Geometric interpretation of the inverse

Geometric Transformation of Inverse

■ Matrix inverse: undoing geometric transformation imposed by multiplication.

► This geometric effect is unsurprising.

- Following equations

- P is $2 \times N$ matrix of original geometric coordinates.
- T is transformation matrix.
- Q is matrix of transformed coordinates.
- U is matrix of back-transformed coordinates.

- Interpretation of this equation: undoing the transform imposed by the matrix.

$$Q = TP$$

$$U = T^{-1}Q$$

$$U = T^{-1}TP$$

Math of geometric effect

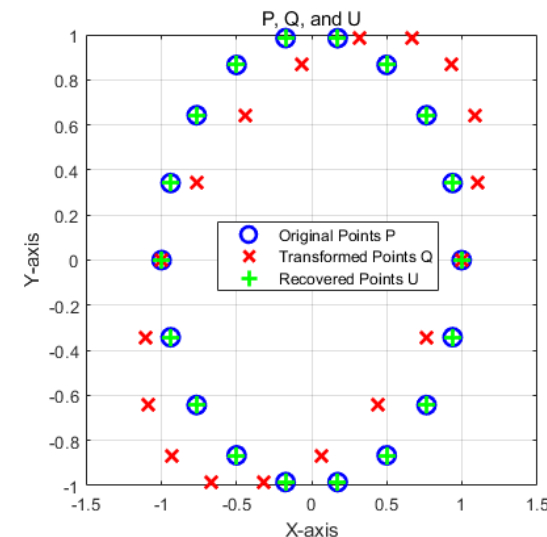


Figure of geometric effect

Code Exercise of Geometric Transformation of Inverse

Code Exercise (08_02)

- ▶ Generate the points on a circle.
- ▶ Define the Transformation matrix T and inverse matrix T^{-1} .
- ▶ Check the result of TP and $T^{-1}TP$.

```
% Clear workspace, command window, and close all figures
clc; clear; close all;

% Define angles 0 to 340 degrees in 20 degree increments
angles = 0:20:340;

% Convert degrees to radians for computation
radians = deg2rad(angles);

% Create points on a circle with radius 1
P = [cos(radians); sin(radians)];

% Define a transformation matrix T
T = [1 0.5; 0 1];

% Apply the transformation to create Q
Q = T * P;

% Calculate U, which should be the same as P
U = inv(T) * Q;

% Visualize the points P, Q, and U on the same plot for comparison
figure;
plot(P(1, :), P(2, :), 'bo', 'LineWidth', 2, 'MarkerSize', 10, 'DisplayName',
'Original Points P');
hold on; % Hold on to plot additional points
plot(Q(1, :), Q(2, :), 'rx', 'LineWidth', 2, 'MarkerSize', 10, 'DisplayName',
'Transformed Points Q');
plot(U(1, :), U(2, :), 'g+', 'LineWidth', 2, 'MarkerSize', 10, 'DisplayName',
'Recovered Points U');
legend show;
axis square;
grid on;
title('P, Q, and U');
xlabel('X-axis');
ylabel('Y-axis');
hold off;
```

MATLAB code example of geometric transformation of inverse

Application of Geometric Transformation of Inverse

- Come in handy when you learn about diagonalizing a matrix through eigen-decomposition.
- Provide intuition for why reduced-rank matrix has no inverse.
 - ▶ Geometric effect of transforming by singular matrix
 - At least one dimension is flattened.
 - Once a dimension flattened, it cannot be unflattened.
 - Like you cannot see your back when facing a mirror.

Summary

Summary

■ Matrix inverse

- ▶ A matrix that transforms a maximum-rank matrix into the identity matrix.
 - Through matrix multiplication.
- ▶ Has many purposes.
 - Including moving matrices around in an equation(e.g., solve for x in $Ax = b$).

■ Square full-rank matrix

- ▶ Has a full inverse.

■ Tall full column-rank matrix

- ▶ Has a left-inverse.

■ Wide full row-rank matrix

- ▶ Has a right-inverse.

■ Reduced-rank matrices cannot be linearly transformed into the identity matrix.

- ▶ But they do have a pseudoinverse.
- ▶ Pseudoinverse transforms matrix into another matrix that is closer to the identity matrix.

Summary

- **Inverse is unique.**

- ▶ If a matrix can be linearly transformed into the identity matrix, there is only one way to do it.

- **Some tricks for computing the inverses of some kinds of matrices.**

- ▶ Including 2×2 and diagonal matrices.
 - ▶ These shortcuts are simplifications of the full formula for computing a matrix inverse.

- **Due to the risk of numerical precision errors, production-level algorithms try to avoid explicitly inverting matrices or will decompose a matrix into other matrices that can be inverted with greater numerical stability.**

Code exercises

Hand Made Inverse Algorithm

- Implement the full-inverse algorithm for a 2×2 matrix using matrix elements a, b, c , and d . (not using inverse function)
- Hint: Remember that the determinant of a scalar is its absolute value.

```
% Clear workspace, command window, and close all figures
clc; clear; close all;

% Assume that input value is [a,b;c,d] (2x2 matrix)
% then make a function that return a inverse matrix
% NOTE: Do not use inverse function in MATLAB

% input1
a1 = 2;
b1 = 1;
c1 = 6;
d1 = 8;

% input2
a2 = 4;
b2 = 10;
c2 = 2;
d2 = 5;

% Write here (start)
% Create error when inverse matrix does not exist
function inverse_matrix = handMadeInverseMatrix(a, b, c, d)

% Compute the determinant
% Check error
if
    error('Matrix is not invertible');
end
% Compute the inverse manually

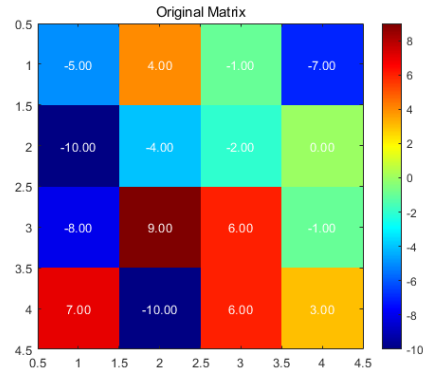
end
% Write here (end)

% Call the function and store the inverse matrix
% Display the result
inv1 = handMadeInverseMatrix(a1,b1,c1,d1);
disp('Inverse matrix1:');
disp(inv1);
inv2 = handMadeInverseMatrix(a2,b2,c2,d2);
```

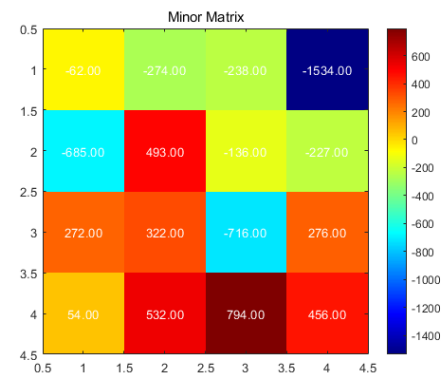
Sample code

Implements of Various Matrices

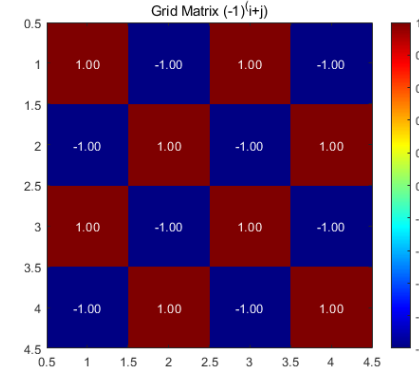
■ Implements of these matrices



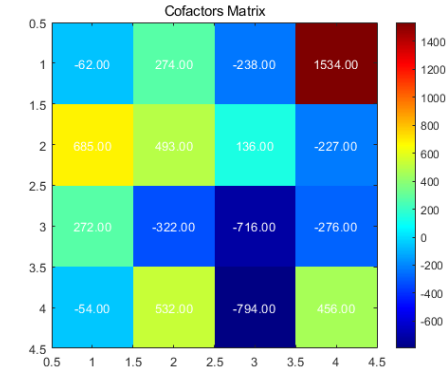
Original matrix



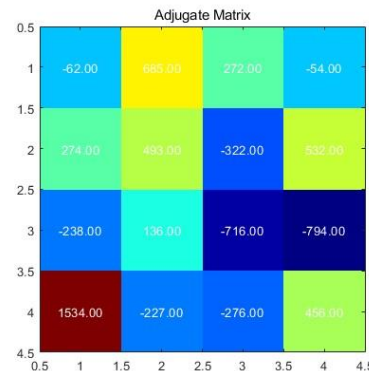
Minors matrix



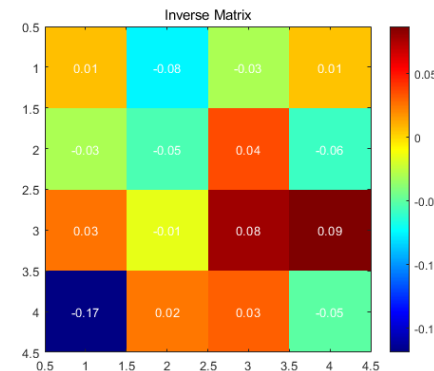
Grid matrix



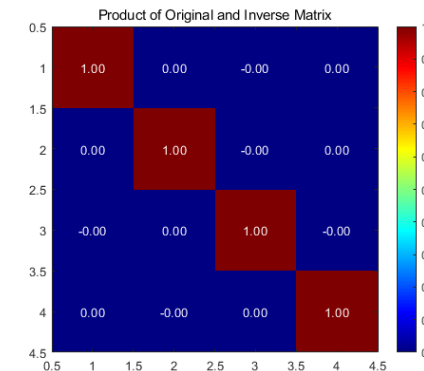
Cofactors matrix



Adjugate matrix



Inverse matrix



Original * Inverse

Implements of Various Matrices

- You can use any MATLAB of custom function.
- Just create these matrices using any method.

```
% Clear workspace, command window, and close all figures
clc; clear; close all;

% Create a 4x4 original matrix with integer elements
originalMatrix = [-5,4,-1,-7;
                 -10,-4,-2,0;
                 -8,9,6,-1;
                 7,-10,6,3];

% Create a 4x4 grid matrix where elements are (-1)^(i+j)
[i, j] = meshgrid(1:4, 1:4);
gridMatrix = (-1).^(i+j);

% Compute the minor matrix
minorMatrix = zeros(4);
for row = 1:4
    for col = 1:4
        subMatrix = originalMatrix;
        subMatrix(row, :) = [];
        subMatrix(:, col) = [];
        minorMatrix(row, col) = det(subMatrix);
    end
end

%%%%%% TODO %%%%%%%%%
% Refer to the lecture slide 43
% Compute the cofactors matrix
% Use '.*' function to compute the cofactorsMatrix
cofactorsMatrix = ;

% Check if the matrix is invertible
determinant = det(originalMatrix);
if determinant == 0
    error('The original matrix is singular and does not have an inverse.');
```

```
% Compute the adjugate matrix and inverse matrix of the original matrix
% Use 'transpose(matrix)' to compute the adjugate matrix
% Use adjugateMatrix to compute inverse matrix
adjugateMatrix = ;
inverseMatrix = ;

% Compute the product of the original matrix and its inverse
identityMatrix = ;
%%%%%%%%%%

% Visualize all matrices
visualizeMatrix(originalMatrix, 'Original Matrix');
visualizeMatrix(minorMatrix, 'Minors Matrix');
visualizeMatrix(gridMatrix, 'Grid Matrix (-1)^(i+j)');
visualizeMatrix(cofactorsMatrix, 'Cofactors Matrix');
visualizeMatrix(adjugateMatrix, 'Adjugate Matrix');
visualizeMatrix(inverseMatrix, 'Inverse Matrix');
visualizeMatrix(identityMatrix, 'Product of Original and Inverse Matrix');
```

```
% Function definition : visualize matrix
function visualizeMatrix(matrix, titleStr)
    figure;
    imagesc(matrix);
    colorbar;
    title(titleStr);
    colormap jet;
    axis square;
    % Add text annotations for each element
    for i = 1:size(matrix, 1)
        for j = 1:size(matrix, 2)
            text(j, i, num2str(matrix(i, j), '%.2f'), ...
                'HorizontalAlignment', 'center', ...
                'Color', 'white', 'FontSize', 10);
        end
    end
end
```

Sample code



**THANK YOU
FOR YOUR ATTENTION**