Lec 14: Square Linear Systems – LU Factorization

Gaussian Elimination

Simple $\begin{cases} \bullet\ \text{upper} - \Delta : \text{back. subs.} \\ \bullet\ \text{lower} - \Delta : \text{forward. elim.} \end{cases}$

General

Gaussian elimination

General Method: Gaussian Elimination

- *Gaussian elimination* is an algorithm for solving a general system of linear equations that involves a sequence of row operations performed on the associated matrix of coefficients.

  (Upper-Δ)
  echelon form

  "Simpler yet shares the same solutions."

- This is also known as the method of row reduction.
- There are three variations to this method:
  - G.E. without pivoting ③
  - G.E. with partial pivoting (that is, row pivoting) ①, ③
  - G.E. with full pivoting (that is, row and column pivoting)

    ①, ③ + column pivot.

Allowed row ops.

1. Swap : $R_i \leftrightarrow R_j$
2. Scaling : $R_i \longrightarrow cR_i$
3. lin. comb : $R_i \longrightarrow R_i + cR_j$

G.E. Without Pivoting: Example

Key Example

Solve the following system of equations.

$$\begin{cases} 2x_1 + 2x_2 + x_3 = 6 \\ -4x_1 + 6x_2 + x_3 = -8 \\ 5x_1 - 5x_2 + 3x_3 = 4 \end{cases} \xrightarrow{\text{matrix equation}} \underbrace{\begin{bmatrix} 2 & 2 & 1 \\ -4 & 6 & 1 \\ 5 & -5 & 3 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} 6 \\ -8 \\ 4 \end{bmatrix}}_{\mathbf{b}}$$

Step 1: Write the corresponding *augmented matrix* and row-reduce to an echelon form.

$$\left[\begin{array}{ccc|c} 2 & 2 & 1 & 6 \\ -4 & 6 & 1 & -8 \\ 5 & -5 & 3 & 4 \end{array}\right] \to \left[\begin{array}{ccc|c} 2 & 2 & 1 & 6 \\ \boxed{0} & 10 & 3 & 4 \\ \boxed{0} & 10 & -0.5 & 11 \end{array}\right] \to \left[\begin{array}{ccc|c} 2 & 2 & 1 & 6 \\ 0 & 10 & 3 & 4 \\ 0 & \boxed{0} & 3.5 & -7 \end{array}\right].$$

Step 2: Solve for x_3, then x_2, and then x_1 via *backward substitution*.

$$\mathbf{x} = (3, 1, -2)^{\mathrm{T}}.$$

G.E. without Pivoting: General Procedure

As shown in the example, G.E. without pivoting involves two steps:

1. **Row reduction:** Transform $A\mathbf{x} = \mathbf{b}$ to $U\mathbf{x} = \boldsymbol{\beta}$ where

$$U = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ & u_{22} & \cdots & u_{2n} \\ & & \ddots & \vdots \\ \text{0} & & & u_{nn} \end{bmatrix} \quad \text{and} \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix}.$$

2. **Backward substitution:** Solve $U\mathbf{x} = \boldsymbol{\beta}$ for $\mathbf{x}$ by

$$\begin{cases} x_n = \dfrac{\beta_n}{u_{nn}} \quad \text{and} \\ x_i = \dfrac{1}{u_{ii}} \left(\beta_i - \displaystyle\sum_{j=i+1}^{n} u_{ij} x_j \right), \quad \text{for } i = n-1, n-2, \ldots, 1 \, . \end{cases}$$

G.E. without Pivoting: MATLAB Implementation

```matlab
1  function x = GEnp(A, b)
2    % Step 1: Row reduction to upper tri. system
3    S = [A, b];               % augmented matrix
4    n = size(A, 1);
5    for j = 1:n-1
6      for i = j+1:n
7          mult = -S(i,j)/S(j,j);
8          S(i,:) = S(i,:) + mult*S(j,:);
9      end
10   end
11   % Step 2: Backward substitution
12   U = S(:,1:end-1);
13   beta = S(:,end);
14   x = backsub(U, beta);
15 end
```

$[A \mid \vec{b}]$

% i row ; j column

% $R_i \to R_i + (\#) R_j$

our code from last Friday.

Exercise. Rewrite Lines 6–9 without using a loop. (Think *vectorized*!)

Row Reduction

In j^{th} step :

$$
\begin{array}{r c}
 & \begin{array}{ccccc} & & & j^{th} \text{ col} & \\ & & & \downarrow & \end{array} \\
 & \left[\begin{array}{ccccc|c}
a_{11} & & & & \star & b_1 \\
0 & a_{22} & & & a_{j+1,1} \; - - - - & b_2 \\
0 & 0 & \ddots & 1 & & \vdots \\
\vdots & \vdots & & & & \\
j^{th} \text{ row} \rightarrow \quad 0 & 0 & \cdots & a_{jj} & & b_j \\
(j+1)^{st} \text{ row} \rightarrow \quad 0 & 0 & \cdots & a_{j+1,j} & \ddots & \vdots \\
\vdots & \vdots & & \vdots & & \\
0 & 0 & \cdots & a_{i,j} & & \vdots \\
\vdots & \vdots & & \vdots & & \\
0 & 0 & \cdots & a_{n,j} & & b_n
\end{array}\right] \\
 & \qquad\qquad A \qquad\qquad\qquad \vec{b}
\end{array}
$$

$$R_{j+1} \longrightarrow R_{j+1} + \underbrace{\left(-\frac{a_{j+1,j}}{a_{jj}}\right)}_{\text{"multiplier"}} R_j$$

$$\text{mult} = -A(\underbrace{j+1}_{i}, j) \,/\, A(j,j)$$

G.E. with Partial Pivoting: Procedure

Step 1: Row Reduction ←
Step 2: Back. subs

① swap, ③ lin. comb.

In this variation of G.E., reduction to echelon form is done slightly differently.

- On the augmented matrix $[A \mid \mathbf{b}]$,

Key Process (partial pivoting)

1. Find the entry in the first column with the largest absolute value. This entry is called the *pivot*.
2. Perform a row interchange, if necessary, so that the pivot is on the first diagonal position.
3. Use elementary row operations to reduce the remaining entries in the first column to zero.

- Once done, ignore the first row and first column and repeat the **Key Process** on the remaining submatrix.
- Continue this until the matrix is in a row-echelon form.

①
$$\left[\begin{array}{c|c} 2 & \\ 5 & \\ -7 & \end{array}\right] \text{ pivot}$$

[pivot] $\downarrow$ $R_1 \leftrightarrow R_3$

②
$$\left[\begin{array}{c|c} -7 & \\ 5 & \\ 2 & \end{array}\right]$$

[zero] $\downarrow$ elem. row ops

③
$$\left[\begin{array}{c|c} -7 & \\ 0 & \\ 0 & \end{array}\right]$$

G.E. with Partial Pivoting: Example

$R_3 \to R_3 + \frac{4}{5} R_1$

Let's solve the example on p. 4 again, now using G.E. with partial pivoting.

$1 + \frac{4}{5} 3$

1st column:

$$\left[\begin{array}{ccc|c} 2 & 2 & 1 & 6 \\ -4 & 6 & 1 & -8 \\ 5 & -5 & 3 & 4 \end{array}\right] \xrightarrow{\text{pivot}} \left[\begin{array}{ccc|c} 5 & -5 & 3 & 4 \\ -4 & 6 & 1 & -8 \\ 2 & 2 & 1 & 6 \end{array}\right] \xrightarrow{\text{zero}} \left[\begin{array}{ccc|c} 5 & -5 & 3 & 4 \\ 0 & 2 & 3.4 & -4.8 \\ 0 & 4 & -0.2 & 4.4 \end{array}\right]$$

$= 1 + \frac{12}{5}$

$= 3 + \frac{2}{5} = 3.4$

2nd column:

$$\left[\begin{array}{ccc|c} 5 & -5 & 3 & 4 \\ 0 & 2 & 3.4 & -4.8 \\ 0 & 4 & -0.2 & 4.4 \end{array}\right] \xrightarrow{\text{pivot}} \left[\begin{array}{ccc|c} 5 & -5 & 3 & 4 \\ 0 & 4 & -0.2 & 4.4 \\ 0 & 2 & 3.4 & -4.8 \end{array}\right] \xrightarrow{\text{zero}} \left[\begin{array}{ccc|c} 5 & -5 & 3 & 4 \\ 0 & 4 & -0.2 & 4.4 \\ 0 & 0 & 3.5 & -7 \end{array}\right]$$

Now that the last matrix is upper triangular, we work up from the third equation to the second to the first and obtain the same solution as before.

G.E. with Partial Pivoting: MATLAB Implementation

Exercise

Write a MATLAB function `GEpp.m` which carries out G.E. with partial pivoting.

Hints

- Modify `GEnp.m` on p. 6 to incorporate partial pivoting.
- The only part that needs to be changed is the for-loop starting at Line 5.
  - Right after `for j = 1:n-1`, find the index of the pivot element of the jth column of A below the diagonal.

```
[~, iM] = max(abs(A(j,j:end)));
iM = iM + j - 1;
```

  - If the pivot element is not on the diagonal, swap rows so that it is on the diagonal.

```
if j ~= iM
    S([j iM], :) = S([iM j], :)
end
```

Why Is Pivoting Necessary?

Example

Given $\epsilon \ll 1$, solve the system

a tiny positive #.

$$\begin{bmatrix} -\epsilon & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1-\epsilon \\ 0 \end{bmatrix}$$

using Gaussian elimination with and without partial pivoting.

Without pivoting: By $R_2 \to R_2 + (1/\epsilon)R_1$, we have

$$\begin{bmatrix} -\epsilon & 1 \\ 0 & -1+1/\epsilon \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1-\epsilon \\ 1/\epsilon - 1 \end{bmatrix} \xRightarrow[\text{subs.}]{\text{back.}} \begin{cases} x_2 = 1, \\ x_1 = \dfrac{\overbrace{(1-\epsilon) - 1}^{\text{nearly}}}{-\epsilon}. \end{cases}$$

Since ϵ is small, $1-\epsilon \approx 1$

- In exact arithmetic, this yields the correct solution. $x_1 = x_2 = 1$
- In floating-point arithmetic, calculation of x_1 suffers from catastrophic cancellation.

Why Is Pivoting Necessary? (Cont')

Upshot: Pivoting provides a stable algorithm.

(Last wed)

Example

Given $\epsilon \ll 1$, solve the system

$$\begin{bmatrix} -\epsilon & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1-\epsilon \\ 0 \end{bmatrix}$$

using Gaussian elimination with and without partial pivoting.

With partial pivoting: First, swap the rows $R_1 \leftrightarrow R_2$, and then do $R_2 \to R_2 + \epsilon R_1$ to obtain

$$\begin{bmatrix} 1 & -1 \\ 0 & 1-\epsilon \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1-\epsilon \end{bmatrix} \underset{\text{subs.}}{\overset{\text{back.}}{\Longrightarrow}} \begin{cases} x_2 = 1, \\ x_1 = \dfrac{0-(-1)}{1}. = 1. \end{cases}$$

- Each of the arithmetic steps (to compute x_1, x_2) is well-conditioned.
- The solution is computed stably.

LU Factorization

Emulation of Gaussian Elimination

In this section, we emulate row operations steps required in Gaussian elimination by matrix multiplications. **Two major operations.**

"swap rows of identity matrix"

- Row interchange $\boxed{R_i \leftrightarrow R_j:}$

$$P(i,j)A, \quad \text{where } P(i,j) \text{ is an elementary permutation matrix.}$$

- Row replacement $\boxed{R_i \to R_i + cR_j:}$

$$(I + c\mathbf{e}_i\mathbf{e}_j^{\mathrm{T}})A$$

See Appendix for more details.

Key Example Revisited

Let's work out the key example from last time once again, now in matrix form $A\mathbf{x} = \mathbf{b}$.

$$\begin{bmatrix} 2 & 2 & 1 \\ -4 & 6 & 1 \\ 5 & -5 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ -8 \\ 4 \end{bmatrix}.$$

[Pivot] Switch R_1 and R_3 using $P(1,3)$:

$$\underbrace{\begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}}_{P(1,3)} \left[\begin{bmatrix} 2 & 2 & 1 \\ -4 & 6 & 1 \\ 5 & -5 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ -8 \\ 4 \end{bmatrix} \right] \longrightarrow \begin{bmatrix} 5 & -5 & 3 \\ -4 & 6 & 1 \\ 2 & 2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 4 \\ -8 \\ 6 \end{bmatrix}$$

$P(1,3)A$

[Zero] Do row operations $R_2 \to R_2 + (4/5)R_1$ and $R_3 \to R_3 - (2/5)R_1$:

$R_1 \longrightarrow R_1$

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 4/5 & 1 & 0 \\ -2/5 & 0 & 1 \end{bmatrix}}_{G_1} \left[\begin{bmatrix} 5 & -5 & 3 \\ -4 & 6 & 1 \\ 2 & 2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 4 \\ -8 \\ 6 \end{bmatrix} \right]$$

$G_1 P(1,3) A$

$$\longrightarrow \begin{bmatrix} 5 & -5 & 3 \\ 0 & 2 & 3.4 \\ 0 & 4 & -0.2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 4 \\ -4.8 \\ 4.4 \end{bmatrix}$$

Key Example Revisited (cont')

$P(2,3)\, G_1\, P(1,3)\, A$

[Pivot] Switch R_2 and R_3 using $P(2,3)$:

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}}_{P(2,3)} \left[\begin{bmatrix} 5 & -5 & 3 \\ 0 & 2 & 3.4 \\ 0 & 4 & -0.2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 4 \\ -4.8 \\ 4.4 \end{bmatrix} \right]$$

$$\longrightarrow \begin{bmatrix} 5 & -5 & 3 \\ 0 & 4 & -0.2 \\ 0 & 2 & 3.4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 4 \\ 4.4 \\ -4.8 \end{bmatrix}$$

[Zero] Do a row operation $R_3 \to R_3 - (1/2)R_2$:

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1/2 & 1 \end{bmatrix}}_{G_2} \left[\begin{bmatrix} 5 & -5 & 3 \\ 0 & 4 & -0.2 \\ 0 & 2 & 3.4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 4 \\ 4.4 \\ -4.8 \end{bmatrix} \right]$$

$U = G_2\, P(2,3)\, G_1\, P(1,3)\, A$

$$\longrightarrow \underbrace{\begin{bmatrix} 5 & -5 & 3 \\ 0 & 4 & -0.2 \\ 0 & 0 & 3.5 \end{bmatrix}}_{U} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 4 \\ 4.4 \\ -7 \end{bmatrix}$$

Analysis of Example

- The previous calculations can be summarized as

$$G_2 P(2,3) G_1 P(1,3) A = U\,. \tag{$\star$}$$

- Using the noted properties of permutation matrices and GTMs, ($\star$) can be written as

$$G_2 P(2,3) G_1 \underbrace{P(2,3)P(2,3)}_{=I} P(1,3) A = U$$

$$\boxed{P(i,j)^2 = I}$$

$$\longrightarrow G_2 \underbrace{P(2,3)G_1P(2,3)}_{=:\widetilde{G}_1} \underbrace{P(2,3)P(1,3)}_{=:P} A = U\,. \rightarrow G_2\widetilde{G}_1 PA = U$$

$$PA = \underbrace{(G_2\widetilde{G}_1)^{-1}}_{= L} U$$

- The above can be summarized as $PA = LU$ where $L = (G_2\widetilde{G}_1)^{-1}$ is a lower triangular matrix.

G_j : Gauss. Trans. Mat. (unit lower-Δ)

Generalization – PLU Factorization

For an arbitrary matrix $A \in \mathbb{R}^{n\times n}$, the partial pivoting and row operations are intermixed as

$$G_{n-1}P(n-1, r_{n-1}) \cdots G_2 P(2, r_2) G_1 P(1, r_1) A = U\,.$$

Going through the same calculations as above, it can always be written as

$$\left(\widetilde{G}_{n-1} \cdots \widetilde{G}_2 \widetilde{G}_1\right) P(n-1, r_{n-1}) \cdots P(2, r_2) P(1, r_1) A = U\,,$$

which again leads to $PA = LU$:

$$\underbrace{P(n-1, r_{n-1}) \cdots P(2, r_2) P(1, r_1)}_{=:P} A = \underbrace{\left(\widetilde{G}_{n-1} \cdots \widetilde{G}_2 \widetilde{G}_1\right)^{-1}}_{=:L} U\,.$$

This is called the **PLU factorization** of matrix A.

LU and PLU Factorization

If no pivoting is required, the previous procedure simplifies to

$$G_{n-1} \cdots G_2 G_1 A = U \, .$$

which leads to $A = LU$:

$$A = \underbrace{\left(G_{n-1} \cdots G_2 G_1\right)^{-1}}_{=:L} U \, .$$

This is called the **LU factorization** of matrix A.

Implementation of LU Factorization

```
function [L,U] = mylu(A)
% MYLU   LU factorization (demo only--not stable!).
% Input:
%   A    square matrix
% Output:
%   L,U  unit lower triangular and upper triangular such that
    LU=A
  n = length(A);
  L = eye(n);    % ones on diagonal
  % Gaussian elimination
  for j = 1:n-1
    for i = j+1:n
      L(i,j) = A(i,j) / A(j,j);    % row multiplier
      A(i,j:n) = A(i,j:n) - L(i,j)*A(j,j:n);
    end
  end
  U = triu(A);
end
```

Implementation of LU Factorization

Exercise. Write a MATLAB function `myplu` for PLU factorization by modifying the previous function `mylu.m`.

```
function [L,U,P] = myplu(A)
% MYPLU   PLU factorization (demo only--not stable!).
% Input:
%   A    square matrix
% Output:
%   P,L,U  permutation, unit lower triangular, and upper
   triangular such that LU=PA

% Your code here.




end
```

Solving a Square System Using PLU Factorization

Multiplying $A\mathbf{x} = \mathbf{b}$ on the left by P we obtain

$$\underbrace{PA}_{=LU}\mathbf{x} = \underbrace{P\mathbf{b}}_{=:\boldsymbol{\beta}} \quad \longrightarrow \quad LU\mathbf{x} = \boldsymbol{\beta}\,,$$

which can be solved in two steps:

- Define $U\mathbf{x} = \mathbf{y}$ and solve for $\mathbf{y}$ in the equation

$$L\mathbf{y} = \boldsymbol{\beta}\,. \qquad \text{(forward elimination)}$$

- Having calculated $\mathbf{y}$, solve for $\mathbf{x}$ in the equation

$$U\mathbf{x} = \mathbf{y}\,. \qquad \text{(backward substitution)}$$

Solving a Square System Using PLU Factorization

- Using the instructional codes (`backsub`, `forelim`, `myplu`):

```
[L,U,P] = myplu(A);
x = backsub( U, forelim(L, P*b) );
```

- Using MATLAB's built-in functions:

```
[L,U,P] = lu(A);
x = U \ (L \ (P*b));
```

  - The backslash is designed so that triangular systems are solved with the appropriate substitution.

- The most compact way:

```
x = A \ b;
```

  - The backslash does partial pivoting and triangular substitutions silently and automatically.

Appendix: Row and Column Operations

Notation and Terminology

Notation: Unit Basis Vectors

Throughout this tutorial, suppose $n \in \mathbb{N}$ is fixed. Let I be the $n \times n$ identity matrix and denote by $\mathbf{e}_j$ its jth column, *i.e.*,

$$I = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} = \left[\begin{array}{c|c|c|c} & & & \\ \mathbf{e}_1 & \mathbf{e}_2 & \cdots & \mathbf{e}_n \\ & & & \end{array}\right].$$

That is,

$$\mathbf{e}_1 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad \mathbf{e}_2 = \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \quad \cdots, \quad \mathbf{e}_n = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}.$$

Notation: Concatenation

Let $A \in \mathbb{R}^{n \times n}$. We can view it as a concatenation of its rows or columns as visualized below.

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} = \left[\begin{array}{c|c|c|c} \mathbf{a}_1 & \mathbf{a}_2 & \cdots & \mathbf{a}_n \end{array}\right] = \left[\begin{array}{c} \boldsymbol{\alpha}_1^{\mathrm{T}} \\ \hline \boldsymbol{\alpha}_2^{\mathrm{T}} \\ \hline \vdots \\ \hline \boldsymbol{\alpha}_n^{\mathrm{T}} \end{array}\right].$$

Basic Row and Column Operations

Row or Column Extraction

A row or a column of A can be extracted using columns of I.

Operation	Mathematics	MATLAB
extract the ith row of A	$\mathbf{e}_i^{\mathrm{T}} A$	`A(i,:)`
extract the jth column of A	$A\mathbf{e}_j$	`A(:,j)`
extract the (i, j) entry of A	$\mathbf{e}_i^{\mathrm{T}} A\mathbf{e}_j$	`A(i,j)`

Elementary Permutation Matrices

Definition 1 (Elementary Permutation Matrix)

For $i, j \in \mathbb{N}[1, n]$ distinct, denote by $P(i, j)$ the $n \times n$ matrix obtained by interchanging the ith and jth rows of the $n \times n$ identity matrix. Such matrices are called *elementary permutation matrices*.

Example. ($n = 4$)

$$P(1,2) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad P(1,3) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \cdots$$

Notable Properties.

- $P(i, j) = P(j, i)$
- $P(i, j)^2 = I$

Row or Column Interchange

Elementary permutation matrices are useful in interchanging rows or columns.

Operation	Mathematics	MATLAB
$\boldsymbol{\alpha}_i^{\mathrm{T}} \leftrightarrow \boldsymbol{\alpha}_j^{\mathrm{T}}$	$P(i,j)A$	`A([i,j],:)=A([j,i],:)`
$\mathbf{a}_i \leftrightarrow \mathbf{a}_j$	$AP(i,j)$	`A(:,[i,j])=A(:,[j,i])`

Permutation Matrices

Definition 2 (Permutation Matrix)

A *permutation matrix* $P \in \mathbb{R}^{n \times n}$ is a square matrix obtained from the same-sized identity matrix by re-ordering of rows.

Notable Properties.

- $P^{\mathrm{T}} = P^{-1}$
- A product of *elementary permutation matrices* is a permutation matrix.

Row and Column Operations. For any $A \in \mathbb{R}^{n \times n}$,

- PA permutes the rows of A.
- AP permutes the columns of A.

Row or Column Rearrangement

Question

Let $A \in \mathbb{R}^{6\times 6}$, and suppose that it is stored in MATLAB. Rearrange rows of A by moving 1st to 2nd, 2nd to 3rd, 3rd to 5th, 4th to 6th, 5th to 4th, and 6th to 1st, that is,

$$\begin{bmatrix} \boldsymbol{\alpha}_1^{\mathrm{T}} \\ \hline \boldsymbol{\alpha}_2^{\mathrm{T}} \\ \hline \boldsymbol{\alpha}_3^{\mathrm{T}} \\ \hline \boldsymbol{\alpha}_4^{\mathrm{T}} \\ \hline \boldsymbol{\alpha}_5^{\mathrm{T}} \\ \hline \boldsymbol{\alpha}_6^{\mathrm{T}} \end{bmatrix} \longrightarrow \begin{bmatrix} \boldsymbol{\alpha}_6^{\mathrm{T}} \\ \hline \boldsymbol{\alpha}_1^{\mathrm{T}} \\ \hline \boldsymbol{\alpha}_2^{\mathrm{T}} \\ \hline \boldsymbol{\alpha}_5^{\mathrm{T}} \\ \hline \boldsymbol{\alpha}_3^{\mathrm{T}} \\ \hline \boldsymbol{\alpha}_4^{\mathrm{T}} \end{bmatrix}$$

Row or Column Rearrangement

Solution.

- Mathematically: PA where

$$P = \left[\begin{array}{c} \mathbf{e}_6^{\mathrm{T}} \\ \hline \mathbf{e}_1^{\mathrm{T}} \\ \hline \mathbf{e}_2^{\mathrm{T}} \\ \hline \mathbf{e}_5^{\mathrm{T}} \\ \hline \mathbf{e}_3^{\mathrm{T}} \\ \hline \mathbf{e}_4^{\mathrm{T}} \end{array}\right] = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}.$$

- MATLAB:

```
A = A([6 1 2 5 3 4], :)
% short for  A([1 2 3 4 5 6], :) = A([6 1 2 5 3 4], :)
```

Gaussian Transformation Matrices

Elementary Row Operation and GTM

Let $1 \leqslant j < i \leqslant n$.

- The row operation $R_i \to R_i + cR_j$ on $A \in \mathbb{R}^{n\times n}$, for some $c \in \mathbb{R}$, can be emulated by a matrix multiplication[1]

$$(I + c\,\mathbf{e}_i\mathbf{e}_j^{\mathrm{T}})A.$$

- In the context of Gaussian elimination, the operation of introducing zeros below the jth diagonal entry can be done via

$$\underbrace{\Big(I + \sum_{i=j+1}^{n} c_{i,j}\,\mathbf{e}_i\mathbf{e}_j^{\mathrm{T}}\Big)}_{=G_j} A, \quad 1 \leqslant j < n.$$

The matrix G_j is called a *Gaussian transformation matrix* (GTM).

[1] Many linear algebra texts refer to the matrix in parentheses as an *elementary matrix*.

Elementary Row Operation and GTM (cont')

- To emulate $(I + c\mathbf{e}_i\mathbf{e}_j^{\mathrm{T}})A$ in MATLAB:

```
A(i,:) = A(i,:) + c*A(j,:);
```

- To emulate

$$G_jA = (I + \sum_{i=j+1}^{n} c_{i,j}\mathbf{e}_i\mathbf{e}_j^{\mathrm{T}})A$$

  in MATLAB:

```
for i = j+1:n
    c = ....
    A(i,:) = A(i,:) + c*A(j,:);
end
```

  This can be done without using a loop.

Analytical Properties of GTM

- GTMs are *unit* lower triangular matrices.
- The product of GTMs is another unit lower triangular matrix.
- The inverse of a GTM is also a unit lower triangular matrix.