SOLVING LINEAR SYSTEMS OF EQUATIONS

- **Background on linear systems**
- **Gaussian elimination and the Gauss-Jordan algorithms**
- **The LU factorization**
- **Gaussian Elimination with pivoting – permutation matrices.**
- **Case of banded systems**

Background: Linear systems

The Problem: A is an $n \times n$ matrix, and b a vector of $\mathbb{R}^n$. Find x such that:

$$Ax = b$$

➤ x is the unknown vector, b the right-hand side, and A is the coefficient matrix

Example:

$$\begin{cases} 2x_1 + 4x_2 + 4x_3 = 6 \\ x_1 + 5x_2 + 6x_3 = 4 \\ x_1 + 3x_2 + x_3 = 8 \end{cases} \quad \text{or} \quad \begin{pmatrix} 2 & 4 & 4 \\ 1 & 5 & 6 \\ 1 & 3 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 6 \\ 4 \\ 8 \end{pmatrix}$$

✍1 Solution of above system ?

➤ Standard mathematical solution by Cramer's rule:

$$x_i = \det(A_i)/\det(A)$$

A_i = matrix obtained by replacing i-th column by b.

➤ Note: This formula is useless in practice beyond $n = 3$ or $n = 4$.

Three situations:

1. The matrix A is nonsingular. There is a unique solution given by $x = A^{-1}b$.
2. The matrix A is singular and $b \in \mathrm{Ran}(A)$. There are infinitely many solutions.
3. The matrix A is singular and $b \notin \mathrm{Ran}(A)$. There are no solutions.

Example: (1) Let $A = \begin{pmatrix} 2 & 0 \\ 0 & 4 \end{pmatrix}$ $b = \begin{pmatrix} 1 \\ 8 \end{pmatrix}$. A is nonsingular ➤ a unique solution $x = \begin{pmatrix} 0.5 \\ 2 \end{pmatrix}$.

Example: (2) Case where A is singular & $b \in \mathrm{Ran}(A)$:

$$A = \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 0 \end{pmatrix}.$$

➤ infinitely many solutions: $x(\alpha) = \begin{pmatrix} 0.5 \\ \alpha \end{pmatrix}$ $\forall \alpha$.

Example: (3) Let A same as above, but $b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$.

➤ No solutions since 2nd equation cannot be satisfied

Triangular linear systems

Example:

$$\begin{pmatrix} 2 & 4 & 4 \\ 0 & 5 & -2 \\ 0 & 0 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ 4 \end{pmatrix}$$

- One equation can be trivially solved: the last one. $x_3 = 2$
- x_3 is known we can now solve the 2nd equation:

$$5x_2 - 2x_3 = 1 \rightarrow 5x_2 - 2 \times 2 = 1 \rightarrow x_2 = 1$$

- Finally x_1 can be determined similarly:

$$2x_1 + 4x_2 + 4x_3 = 2 \rightarrow \ldots \rightarrow x_1 = -5$$

ALGORITHM : 1. *Back-Substitution algorithm*

For $i = n : -1 : 1$ *do:*
  $t := b_i$
  For $j = i + 1 : n$ *do*
    $t := t - a_{ij} x_j$
  End
  $x_i = t / a_{ii}$
End

$$t := b_i - (a_{i,i+1:n}, x_{i+1:n}) = b_i - \text{an inner product}$$

- We must require that each $a_{ii} \neq 0$
- Operation count?

Column version of back-substitution

Back-Substitution algorithm. Column version

For $j = n : -1 : 1$ do:
  $x_j = b_j / a_{jj}$
  For $i = 1 : j - 1$ do
    $b_i := b_i - x_j * a_{ij}$
  End
End

✍2 Justify the above algorithm [Show that it does indeed compute the solution]

- Analogous algorithms for *lower* triangular systems.

Linear Systems of Equations: Gaussian Elimination

- Back to arbitrary linear systems.

Principle of the method: Since triangular systems are easy to solve, we will transform a linear system into one that is triangular. Main operation: combine rows so that zeros appear in the required locations to make the system triangular.

Notation: use a Tableau:

$$\begin{cases} 2x_1 + 4x_2 + 4x_3 = 2 \\ x_1 + 3x_2 + 1x_3 = 1 \\ x_1 + 5x_2 + 6x_3 = -6 \end{cases} \quad \text{tableau: } \left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 1 & 3 & 1 & 1 \\ 1 & 5 & 6 & -6 \end{array}\right]$$

- Main operation used: scaling and adding rows.

Example: Replace row2 by: row2 - $\frac{1}{2}$*row1:

$$\left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 1 & 3 & 1 & 1 \\ 1 & 5 & 6 & -6 \end{array}\right] \rightarrow \left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 1 & 5 & 6 & -6 \end{array}\right]$$

- This is equivalent to:

$$\left[\begin{array}{ccc} 1 & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ 0 & 0 & 1 \end{array}\right] \times \left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 1 & 3 & 1 & 1 \\ 1 & 5 & 6 & -6 \end{array}\right] = \left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 1 & 5 & 6 & -6 \end{array}\right]$$

- The left-hand matrix is of the form $M = I - ve_1^T$ with $v = \begin{pmatrix} 0 \\ \frac{1}{2} \\ 0 \end{pmatrix}$

Linear Systems of Equations: Gaussian Elimination

Go back to original system. Step 1 must transform:

$$\left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 1 & 3 & 1 & 1 \\ 1 & 5 & 6 & -6 \end{array}\right] \text{ into: } \left[\begin{array}{ccc|c} x & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \end{array}\right]$$

$row_2 := row_2 - \frac{1}{2} \times row_1$: $\quad row_3 := row_3 - \frac{1}{2} \times row_1$:

$$\left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 1 & 5 & 6 & -6 \end{array}\right] \qquad \left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 3 & 4 & -7 \end{array}\right]$$

- Equivalent to

$$\left[\begin{array}{ccc} 1 & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ -\frac{1}{2} & 0 & 1 \end{array}\right] \times \left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 1 & 3 & 1 & 1 \\ 1 & 5 & 6 & -6 \end{array}\right] = \left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 3 & 4 & -7 \end{array}\right]$$

$$[A, b] \rightarrow [M_1 A, M_1 b]; \quad M_1 = I - v^{(1)} e_1^T; \quad v^{(1)} = \begin{pmatrix} 0 \\ \frac{1}{2} \\ \frac{1}{2} \end{pmatrix}$$

- New system $A_1 x = b_1$. Step 2 must now transform:

$$\left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 3 & 4 & -7 \end{array}\right] \text{ into: } \left[\begin{array}{ccc|c} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \end{array}\right]$$

$$row_3 := row_3 - 3 \times row_2 : \rightarrow \left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 7 & -7 \end{array}\right]$$

- Equivalent to

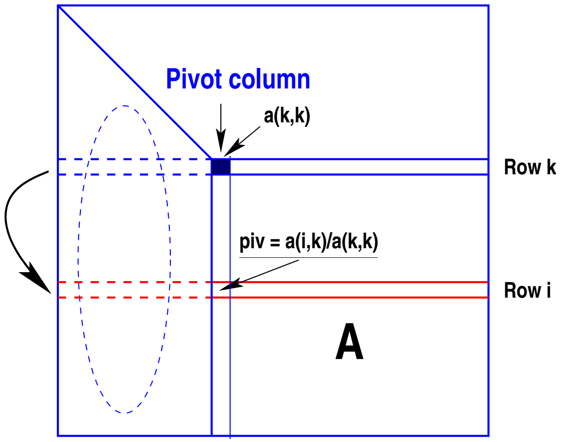
$$\left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -3 & 1 \end{array}\right] \times \left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 3 & 4 & -7 \end{array}\right] = \left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 7 & -7 \end{array}\right]$$

- Second transformation is as follows:

$$[A_1, b_1] \rightarrow [M_2 A_1, M_2 b_1]; \; M_2 = I - v^{(2)} e_2^T; \; v^{(2)} = \begin{pmatrix} 0 \\ 0 \\ 3 \end{pmatrix}$$

- Triangular system ➤ Solve.

Gaussian Elimination in a picture




```
For i=k+1:n Do:
  piv = a(i,k)/a(k,k)
  row(i):=row(i) – piv*row(k)
```


ALGORITHM : 2. *Gaussian Elimination*

1. *For* $k = 1 : n-1$ *Do:*
2. *For* $i = k+1 : n$ *Do:*
3. $piv := a_{ik}/a_{kk}$
4. *For* $j := k+1 : n+1$ *Do :*
5. $a_{ij} := a_{ij} - piv * a_{kj}$
6. *End*
6. *End*
7. *End*

➤ Operation count:

$$T = \sum_{k=1}^{n-1} \sum_{i=k+1}^{n} [1 + \sum_{j=k+1}^{n+1} 2] = \sum_{k=1}^{n-1} \sum_{i=k+1}^{n} (2(n-k)+3) = \ldots$$

✍3 Complete the above calculation. Order of the cost?

The LU factorization

➤ Now ignore the right-hand side from the transformations.

Observation: Gaussian elimination is equivalent to $n-1$ successive Gaussian transformations, i.e., multiplications with matrices of the form $M_k = I - v^{(k)} e_k^T$, where the first k components of $v^{(k)}$ equal zero.

➤ Set $A_0 \equiv A$

$$A \rightarrow M_1 A_0 = A_1 \rightarrow M_2 A_1 = A_2 \rightarrow M_3 A_2 = A_3 \cdots$$
$$\rightarrow M_{n-1} A_{n-2} = A_{n-1} \equiv U$$

➤ Last $A_k \equiv U$ is an upper triangular matrix.

➤ At each step we have: $A_k = M_{k+1}^{-1} A_{k+1}$. Therefore:

$$\begin{aligned} A_0 &= M_1^{-1} A_1 \\ &= M_1^{-1} M_2^{-1} A_2 \\ &= M_1^{-1} M_2^{-1} M_3^{-1} A_3 \\ &= \ldots \\ &= M_1^{-1} M_2^{-1} M_3^{-1} \cdots M_{n-1}^{-1} A_{n-1} \end{aligned}$$

➤ $$L = M_1^{-1} M_2^{-1} M_3^{-1} \cdots M_{n-1}^{-1}$$

➤ Note: L is Lower triangular, A_{n-1} is upper triangular

➤ LU decomposition : $A = LU$

How to get L?

$$L = M_1^{-1} M_2^{-1} M_3^{-1} \cdots M_{n-1}^{-1}$$

➤ Consider only the first 2 matrices in this product.

➤ Note $M_k^{-1} = (I - v^{(k)} e_k^T)^{-1} = (I + v^{(k)} e_k^T)$. So:

$M_1^{-1} M_2^{-1} = (I + v^{(1)} e_1^T)(I + v^{(2)} e_2^T) = I + v^{(1)} e_1^T + v^{(2)} e_2^T.$

➤ Generally, $\boxed{M_1^{-1} M_2^{-1} \cdots M_k^{-1} = I + v^{(1)} e_1^T + v^{(2)} e_2^T + \cdots v^{(k)} e_k^T}$

The L factor is a lower triangular matrix with ones on the diagonal. Column k of L, contains the multipliers l_{ik} used in the k-th step of Gaussian elimination.

➤ There is an 'algorithmic' approach to understanding the LU factorization [see supplemental notes]

A matrix A has an LU decomposition if

$$\det(A(1:k, 1:k)) \neq 0 \quad \text{for} \quad k = 1, \cdots, n-1.$$

In this case, the determinant of A satisfies:

$$\det A = \det(U) = \prod_{i=1}^{n} u_{ii}$$

If, in addition, A is nonsingular, then the LU factorization is unique.

✍4 Practical use: Show how to use the LU factorization to solve linear systems with the same matrix A and different b's.

✍5 LU factorization of the matrix $A = \begin{pmatrix} 2 & 4 & 4 \\ 1 & 5 & 6 \\ 1 & 3 & 1 \end{pmatrix}$?

✍6 Determinant of A?

✍7 True or false: "Computing the LU factorization of matrix A involves more arithmetic operations than solving a linear system $Ax = b$ by Gaussian elimination".

Gauss-Jordan Elimination

Principle of the method: We will now transform the system into one that is even easier to solve than triangular systems, namely a diagonal system. The method is very similar to Gaussian Elimination. It is just a bit more expensive.

Back to original system. Step 1 must transform:

$$\left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 1 & 3 & 1 & 1 \\ 1 & 5 & 6 & -6 \end{array}\right] \text{ into: } \left[\begin{array}{ccc|c} x & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \end{array}\right]$$

$row_2 := row_2 - 0.5 \times row_1$: $row_3 := row_3 - 0.5 \times row_1$:

$$\left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 1 & 5 & 6 & -6 \end{array}\right] \qquad \left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 3 & 4 & -7 \end{array}\right]$$

Step 2: $\left[\begin{array}{ccc|c} 2 & 4 & 4 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 3 & 4 & -7 \end{array}\right]$ into: $\left[\begin{array}{ccc|c} x & 0 & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \end{array}\right]$

$row_1 := row_1 - 4 \times row_2$: $row_3 := row_3 - 3 \times row_2$:

$$\left[\begin{array}{ccc|c} 2 & 0 & 8 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 3 & 4 & -7 \end{array}\right] \qquad \left[\begin{array}{ccc|c} 2 & 0 & 8 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 7 & -7 \end{array}\right]$$

There is now a third step:

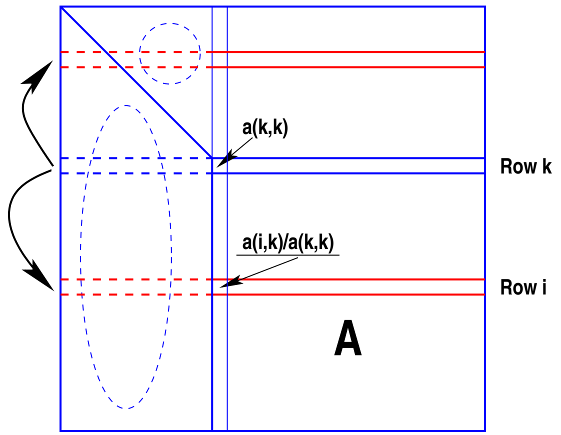
To transform: $\left[\begin{array}{ccc|c} 2 & 0 & 8 & 2 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 7 & -7 \end{array}\right]$ into: $\left[\begin{array}{ccc|c} x & 0 & 0 & x \\ 0 & x & 0 & x \\ 0 & 0 & x & x \end{array}\right]$

$row_1 := row_1 - \frac{8}{7} \times row_3$: $row_2 := row_2 - \frac{-1}{7} \times row_3$:

$$\left[\begin{array}{ccc|c} 2 & 0 & 0 & 10 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 7 & -7 \end{array}\right] \qquad \left[\begin{array}{ccc|c} 2 & 0 & 0 & 10 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 7 & -7 \end{array}\right]$$

Solution: $x_3 = -1;\ x_2 = -1;\ x_1 = 5$

Gauss-Jordan Elimination in a picture



ALGORITHM : 3. *Gauss-Jordan elimination*

1. *For* $k = 1 : n$ *Do:*
2. *For* $i = 1 : n$ *and if* $i! = k$ *Do :*
3. $piv := a_{ik}/a_{kk}$
4. *For* $j := k + 1 : n + 1$ *Do :*
5. $a_{ij} := a_{ij} - piv * a_{kj}$
6. *End*
6. *End*
7. *End*

➤ Operation count:

$$T = \sum_{k=1}^{n} \sum_{i=1}^{n-1} [1 + \sum_{j=k+1}^{n+1} 2] = \sum_{k=1}^{n} \sum_{i=1}^{n-1} (2(n-k)+3) = \cdots$$

✍8 Complete the above calculation. Order of the cost? How does it compare with Gaussian Elimination?

```
 function x = gaussj (A, b)
%--------------------------------------------------
%  function x = gaussj (A, b)
%  solves A x  = b by Gauss-Jordan elimination
%--------------------------------------------------
 n = size(A,1) ;
 A = [A,b];
 for k=1:n
   for i=1:n
     if (i ~= k)
         piv = A(i,k) / A(k,k) ;
         A(i,k+1:n+1) = A(i,k+1:n+1) - piv*A(k,k+1:n+1);
      end
   end
 end
 x = A(:,n+1) ./ diag(A) ;
```


Gaussian Elimination: Partial Pivoting

Consider again GE for the system:

$$\begin{cases} 2x_1 + 2x_2 + 4x_3 = 2 \\ x_1 + x_2 + x_3 = 1 \\ x_1 + 4x_2 + 6x_3 = -5 \end{cases} \quad \text{Or: } \left[\begin{array}{ccc|c} 2 & 2 & 4 & 2 \\ 1 & 1 & 1 & 1 \\ 1 & 4 & 6 & -5 \end{array}\right]$$

➤ $row_2 := row_2 - \frac{1}{2} \times row_1$:

$$\left[\begin{array}{ccc|c} 2 & 2 & 4 & 2 \\ 0 & 0 & -1 & 0 \\ 1 & 4 & 6 & -5 \end{array}\right]$$

➤ $row_3 := row_3 - \frac{1}{2} \times row_1$:

$$\left[\begin{array}{ccc|c} 2 & 2 & 4 & 2 \\ 0 & 0 & -1 & 0 \\ 0 & 3 & 4 & -6 \end{array}\right]$$

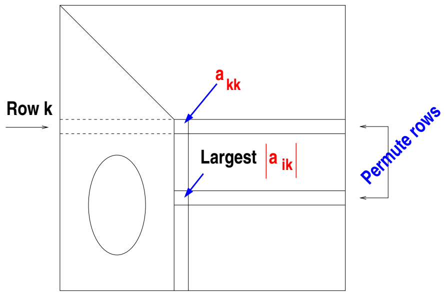
➤ Pivot a_{22} is zero. Solution : permute rows 2 and 3:

$$\left[\begin{array}{ccc|c} 2 & 2 & 4 & 2 \\ 0 & 3 & 4 & -6 \\ 0 & 0 & -1 & 0 \end{array}\right]$$

Gaussian Elimination with Partial Pivoting

Partial Pivoting

➤ General situation:



Always permute row k with row l such that $|a_{lk}| = \max_{i=k,\ldots,n} |a_{ik}|$

➤ More ‘stable’ algorithm.

✍9 The matlab script *gaussp* will be provided. Explore it from the angle of an actual implementation in a language like C. Is it necessary to ‘physically’ move the rows? (moving data around is not free).

Pivoting and permutation matrices

➤ A permutation matrix is a matrix obtained from the identity matrix by permuting its rows

➤ For example for the permutation $\pi = \{3, 1, 4, 2\}$ we obtain

$$P = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

➤ Important observation: the matrix PA is obtained from A by permuting its rows with the permutation π

$$(PA)_{i,:} = A_{\pi(i),:}$$

✍10 What is the matrix PA when

$$P = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix} \quad A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 0 & -1 & 2 \\ -3 & 4 & -5 & 6 \end{pmatrix} ?$$

➤ Any permutation matrix is the product of interchange permutations, which only swap two rows of I.

➤ Notation: E_{ij} = Identity with rows i and j swapped

Example: To obtain $\pi = \{3, 1, 4, 2\}$ from $\pi = \{1, 2, 3, 4\}$ – we need to swap $\pi(2) \leftrightarrow \pi(3)$ then $\pi(3) \leftrightarrow \pi(4)$ and finally $\pi(1) \leftrightarrow \pi(2)$. Hence:

$$P = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix} = E_{1,2} \times E_{3,4} \times E_{2,3}$$

✍11 In the previous example where

```
>> A = [ 1 2 3 4; 5 6 7 8; 9  0 -1 2 ; -3 4 -5 6]
```

Matlab gives $det(A) = -896$. What is $det(PA)$?

➤ At each step of G.E. with partial pivoting:

$$M_{k+1}E_{k+1}A_k = A_{k+1}$$

where E_{k+1} encodes a swap of row $k+1$ with row $l > k+1$.

➤ Notes: (1) $E_i^{-1} = E_i$ and (2) $M_j^{-1} \times E_{k+1} = E_{k+1} \times \tilde{M}_j^{-1}$ for $k \geq j$, where $\tilde{M}_j$ has a permuted Gauss vector:

$$\begin{aligned} (I + v^{(j)}e_j^T)E_{k+1} &= E_{k+1}(I + E_{k+1}v^{(j)}e_j^T) \\ &\equiv E_{k+1}(I + \tilde{v}^{(j)}e_j^T) \\ &\equiv E_{k+1}\tilde{M}_j \end{aligned}$$

➤ Here we have used the fact that above row $k+1$, the permutation matrix E_{k+1} looks just like an identity matrix.

Result:

$$
\begin{aligned}
A_0 &= E_1 M_1^{-1} A_1 \\
&= E_1 M_1^{-1} E_2 M_2^{-1} A_2 = E_1 E_2 \tilde{M}_1^{-1} M_2^{-1} A_2 \\
&= E_1 E_2 \tilde{M}_1^{-1} M_2^{-1} E_3 M_3^{-1} A_3 \\
&= E_1 E_2 E_3 \tilde{M}_1^{-1} \tilde{M}_2^{-1} M_3^{-1} A_3 \\
&= \ldots \\
&= E_1 \cdots E_{n-1} \times \tilde{M}_1^{-1} \tilde{M}_2^{-1} \tilde{M}_3^{-1} \cdots \tilde{M}_{n-1}^{-1} \times A_{n-1}
\end{aligned}
$$

➤ In the end

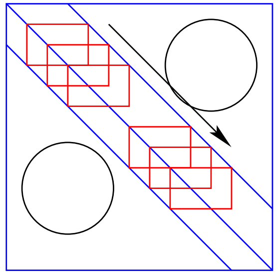
$PA = LU$ with $P = E_{n-1} \cdots E_1$

Special case of banded matrices

➤ Banded matrices arise in many applications

➤ A has upper bandwidth q if $a_{ij} = 0$ *for* $j - i > q$

➤ A has lower bandwidth p if $a_{ij} = 0$ *for* $i - j > p$



✍12 Explain how GE would work on a banded system (you want to avoid operations involving zeros) –
Hint: see picture

➤ Simplest case: tridiagonal ➤ $p = q = 1$.

➤ First observation: Gaussian elimination (no pivoting) preserves the initial banded form. Consider first step of Gaussian elimination:

```
For i = 2 : n Do:
    a_i1 := a_i1 / a_11 (pivots)
    For j := 2 : n Do :
      a_ij := a_ij − a_i1 ∗ a_1j
    End
End
```

➤ If A has upper bandwidth q and lower bandwidth p then so is the resulting $[L/U]$ matrix. ➤ Band form is preserved (induction)

✍13 Operation count?

What happens when partial pivoting is used?

If A has lower bandwidth p, upper bandwidth q, and if Gaussian elimination with partial pivoting is used, then the resulting U has upper bandwidth $p + q$. L has at most $p + 1$ nonzero elements per column (bandedness is lost).

➤ Simplest case: tridiagonal ➤ $p = q = 1$.

Example:

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