

# Parallel Scientific Computing

Course AMS301 — Fall 2022 — Lecture 5

Direct methods for linear systems

# Solution procedures for linear systems — Recap

Find  $\mathbf{x} \in \mathbb{R}^n$  such that  $\boxed{\mathbf{Ax} = \mathbf{b}}$  with  $\mathbf{A} \in \mathbb{R}^{n \times n}$  and  $\mathbf{b} \in \mathbb{R}^n$ .

## Solution procedures

- ▶ **Direct methods:** Factorization of  $\mathbf{A}$  into triangular and diagonal matrices (ex.  $\mathbf{A} = \mathbf{LU}$ ) and solution of simpler problems.

$$\mathbf{Ax} = \mathbf{b} \quad \Leftrightarrow \quad \mathbf{LUx} = \mathbf{b} \quad \Leftrightarrow \quad \begin{cases} \mathbf{Ly} = \mathbf{b} \\ \mathbf{Ux} = \mathbf{y} \end{cases}$$

Advantages: exact solution known after a given number of operations

Difficulties: heavy computational cost (*operations/memory*), hard to parallelize

- ▶ **Iterative methods:** Iterative procedure to minimizing an error  $\|\mathbf{x}^{(\ell)} - \mathbf{x}_{\text{ref}}\|$  and/or a residual  $\|\mathbf{Ax}^{(\ell)} - \mathbf{b}\|$ .

$$\begin{cases} \mathbf{x}^{(0)} = \text{Iter}_{(0)}(\mathbf{A}, \mathbf{b}) \\ \mathbf{x}^{(\ell+1)} = \text{Iter}^{(\ell+1)}(\mathbf{x}^{(\ell)}, \mathbf{x}^{(\ell-1)}, \dots, \mathbf{A}, \mathbf{b}), \quad \text{pour } \ell \geq 0 \end{cases}$$

Advantages: limited cost per iteration (*operations/memory*), easy to parallelize

Difficulties: approximate solution, control of the convergence of the process

## Direct methods for linear systems

*Solution of triangular systems*

*Gaussian elimination*

*Matrix factorization*

## Definitions

- ▶ Lower triangular matrix:  $\mathbf{L} \in \mathbb{R}^{n \times n}$  with  $L_{ij} = 0$  if  $i < j$
- ▶ Upper triangular matrix:  $\mathbf{U} \in \mathbb{R}^{n \times n}$  with  $U_{ij} = 0$  if  $i > j$

$$\mathbf{L} = \begin{bmatrix} L_{11} & 0 & 0 \\ L_{21} & L_{22} & 0 \\ L_{31} & L_{32} & L_{33} \end{bmatrix} \quad \mathbf{U} = \begin{bmatrix} U_{11} & U_{12} & U_{13} \\ 0 & U_{22} & U_{23} \\ 0 & 0 & U_{33} \end{bmatrix}$$

## Properties

- ▶ The determinant of a triangular matrix is the product of the diagonal elements:

$$\det(\mathbf{L}) = \prod_{i=1}^n L_{ii} \quad \det(\mathbf{U}) = \prod_{i=1}^n U_{ii}$$

- ▶ A triangular matrix is invertible if, and only if, its diagonal elements are nonzero.

## Triangular systems — Approaches by points [2/2]

### Procedure to solve $\mathbf{Lx} = \mathbf{b}$ (*Forward substitution*)

Data:  $\mathbf{L}$  et  $\mathbf{b}$

Initialization:  $\mathbf{x} \leftarrow 0$

**for**  $i = 1, \dots, n$  **do**

$x_i \leftarrow [b_i - \sum_{j=1}^{i-1} L_{ij}x_j] / L_{ii}$

**end**

### Procedure to solve $\mathbf{Ux} = \mathbf{b}$ (*Back substitution*)

Data:  $\mathbf{U}$  et  $\mathbf{b}$

Initialization:  $\mathbf{x} \leftarrow 0$

**for**  $i = n, \dots, 1$  **do**

$x_i \leftarrow [b_i - \sum_{j=i+1}^n U_{ij}x_j] / U_{ii}$

**end**

### Algorithmic aspects

- ▶ Cost:  $n^2$  operations
- ▶ Weak parallelization (*computation of sums in parallel*)

## Definitions

- ▶ Lower triangular matrix by blocks:  $\mathbf{L} \in \mathbb{R}^{n \times n}$  with  $\mathbf{L}_{IJ} = 0$  si  $I < J$
- ▶ Upper triangular matrix by blocks:  $\mathbf{U} \in \mathbb{R}^{n \times n}$  with  $\mathbf{U}_{IJ} = 0$  si  $I > J$

$$\mathbf{L} = \begin{bmatrix} \mathbf{L}_{11} & 0 & 0 \\ \mathbf{L}_{21} & \mathbf{L}_{22} & 0 \\ \mathbf{L}_{31} & \mathbf{L}_{32} & \mathbf{L}_{33} \end{bmatrix} \quad \mathbf{U} = \begin{bmatrix} \mathbf{U}_{11} & \mathbf{U}_{12} & \mathbf{U}_{13} \\ 0 & \mathbf{U}_{22} & \mathbf{U}_{23} \\ 0 & 0 & \mathbf{U}_{33} \end{bmatrix}$$

The diagonal blocks are square. The other blocks may not be square.

## Properties

- ▶ The determinant of a triangular matrix by blocks is the product of the determinants of the diagonal blocks:

$$\det(\mathbf{L}) = \prod_{I=1}^N \det(\mathbf{L}_{II}) \quad \det(\mathbf{U}) = \prod_{I=1}^N \det(\mathbf{U}_{II})$$

- ▶ A diagonal or triangular matrix by blocks is invertible if, and only if, the diagonal blocks are invertibles.

### Procedure to solve $\mathbf{Lx} = \mathbf{b}$ (*Forward substitution*)

Data:  $\mathbf{L}$  et  $\mathbf{b}$

Initialization:  $\mathbf{x} \leftarrow 0$

**for**  $I = 1, \dots, N$  **do**

$$\left| \mathbf{x}_I \leftarrow \mathbf{L}_{II}^{-1} \left[ \mathbf{b}_I - \sum_{J=1}^{I-1} \mathbf{L}_{IJ} \mathbf{x}_J \right] \right.$$

**end**

### Procedure to solve $\mathbf{Ux} = \mathbf{b}$ (*Back substitution*)

Data:  $\mathbf{U}$  et  $\mathbf{b}$

Initialization:  $\mathbf{x} \leftarrow 0$

**for**  $I = N, \dots, 1$  **do**

$$\left| \mathbf{x}_I \leftarrow \mathbf{U}_{II}^{-1} \left[ \mathbf{b}_I - \sum_{J=I+1}^N \mathbf{U}_{IJ} \mathbf{x}_J \right] \right.$$

**end**

### Algorithmic aspects

- ▶ Cost:  $N$  small system to solve and  $N(N-1)/2$  matrix-vector products
- ▶ Better for parallel computing (*computation of matrix-vector products in parallel*)

## Direct methods for linear systems

*Solution of triangular systems*

*Gaussian elimination*

*Matrix factorization*



# General systems — Gaussian elimination [1/5]

**Goal:** Transform  $\mathbf{Ax} = \mathbf{b}$  into an equivalent system  $\mathbf{Ux} = \mathbf{b}'$ .

## Procedure

- Initialization:

$$\begin{bmatrix} A_{11}^{(1)} & A_{12}^{(1)} & \cdots & A_{1n}^{(1)} \\ A_{21}^{(1)} & A_{22}^{(1)} & \cdots & A_{2n}^{(1)} \\ \vdots & \vdots & & \vdots \\ A_{n1}^{(1)} & A_{n2}^{(1)} & \cdots & A_{nn}^{(1)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1^{(1)} \\ b_2^{(1)} \\ \vdots \\ b_n^{(1)} \end{bmatrix}$$

- Iteration 1:

$$\begin{bmatrix} A_{11}^{(1)} & A_{12}^{(1)} & \cdots & A_{1n}^{(1)} \\ 0 & A_{22}^{(2)} & \cdots & A_{2n}^{(2)} \\ \vdots & \vdots & & \vdots \\ 0 & A_{n2}^{(2)} & \cdots & A_{nn}^{(2)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_n^{(2)} \end{bmatrix}$$

with

$$\begin{aligned} \alpha_{i1} &= A_{i1}^{(1)} / A_{11}^{(1)} & i &= 2, \dots, n & \text{(multipliers)} \\ A_{ij}^{(2)} &= A_{ij}^{(1)} - \alpha_{i1} A_{1j}^{(1)} & i, j &= 2, \dots, n \\ b_i^{(2)} &= b_i^{(1)} - \alpha_{i1} b_1^{(1)} & i &= 2, \dots, n \end{aligned}$$

# General systems — Gaussian elimination [2/5]

**Goal:** Transform  $\mathbf{Ax} = \mathbf{b}$  into an equivalent system  $\mathbf{Ux} = \mathbf{b}'$ .

## Procedure

► Iteration  $k$ :

$$\begin{bmatrix} A_{11}^{(1)} & A_{12}^{(1)} & \cdots & \cdots & \cdots & A_{1n}^{(1)} \\ 0 & A_{22}^{(2)} & & & & A_{2n}^{(2)} \\ \vdots & & \ddots & & & \vdots \\ 0 & \cdots & 0 & A_{k+1,k+1}^{(k)} & \cdots & A_{k+1,n}^{(k)} \\ \vdots & & \vdots & \vdots & & \vdots \\ 0 & \cdots & 0 & A_{n,k+1}^{(k)} & \cdots & A_{n,n}^{(k)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{k+1} \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_{k+1}^{(k)} \\ \vdots \\ b_n^{(k)} \end{bmatrix}$$

► Iteration  $n - 1$ :

$$\begin{bmatrix} A_{11}^{(1)} & A_{12}^{(1)} & \cdots & \cdots & A_{1n}^{(1)} \\ 0 & A_{22}^{(2)} & & & A_{2n}^{(2)} \\ \vdots & & \ddots & & \vdots \\ 0 & & & \ddots & \vdots \\ 0 & & & & A_{nn}^{(n)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ \vdots \\ b_n^{(n)} \end{bmatrix}$$

## Gaussian elimination

Data:  $\mathbf{A}$  et  $\mathbf{b}$

Initialization:  $\mathbf{A}^{(1)} = \mathbf{A}$  et  $\mathbf{b}^{(1)} = \mathbf{b}$

**for**  $k = 1, \dots, n - 1$  **do**

**for**  $i = k + 1, \dots, n$  **do**

$$\alpha_{ik} = A_{ik}^{(k)} / A_{kk}^{(k)}$$

$$A_{ij}^{(k+1)} = A_{ij}^{(k)} - \alpha_{ik} A_{kj}^{(k)} \quad (j = k + 1, \dots, n)$$

$$b_i^{(k+1)} = b_i^{(k)} - \alpha_{ik} b_k^{(k)}$$

**end**

**end**

Si  $A_{kk}^{(k)} \neq 0!$

## Gaussian elimination (*Rewriting*)

Data:  $\mathbf{A}$  et  $\mathbf{b}$

**for**  $k = 1, \dots, n - 1$  **do**

**for**  $i = k + 1, \dots, n$  **do**

$$\alpha \leftarrow A_{ik} / A_{kk}$$

$$A_{ij} \leftarrow A_{ij} - \alpha A_{kj} \quad (j = k + 1, \dots, n)$$

$$A_{ik} \leftarrow 0$$

$$b_i \leftarrow b_i - \alpha b_k$$

**end**

**end**

Si  $A_{kk} \neq 0!$

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \Leftrightarrow \begin{bmatrix} A'_{11} & A'_{12} & A'_{13} \\ 0 & A'_{22} & A'_{23} \\ 0 & 0 & A'_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b'_1 \\ b'_2 \\ b'_3 \end{bmatrix}$$

## Cost

- ▶ Gaussian elimination:  $2(n-1)n(n+1)/3 + n(n-1)$  flop
- ▶ Back substitution:  $n^2$  flop
- ▶ Largest term:  $2/3 n^3$  flop

## Conditions of use

- ▶ The method must be modified if, at one step,  $A_{kk}^{(k)} = 0$ .
- ▶ The method can be used without modification . . .
  - if **A** is diagonally dominant per line or per column
  - if **A** is symmetric positive definite
- ▶ For the other cases  $\Rightarrow$  Permutation of lines and/or columns

## Gaussian elimination with pivoting

Data:  $\mathbf{A}$  et  $\mathbf{b}$ **for**  $k = 1, \dots, n - 1$  **do**Find  $r \in [k, \dots, n]$  such that  $|A_{rk}|$  is max. (if  $\max_r |A_{rk}| = 0$ , then  $\mathbf{A}$  not invers.)Swap the  $r^{\text{th}}$  and  $k^{\text{th}}$  lines of  $\mathbf{A}$  and  $\mathbf{b}$ **for**  $i = k + 1, \dots, n$  **do**

$$\alpha \leftarrow A_{ik}/A_{kk}$$

$$A_{ij} \leftarrow A_{ij} - \alpha A_{kj} \quad (j = k + 1, \dots, n)$$

$$A_{ik} \leftarrow 0$$

$$b_i \leftarrow b_i - \alpha b_k$$

**end****end****Possible pivoting strategies**

- ▶ Find  $r \in [k, \dots, n]$  such that  $|A_{kr}|$  is maximum (*partial pivoting*)
- ▶ Find  $r \in [k, \dots, n]$  and  $s \in [k, \dots, n]$  such that  $|A_{rs}|$  is maximum (*total pivoting*)

One can always find  $r \in [k, \dots, n]$  such that  $|A_{rk}| > 0$ , otherwise  $\mathbf{A}$  is not invertible.

⇒ The Gaussian elimination with pivoting is applicable to every invertible matrix!

## Direct methods for linear systems

*Solution of triangular systems*

*Gaussian elimination*

*Matrix factorization*

## Principle

- ▶ Factorize  $A$  into triangular matrices:

$$\underbrace{\begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix}}_A = \underbrace{\begin{bmatrix} \times & 0 & 0 \\ \times & \times & 0 \\ \times & \times & \times \end{bmatrix}}_L \underbrace{\begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \end{bmatrix}}_U$$

- ▶ Rewrite the problem with triangular matrices:

$$\mathbf{Ax} = \mathbf{b} \Leftrightarrow \mathbf{LUx} = \mathbf{b} \Leftrightarrow \begin{cases} \mathbf{Ly} = \mathbf{b} \\ \mathbf{Ux} = \mathbf{y} \end{cases}$$

and solve these problems with forward/back substitutions.

## Comments

- ▶ A priori, same cost than Gaussian elimination.
- ▶ If the system must be solved with several right-hand sides, only one factorization is necessary.

### Several factorizations

If  $\mathbf{A}$  is invertible and factorizable (*to define later*), one has:

$$\mathbf{A} = \tilde{\mathbf{L}}\mathbf{U} \quad (\text{Gauss})$$

$$\mathbf{A} = \mathbf{L}\tilde{\mathbf{U}} \quad (\text{Gauss})$$

$$\mathbf{A} = \tilde{\mathbf{L}}\mathbf{D}\tilde{\mathbf{U}} \quad (\text{Gauss-Jordan})$$

$$\mathbf{A} = \tilde{\mathbf{L}}\mathbf{D}\tilde{\mathbf{L}}^T \quad (\text{Crout}) \quad \text{Si } \mathbf{A} \text{ symétrique.}$$

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T \quad (\text{Cholesky}) \quad \text{Si } \mathbf{A} \text{ symétrique définie positive (SDP).}$$

with

$\mathbf{D}$  – diagonal matrix

$\mathbf{L}$  – lower triangular matrix

$\tilde{\mathbf{L}}$  – lower triangular matrix with unit diagonal

$\mathbf{U}$  – upper triangular matrix

$\tilde{\mathbf{U}}$  – upper triangular matrix with unit diagonal



# Gaussian factorization ( $\mathbf{A} = \tilde{\mathbf{L}}\mathbf{U}$ ) [1/4]

## Procedure

- Initialization:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} A_{11}^{(1)} & A_{12}^{(1)} & \cdots & A_{1n}^{(1)} \\ A_{21}^{(1)} & A_{22}^{(1)} & \cdots & A_{2n}^{(1)} \\ \vdots & \vdots & & \vdots \\ A_{n1}^{(1)} & A_{n2}^{(1)} & \cdots & A_{nn}^{(1)} \end{bmatrix}$$

- Iteration 1:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & \cdots & \cdots & 0 \\ \alpha_{21} & 1 & & & 0 \\ \alpha_{31} & 0 & 1 & & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ \alpha_{n1} & 0 & \cdots & \cdots & 1 \end{bmatrix} \begin{bmatrix} A_{11}^{(1)} & A_{12}^{(1)} & \cdots & A_{1n}^{(1)} \\ 0 & A_{22}^{(2)} & \cdots & A_{2n}^{(2)} \\ 0 & A_{32}^{(2)} & \cdots & A_{3n}^{(2)} \\ \vdots & \vdots & & \vdots \\ 0 & A_{n2}^{(2)} & \cdots & A_{nn}^{(2)} \end{bmatrix}$$

with

$$\begin{aligned} \alpha_{i1} &= A_{i1}^{(1)} / A_{11}^{(1)} & i &= 2, \dots, n \\ A_{ij}^{(2)} &= A_{ij}^{(1)} - \alpha_{i1} A_{1j}^{(1)} & i, j &= 2, \dots, n \end{aligned}$$

# Gaussian factorization ( $\mathbf{A} = \tilde{\mathbf{L}}\mathbf{U}$ ) [2/4]

## Procedure (continuation)

### ► Iteration $k$ :

$$\begin{bmatrix}
 1 & 0 & \cdots & \cdots & \cdots & 0 \\
 \alpha_{21} & & & & & 0 \\
 \vdots & \ddots & & & & \vdots \\
 \vdots & & 1 & & & \vdots \\
 \vdots & & \vdots & & 1 & \vdots \\
 \vdots & & \vdots & & \vdots & \vdots \\
 \alpha_{n1} & \cdots & \alpha_{nk} & 0 & \cdots & 1
 \end{bmatrix}
 \begin{bmatrix}
 A_{11}^{(1)} & A_{12}^{(1)} & \cdots & \cdots & \cdots & A_{1n}^{(1)} \\
 0 & A_{22}^{(2)} & & & & A_{2n}^{(2)} \\
 \vdots & & \ddots & & & \vdots \\
 0 & \cdots & 0 & A_{k+1,k+1}^{(k)} & \cdots & A_{k+1,n}^{(k)} \\
 \vdots & & \vdots & \vdots & & \vdots \\
 0 & \cdots & 0 & A_{n,k+1}^{(k)} & \cdots & A_{nn}^{(k)}
 \end{bmatrix}$$

### ► Iteration $n - 1$ :

$$\begin{bmatrix}
 1 & 0 & \cdots & \cdots & 0 \\
 \alpha_{21} & 1 & & & 0 \\
 \vdots & \ddots & & & \vdots \\
 \vdots & & 1 & & \vdots \\
 \alpha_{n1} & \cdots & \cdots & \alpha_{n(n-1)} & 1
 \end{bmatrix}
 \begin{bmatrix}
 A_{11}^{(1)} & A_{12}^{(1)} & \cdots & \cdots & A_{1n}^{(1)} \\
 0 & A_{22}^{(2)} & & & A_{2n}^{(2)} \\
 \vdots & & \ddots & & \vdots \\
 0 & & & & \vdots \\
 0 & & & & A_{nn}^{(n)}
 \end{bmatrix}$$

# Gaussian factorization ( $\mathbf{A} = \tilde{\mathbf{L}}\mathbf{U}$ ) [3/4]

## Gaussian factorization

Data:  $\mathbf{A}$

Initialization:  $\tilde{\mathbf{L}} = \mathbf{I}$  et  $\mathbf{U} = \mathbf{A}$

**for**  $k = 1, \dots, n - 1$  **do**

**for**  $i = k + 1, \dots, n$  **do**

$$\tilde{L}_{ik} \leftarrow U_{ik}/U_{kk}$$

$$U_{ij} \leftarrow U_{ij} - \tilde{L}_{ik}U_{kj} \quad (j = k + 1 \dots n)$$

$$U_{ik} \leftarrow 0$$

**end**

**end**

Si  $U_{kk} \neq 0$  !

## Gaussian factorization (*Rewriting*)

Data:  $\mathbf{A}$

**for**  $k = 1, \dots, n - 1$  **do**

**for**  $i = k + 1, \dots, n$  **do**

$$A_{ik} \leftarrow A_{ik}/A_{kk}$$

$$A_{ij} \leftarrow A_{ij} - A_{ik}A_{kj} \quad (j = k + 1 \dots n)$$

**end**

**end**

Si  $A_{kk} \neq 0$  !

# Gaussian factorization ( $\mathbf{A} = \tilde{\mathbf{L}}\mathbf{U}$ ) [4/4]

## Theorem – Unicity of the Gaussian factorization ( $\mathbf{A} = \tilde{\mathbf{L}}\mathbf{U}$ )

The Gaussian factorization, if it exists, is unique.

## Theorem – Existence of the Gaussian factorization ( $\mathbf{A} = \tilde{\mathbf{L}}\mathbf{U}$ ) *(not necessary)*

Every matrix that is symmetric positive definite is factorizable.

## Theorem – Existence of the Gaussian factorization after permutation ( $\mathbf{P}\mathbf{A} = \tilde{\mathbf{L}}\mathbf{U}$ )

For every invertible matrix, there exists a sequence of elementary permutations such that the permuted matrix admits a Gaussian factorization.

### Permutation matrix

- ▶ An elementary permutation matrix  $\mathbf{P}^{(i_1, i_2)}$  is a matrix which the application on a matrix  $\mathbf{A}$  permutes the lines  $i_1$  and  $i_2$ :

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix} = \begin{bmatrix} A_{31} & A_{32} & A_{33} & A_{34} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{11} & A_{12} & A_{13} & A_{14} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix}$$

- ▶ The permutation matrix  $\mathbf{P}^{(i_1, i_2)}$  is obtained by swapping the lines  $i_1$  and  $i_2$  and the columns  $i_1$  and  $i_2$  from the identity matrix.

Interesting properties:  $\mathbf{P}^{(i_1, i_2)}\mathbf{P}^{(i_1, i_2)} = \mathbf{I}_n$  et  $(\mathbf{P}^{(i_1, i_2)})^T = \mathbf{P}^{(i_1, i_2)}$

## Solution after factorization

### Gaussian factorization (*Rewriting*)

Data:  $A$

**for**  $k = 1, \dots, n - 1$  **do**

**for**  $i = k + 1, \dots, n$  **do**

$$A_{ik} \leftarrow A_{ik}/A_{kk}$$

Si  $A_{kk} \neq 0!$

$$A_{ij} \leftarrow A_{ij} - A_{ik}A_{kj} \quad (j = k + 1 \dots n)$$

**end**

**end**

### Cost of the entire solution procedure

- Gaussian or Gauss-Jordan factorization:  $2/3 n^3$  operations
- Crout or Cholesky factorization:  $1/3 n^3$  operations
- Forward and back substitution:  $2 n^2$  operations
- Additional operations with pivoting strategies

### Parallelization

- Parallelization not easy for problems with dense matrices
- Strategies are possible with blocks approaches.
- Strategies are possible with sparse matrices.

## Summary

### ► Triangular systems

- Simple computational procedures, but not suited for parallel computing!
- Computational cost:
  - By points:  $n^2$  scalar operations
  - By blocks:  $N$  small systems to solve and  $\mathcal{O}(N^2)$  matrix-vector products
- Parall. comp.: by-block strategy (*dense mat.*) or ad-hoc strategy (*sparse mat.*)

### ► General systems

- Approaches:
  - Gaussian elimination  $\Rightarrow$  Gives the solution for a given vector  $\mathbf{b}$
  - LU factorization + Two triangular systems to solve  
 $\Rightarrow$  Factorization computed once and used for any vector  $\mathbf{b}$
- For any nonsingular matrix  $\mathbf{A}$ ,  $\exists$  permutation matrix  $\mathbf{P}$  such that  $\mathbf{PA} = \tilde{\mathbf{L}}\mathbf{U}$
- Different factorizations:  $\tilde{\mathbf{L}}\mathbf{U}$ ,  $\tilde{\mathbf{L}}\mathbf{D}\tilde{\mathbf{U}}$  (Gen),  $\tilde{\mathbf{L}}\mathbf{D}\tilde{\mathbf{L}}^\top$  (Sym),  $\mathbf{L}\mathbf{L}^\top$  (SDP)
- Computational cost:  $\mathcal{O}(n^3)$  operations
- Parall. comp.: by-block strategy (*dense mat.*) or ad-hoc strategy (*sparse mat.*)

## Resources

- ▶ *Méthodes Numériques : Algorithmes, analyse et applications*  
A. Quarteroni, R. Sacco, F. Saleri (2007), Springer
- ▶ *Calcul scientifique parallèle*  
F. Magoulès et F.-X. Roux (2017), Dunod
- ▶ *Calcul scientifique parallèle*  
P. Ciarlet et E. Jamelot, polycopié de cours