Analyse Numérique I

Sup'Galilée, Ingénieurs MACS, 1ère année

François Cuvelier

Laboratoire d'Analyse Géométrie et Applications
Institut Galilée
Université Paris XIII.

2016/09/26

Chapitre IV

Résolution de systèmes linéaires

Plan

Soient $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ une matrice inversible et $\boldsymbol{b} \in \mathbb{K}^n$.

Résoudre

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$$

Le calcul de la matrice inverse $\mathbb{A}^{-1}$ revient à résoudre n systèmes linéaires.

Pour résoudre un système linéaire, on ne calcule pas la matrice inverse associée.

- **Méthodes directes :** On cherche $\mathbb{M}$ inversible tel que $\mathbb{MA}$ *facilement* inversible

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \mathbb{MA}\boldsymbol{x} = \mathbb{M}\boldsymbol{b}.$$

- **Méthodes itératives :** On cherche $\mathbb{B}$ et $\boldsymbol{c}$,

$$\boldsymbol{x}^{[k+1]} = \mathbb{B}\boldsymbol{x}^{[k]} + \boldsymbol{c}, \quad k \geqslant 0, \ \boldsymbol{x}^{[0]} \text{ donné}$$

en espérant $\lim_{k \to +\infty} \boldsymbol{x}^{[k]} = \boldsymbol{x}$.

Plan

Système diagonal

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ diagonale inversible et $\boldsymbol{b} \in \mathbb{K}^n$.

$$x_i = b_i / A_{i,i}, \quad \forall i \in [\![1, n]\!]. \tag{1}$$

Algorithme 1 Fonction RSLMatDiag permettant de résoudre le système linéaire à matrice diagonale inversible

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}.$$

Données : $\mathbb{A}$: matrice diagonale de $\mathcal{M}_n(\mathbb{R})$ inversible.
$\boldsymbol{b}$: vecteur de $\mathbb{R}^n$.
Résultat : $\boldsymbol{x}$: vecteur de $\mathbb{R}^n$.

1: **Fonction** $\boldsymbol{x} \leftarrow$ RSLMatDiag $(\mathbb{A}, \boldsymbol{b})$
2: **Pour** $i \leftarrow 1$ **à** n **faire**
3: $x(i) \leftarrow b(i)/A(i,i)$
4: **Fin Pour**
5: **Fin Fonction**

Système triangulaire inférieur

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ triangulaire inférieure inversible ($A_{i,j} = 0$ si $i < j$)

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \begin{pmatrix} A_{1,1} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ A_{n,1} & \dots & \dots & A_{n,n} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ \vdots \\ b_n \end{pmatrix}$$

$\mathbb{A}$ inversible $\iff$

Système triangulaire inférieur

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ triangulaire inférieure inversible ($A_{i,j} = 0$ si $i < j$)

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \begin{pmatrix} A_{1,1} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ A_{n,1} & \dots & \dots & A_{n,n} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ \vdots \\ b_n \end{pmatrix}$$

$$\mathbb{A} \text{ inversible} \iff A_{i,i} \neq 0, \forall i \in [\![1, n]\!]$$

Système triangulaire inférieur

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ triangulaire inférieure inversible ($A_{i,j} = 0$ si $i < j$)

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \begin{pmatrix} A_{1,1} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ A_{n,1} & \dots & \dots & A_{n,n} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ \vdots \\ b_n \end{pmatrix}$$

$$\mathbb{A} \text{ inversible} \iff A_{i,i} \neq 0, \forall i \in [\![1, n]\!]$$

Soit $i \in [\![1, n]\!]$, $\quad (\mathbb{A}\boldsymbol{x})_i = b_i, \iff \sum_{j=1}^{n} A_{i,j} x_j = b_i.$

$$b_i = \sum_{j=1}^{i-1} A_{i,j} x_j + A_{i,i} x_i + \sum_{j=i+1}^{n} \underbrace{A_{i,j}}_{=0} x_j = \sum_{j=1}^{i-1} A_{i,j} x_j + A_{i,i} x_i$$

$$x_i = \frac{1}{A_{i,i}} \left(b_i - \sum_{j=1}^{i-1} A_{i,j} x_j \right), \ \forall i \in [\![1, n]\!]. \tag{2}$$

$$x_i = \frac{1}{A_{i,i}} \left(b_i - \sum_{j=1}^{i-1} A_{i,j} x_j \right), \ \forall i \in [\![1, n]\!].$$

Algorithme 2 $\mathcal{R}_0$

1: Résoudre $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$ en calculant successivement $x_1, x_2, \ldots, x_n$.

Algorithme 2 $\mathcal{R}_1$

1: **Pour** $i \leftarrow 1$ **à** n **faire**

2: $x_i \leftarrow \frac{1}{A_{i,i}} \left(b_i - \sum_{j=1}^{i-1} A_{i,j} x_j \right)$

3: **Fin Pour**

$$x_i = \frac{1}{A_{i,i}} \left(b_i - \sum_{j=1}^{i-1} A_{i,j} x_j \right), \ \forall i \in [\![1, n]\!].$$

Algorithme 2 $\mathcal{R}_1$

1: **Pour** $i \leftarrow 1$ **à** n **faire**

2: $x_i \leftarrow \frac{1}{A_{i,i}} \left(b_i - \sum_{j=1}^{i-1} A_{i,j} x_j \right)$

3: **Fin Pour**

Algorithme 2 $\mathcal{R}_2$

1: **Pour** $i \leftarrow 1$ **à** n **faire**

2: $S \leftarrow \sum_{j=1}^{i-1} A_{i,j} x_j$

3: $x_i \leftarrow (b_i - S)/A_{i,i}$

4: **Fin Pour**

$$x_i = \frac{1}{A_{i,i}} \left(b_i - \sum_{j=1}^{i-1} A_{i,j} x_j \right), \quad \forall i \in [\![1, n]\!].$$

Algorithme 2 $\mathcal{R}_3$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $S \leftarrow \sum_{j=1}^{i-1} A_{i,j} x_j$
3: $x_i \leftarrow (b_i - S)/A_{i,i}$
4: **Fin Pour**

Algorithme 2 $\mathcal{R}_4$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $S \leftarrow 0$
3: **Pour** $j \leftarrow 1$ **à** $i - 1$ **faire**
4: $S \leftarrow S + A(i,j) * x(j)$
5: **Fin Pour**
6: $x_i \leftarrow (b_i - S)/A_{i,i}$
7: **Fin Pour**

$$x_i = \frac{1}{A_{i,i}} \left(b_i - \sum_{j=1}^{i-1} A_{i,j} x_j \right), \ \forall i \in [\![1, n]\!].$$

Algorithme 2 Fonction RSLTRIINF permettant de résoudre le système linéaire triangulaire inférieur inversible

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}.$$

Données : $\mathbb{A}$: matrice triangulaire de $\mathcal{M}_n(\mathbb{K})$ inférieure inversible.
$\boldsymbol{b}$: vecteur de $\mathbb{K}^n$.
Résultat : $\boldsymbol{x}$: vecteur de $\mathbb{K}^n$.

1: **Fonction** $\boldsymbol{x} \leftarrow$ RSLTRIINF ($\mathbb{A}, \boldsymbol{b}$)
2: **Pour** $i \leftarrow 1$ **à** n **faire**
3: $S \leftarrow 0$
4: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
5: $S \leftarrow S + A(i,j) * x(j)$
6: **Fin Pour**
7: $x(i) \leftarrow (b(i) - S)/A(i,i)$
8: **Fin Pour**
9: **Fin Fonction**

Système triangulaire supérieur

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ triangulaire supérieure inversible ($A_{i,j} = 0$ si $i < j$)

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \begin{pmatrix} A_{1,1} & \dots & \dots & A_{1,n} \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & A_{n,n} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ \vdots \\ b_n \end{pmatrix}$$

Exercice

Ecrire la fonction RSLTriSup permettant de résoudre le système triangulaire supérieure $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$.

Plan

Lemme 1.1:

Soit $(i,j) \in [\![1,n]\!]^2$. On note $\mathbb{P}_n^{[i,j]} \in \mathcal{M}_n(\mathbb{R})$ la matrice identitée dont on a permuté les lignes i et j. Alors la matrice $\mathbb{P}_n^{[i,j]}$ est symétrique et orthogonale. Pour toute matrice $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$,

1. la matrice $\mathbb{P}_n^{[i,j]}\mathbb{A}$ est matrice $\mathbb{A}$ dont on a permuté les **lignes** i et j,
2. la matrice $\mathbb{A}\mathbb{P}_n^{[i,j]}$ est matrice $\mathbb{A}$ dont on a permuté les **colonnes** i et j,

Lemme 1.2:

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ avec $A_{1,1} \neq 0$. Il existe une matrice $\mathbb{E} \in \mathcal{M}_n(\mathbb{C})$ triangulaire inférieure à diagonale unité telle que

$$\mathbb{E}\mathbb{A}\boldsymbol{e}_1 = A_{1,1}\boldsymbol{e}_1 \tag{3}$$

où $\boldsymbol{e}_1$ est le premier vecteur de la base canonique de $\mathbb{C}^n$.

Théorème 2:

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$. Il existe une matrice unitaire $\mathbb{U}$ et une matrice triangulaire supérieure $\mathbb{T}$ telles que

$$\mathbb{A} = \mathbb{U}\mathbb{T}\mathbb{U}^* \tag{4}$$

Plan

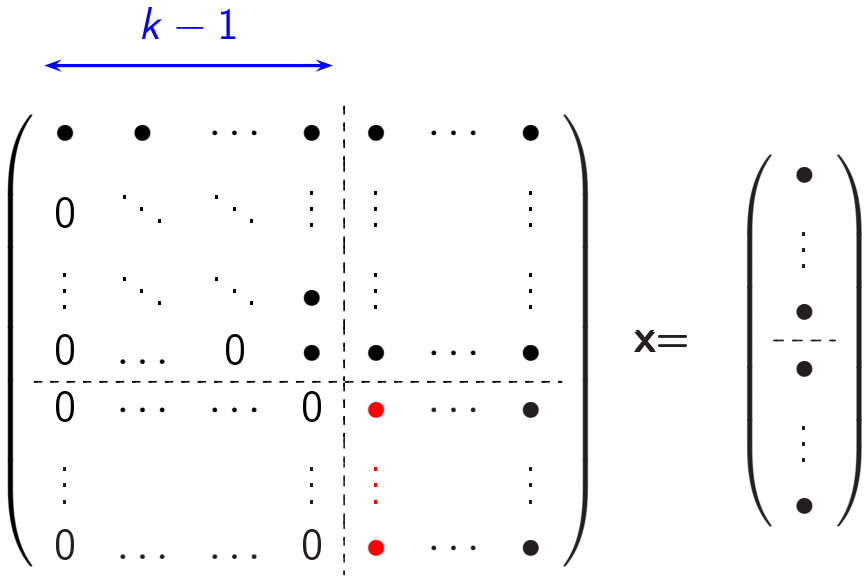
Algorithme de Gauss-Jordan

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \mathbb{U}\boldsymbol{x} = \boldsymbol{f}$$

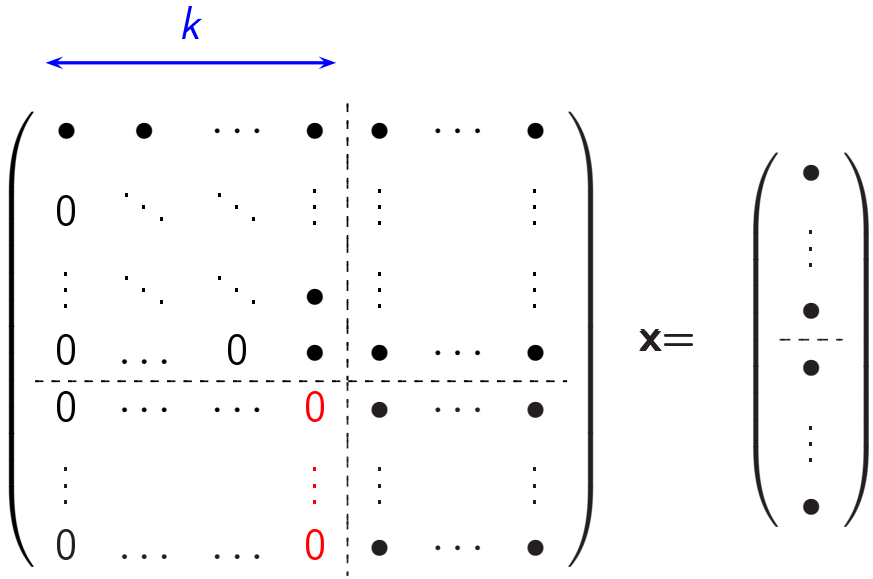
où $\mathbb{U}$ matrice triangulaire supérieure.
Opérations élémentaires sur les matrices :

- $\mathcal{L}_i \leftrightarrow \mathcal{L}_j$ permutation lignes i et j
- $\mathcal{L}_i \leftarrow \mathcal{L}_i + \alpha\mathcal{L}_j$ combinaison linéaire

A l'aide d'opérations élémentaires, on va transformer successivement en $n-1$ étapes le système. A l'étape k, on va *s'arranger* pour annuler les termes sous-diagonaux de la colonne k de la matrice sans modifier les $k-1$ premières colonnes.



Etape k $\Longleftrightarrow$



Algorithme 3 Algorithme de Gauss-Jordan formel pour la résolution de $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$

1: **Pour** $j \leftarrow 1$ **à** $n-1$ **faire**
2: Rechercher l'indice k de la ligne du pivot (sur la colonne j, $k \in [\![j, n]\!]$)
3: Permuter les lignes j $(\mathcal{L}_j)$ et k $(\mathcal{L}_k)$ du système si besoin.
4: **Pour** $i \leftarrow j+1$ **à** n **faire**
5: Eliminer en effectuant $\mathcal{L}_i \leftarrow \mathcal{L}_i - \frac{A_{i,j}}{A_{j,j}}\mathcal{L}_j$
6: **Fin Pour**
7: **Fin Pour**
8: Résoudre le système triangulaire supérieur par la méthode de la remontée.

Algorithme 4 Algorithme de Gauss-Jordan avec fonctions pour la résolution de $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$

Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{K})$ inversible.
$\boldsymbol{b}$: vecteur de $\mathbb{K}^n$.

Résultat : $\boldsymbol{x}$: vecteur de $\mathbb{R}^n$.

1: **Fonction** $\boldsymbol{x} \leftarrow$ RSLGauss $(\mathbb{A}, \boldsymbol{b})$
2: **Pour** $j \leftarrow 1$ **à** $n-1$ **faire**
3: $k \leftarrow$ ChercheIndPivot $(\mathbb{A}, j)$ ▷ à écrire
4: $[\mathbb{A}, \boldsymbol{b}] \leftarrow$ PermLignesSys $(\mathbb{A}, \boldsymbol{b}, j, k)$ ▷ à écrire
5: **Pour** $i \leftarrow j+1$ **à** n **faire**
6: $[\mathbb{A}, \boldsymbol{b}] \leftarrow$ CombLignesSys $(\mathbb{A}, \boldsymbol{b}, j, i, -A(i,j)/A(j,j))$ ▷ à écrire
7: **Fin Pour**
8: **Fin Pour**
9: $\boldsymbol{x} \leftarrow$ RSLTriSup$(\mathbb{A}, \boldsymbol{b})$ ▷ déjà écrite
10: **Fin Fonction**

Algorithme 5 Recherche d'un pivot pour l'algorithme de Gauss-Jordan.

Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{K})$.
j : entier, $1 \leqslant j \leqslant n$.
Résultat : k : entier, indice ligne pivot

```
1: Fonction k ← CherchеIndPivot ( 𝔸, j )
2:    k ← j, pivot ← |𝔸(j, j)|
3:    Pour i ← j + 1 à n faire
4:       Si |𝔸(i, j)| > pivot alors
5:          k ← i
6:          pivot ← |𝔸(i, j)|
7:       Fin Si
8:    Fin Pour
9: Fin Fonction
```

Algorithme 6 Permutte deux lignes d'une matrice et d'un vecteur.

Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{K})$.
$\boldsymbol{b}$: vecteur de $\mathbb{K}^n$.
j, k : entiers, $1 \leqslant j, k \leqslant n$.
Résultat : $\mathbb{A}$ et $\boldsymbol{b}$ modifiés.

```
1: Fonction [𝔸, b] ← PermLignesSys ( 𝔸, b, j, k )
2:    Pour l ← 1 à n faire
3:       t ← 𝔸(j, l)
4:       𝔸(j, l) ← 𝔸(k, l)
5:       𝔸(k, l) ← t
6:    Fin Pour
7:    t ← b(j), b(j) ← b(k), b(k) ← t
8: Fin Fonction
```

Algorithme 7 Combinaison linéaire $\mathcal{L}_i \leftarrow \mathcal{L}_i + \alpha\mathcal{L}_j$ appliqué à une matrice et à un vecteur.

Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{K})$.
$\boldsymbol{b}$: vecteur de $\mathbb{K}^n$.
j, i : entiers, $1 \leqslant j, i \leqslant n$.
alpha : scalaire de $\mathbb{K}$
Résultat : $\mathbb{A}$ et $\boldsymbol{b}$ modifiés.

```
1: Fonction [𝔸, b] ← CombLignesSys ( 𝔸, b, j, i, α )
2:    Pour k ← 1 à n faire
3:       𝔸(i, k) ← 𝔸(i, k) + α ∗ 𝔸(j, k)
4:    Fin Pour
5:    b(i) ← b(i) + αb(j)
6: Fin Fonction
```

Exercice: Méthode de Gauss, écriture algébrique

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ inversible.

Q. 1

Montrer qu'il existe une matrice $\mathbb{G} \in \mathcal{M}_n(\mathbb{C})$ telle que $|\det(\mathbb{G})| = 1$ et $\mathbb{G}\mathbb{A}\boldsymbol{e}_1 = \alpha \boldsymbol{e}_1$ avec $\alpha \neq 0$ et $\boldsymbol{e}_1$ premier vecteur de la base canonique de C^n.

Q. 2

1. *Montrer par récurrence sur l'ordre des matrices que pour toute matrice $\mathbb{A}_n \in \mathcal{M}_n(\mathbb{C})$ inversible, il existe une matrice $\mathbb{S}_n \in \mathcal{M}_n(\mathbb{C})$ telle que $|\det \mathbb{S}_n| = 1$ et $\mathbb{S}_n\mathbb{A}_n = \mathbb{U}_n$ avec $\mathbb{U}_n$ matrice triangulaire supérieure inversible.*
2. *Soit $\boldsymbol{b} \in \mathbb{C}^n$. En supposant connue la décompostion précédente $\mathbb{S}_n\mathbb{A}_n = \mathbb{U}_n$, expliquer comment résoudre le système $\mathbb{A}_n\boldsymbol{x} = \boldsymbol{b}$.*

Q. 3

Que peut-on dire si $\mathbb{A}$ est non inversible?

Indication : utiliser les Lemmes 1.1 et 1.2.

On a donc démontré le théorème suivant

Théorème 3

Soit $\mathbb{A}$ une matrice carrée, inversible ou non. Il existe (au moins) une matrice inversible $\mathbb{G}$ telle que $\mathbb{GA}$ soit triangulaire supérieure.

Plan

Exercice: Factorisation $\mathbb{LU}$

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice dont les sous-matrices principales d'ordre i, notées Δ_i, $i \in [\![1, n]\!]$ (voir Definition ??, page ??) sont inversibles. Montrer qu'il existe des matrices $\mathbb{E}^{[k]} \in \mathcal{M}_n(\mathbb{C})$, $k \in [\![1, n-1]\!]$, triangulaires inférieures à diagonale unité telles que la matrice $\mathbb{U}$ définie par

$$\mathbb{U} = \mathbb{E}^{[n-1]} \dots \mathbb{E}^{[1]} \mathbb{A}$$

soit triangulaire supérieure avec $U_{i,i} = \det \Delta_i / (U_{1,1} \times \cdots \times U_{i-1,i-1})$, $\forall i \in [\![1, n]\!]$.

Théorème 4: Factorisation $\mathbb{LU}$

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice dont les sous-matrices principales sont inversibles alors il existe une unique matrice $\mathbb{L} \in \mathcal{M}_n(\mathbb{C})$ triangulaire inférieure (*lower triangular* en anglais) à diagonale unité et une unique matrice $\mathbb{U} \in \mathcal{M}_n(\mathbb{C})$ triangulaire supérieure (*upper triangular* en anglais) inversible telles ques

$$\mathbb{A} = \mathbb{LU}.$$

preuve :

- **Existence :** exercice précédant $\mathbb{U} = \mathbb{E}^{[n-1]} \dots \mathbb{E}^{[1]} \mathbb{A}$

$$\mathbb{L} = \left(\mathbb{E}^{[n-1]} \dots \mathbb{E}^{[1]} \right)^{-1}$$

- **Unicité :** $\mathbb{A} = \mathbb{L}_1\mathbb{U}_1 = \mathbb{L}_2\mathbb{U}_2 \dots$

Corollaire 4.1:

Si $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ est une matrice hermitienne définie positive alors elle admet une unique factorisation $\mathbb{LU}$.

preuve : $\mathbb{A}$ hermitienne définie positive alors toutes ses sous-matrices principales sont définies positives et donc inversibles.

Remarque 4.2

Si la matrice $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ est inversible mais que ses sous-matrices principales ne sont pas toutes inversibles, il est possible par des permutations préalables de lignes de la matrice de se ramener à une matrice telle que ses sous-matrices principales soient inversibles.

Théorème 5: Factorisation $\mathbb{LU}$ avec permutations

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice inversible. Il existe une matrice $\mathbb{P}$, produit de matrices de permutation, une matrice $\mathbb{L} \in \mathcal{M}_n(\mathbb{C})$ triangulaire inférieure à diagonale unité et une matrice $\mathbb{U} \in \mathcal{M}_n(\mathbb{C})$ triangulaire supérieure telles ques

$$\mathbb{PA} = \mathbb{LU}. \tag{5}$$

Utilisation pratique de la factorisation $\mathbb{LU}$

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ admettant une factorisation $\mathbb{LU}$

Trouver $\boldsymbol{x} \in \mathbb{K}^n$ tel que

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \mathbb{LU}\boldsymbol{x} = \boldsymbol{b} \tag{6}$$

est équivalent à

Utilisation pratique de la factorisation $\mathbb{LU}$

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ admettant une factorisation $\mathbb{LU}$

Trouver $\boldsymbol{x} \in \mathbb{K}^n$ tel que

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \mathbb{LU}\boldsymbol{x} = \boldsymbol{b} \tag{6}$$

est équivalent à

Trouver $\boldsymbol{x} \in \mathbb{K}^n$ solution de

$$\mathbb{U}\boldsymbol{x} = \boldsymbol{y} \tag{7}$$

avec $\boldsymbol{y} \in \mathbb{K}^n$ solution de

$$\mathbb{L}\boldsymbol{y} = \boldsymbol{b}. \tag{8}$$

Algorithme de résolution de systèmes linéaire par $\mathbb{LU}$

Algorithme 8 Fonction RSLFactLU permettant de résoudre, par une factorisation $\mathbb{LU}$, le système linéaire

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$$

où $\mathbb{A}$ une matrice de $\mathcal{M}_n(\mathbb{R})$ définie positive et $\boldsymbol{b} \in \mathbb{R}^n$.

Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{R})$ dont les sous-matrices principales sont inversibles définie positive,
$\boldsymbol{b}$: vecteur de $\mathbb{R}^n$.

Résultat : $\boldsymbol{x}$: vecteur de $\mathbb{R}^n$.

1: **Fonction** $\boldsymbol{x} \leftarrow$ RSLFactLU $(\mathbb{A}, \boldsymbol{b})$
2: $[\mathbb{L}, \mathbb{U}] \leftarrow$ FactLU$(\mathbb{A})$ ▷ Factorisation $\mathbb{LU}$
3: $\boldsymbol{y} \leftarrow$ RSLTriInf$(\mathbb{L}, \boldsymbol{b})$ ▷ Résolution du système $\mathbb{L}\boldsymbol{y} = \boldsymbol{b}$
4: $\boldsymbol{x} \leftarrow$ RSLTriSup$(\mathbb{U}, \boldsymbol{y})$ ▷ Résolution du système $\mathbb{U}\boldsymbol{x} = \boldsymbol{y}$
5: **Fin Fonction**

Il nous faut donc écrire la fonction FactLU

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ admettant une factorisation $\mathbb{LU}$.

$$\begin{pmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,n} \\ A_{2,1} & A_{2,2} & \dots & A_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n,1} & A_{n,2} & \dots & A_{n,n} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ L_{2,1} & 1 & 0 & \dots & 0 \\ L_{3,1} & L_{3,2} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ L_{n,1} & L_{n,2} & \dots & L_{n,n-1} & 1 \end{pmatrix} \begin{pmatrix} U_{1,1} & U_{1,2} & \dots & \dots & U_{1,n} \\ 0 & U_{2,2} & \dots & \dots & U_{2,n} \\ 0 & 0 & U_{3,3} & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & U_{n,n} \end{pmatrix}$$

On connait $\mathbb{A}$, on cherche $\mathbb{L}$ et $\mathbb{U}$

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ admettant une factorisation $\mathbb{LU}$.

$$\begin{pmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,n} \\ A_{2,1} & A_{2,2} & \dots & A_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n,1} & A_{n,2} & \dots & A_{n,n} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ L_{2,1} & 1 & 0 & \dots & 0 \\ L_{3,1} & L_{3,2} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ L_{n,1} & L_{n,2} & \dots & L_{n,n-1} & 1 \end{pmatrix} \begin{pmatrix} U_{1,1} & U_{1,2} & \dots & \dots & U_{1,n} \\ 0 & U_{2,2} & \dots & \dots & U_{2,n} \\ 0 & 0 & U_{3,3} & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & U_{n,n} \end{pmatrix}$$

- **Etape** 1 :
  - On connait la **première ligne** de $\mathbb{L} \Longrightarrow$ on peut calculer la **première ligne** de $\mathbb{U}$

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ admettant une factorisation $\mathbb{LU}$.

$$\begin{pmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,n} \\ A_{2,1} & A_{2,2} & \dots & A_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n,1} & A_{n,2} & \dots & A_{n,n} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ L_{2,1} & 1 & 0 & \dots & 0 \\ L_{3,1} & L_{3,2} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ L_{n,1} & L_{n,2} & \dots & L_{n,n-1} & 1 \end{pmatrix} \begin{pmatrix} U_{1,1} & U_{1,2} & \dots & \dots & U_{1,n} \\ 0 & U_{2,2} & \dots & \dots & U_{2,n} \\ 0 & 0 & U_{3,3} & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & U_{n,n} \end{pmatrix}$$

- **Etape 1 :**
  - On connait la **première ligne** de $\mathbb{L} \Longrightarrow$ on peut calculer la **première ligne** de $\mathbb{U}$
  - On connait la **première colonne** de $\mathbb{U} \Longrightarrow$ on peut calculer la **première colonne** de $\mathbb{U}$

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ admettant une factorisation $\mathbb{LU}$.

$$\begin{pmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,n} \\ A_{2,1} & A_{2,2} & \dots & A_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n,1} & A_{n,2} & \dots & A_{n,n} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ L_{2,1} & 1 & 0 & \dots & 0 \\ L_{3,1} & L_{3,2} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ L_{n,1} & L_{n,2} & \dots & L_{n,n-1} & 1 \end{pmatrix} \begin{pmatrix} U_{1,1} & U_{1,2} & \dots & \dots & U_{1,n} \\ 0 & U_{2,2} & \dots & \dots & U_{2,n} \\ 0 & 0 & U_{3,3} & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & U_{n,n} \end{pmatrix}$$

- **Etape 1 :**
  - On connait la **première ligne** de $\mathbb{L} \Longrightarrow$ on peut calculer la **première ligne** de $\mathbb{U}$
  - On connait la **première colonne** de $\mathbb{U} \Longrightarrow$ on peut calculer la **première colonne** de $\mathbb{U}$
- **Etape 2 :**
  - On connait la **deuxième ligne** de $\mathbb{L} \Longrightarrow$ on peut calculer la **deuxième ligne** de $\mathbb{U}$ car on connait la première ligne de $\mathbb{U}$

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ admettant une factorisation $\mathbb{LU}$.

$$\begin{pmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,n} \\ A_{2,1} & A_{2,2} & \dots & A_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n,1} & A_{n,2} & \dots & A_{n,n} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ L_{2,1} & 1 & 0 & \dots & 0 \\ L_{3,1} & L_{3,2} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ L_{n,1} & L_{n,2} & \dots & L_{n,n-1} & 1 \end{pmatrix} \begin{pmatrix} U_{1,1} & U_{1,2} & \dots & \dots & U_{1,n} \\ 0 & U_{2,2} & \dots & \dots & U_{2,n} \\ 0 & 0 & U_{3,3} & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & U_{n,n} \end{pmatrix}$$

- **Etape 1 :**
  - On connait la **première ligne** de $\mathbb{L} \Longrightarrow$ on peut calculer la **première ligne** de $\mathbb{U}$
  - On connait la **première colonne** de $\mathbb{U} \Longrightarrow$ on peut calculer la **première colonne** de $\mathbb{U}$
- **Etape 2 :**
  - On connait la **deuxième ligne** de $\mathbb{L} \Longrightarrow$ on peut calculer la **deuxième ligne** de $\mathbb{U}$ car on connait la première ligne de $\mathbb{U}$
  - On connait la **deuxième colonne** de $\mathbb{U} \Longrightarrow$ on peut calculer la **deuxième colonne** de $\mathbb{L}$ car on connait la première colonne de $\mathbb{L}$

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ admettant une factorisation $\mathbb{LU}$.

$$\begin{pmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,n} \\ A_{2,1} & A_{2,2} & \dots & A_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n,1} & A_{n,2} & \dots & A_{n,n} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ L_{2,1} & 1 & 0 & \dots & 0 \\ L_{3,1} & L_{3,2} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ L_{n,1} & L_{n,2} & \dots & L_{n,n-1} & 1 \end{pmatrix} \begin{pmatrix} U_{1,1} & U_{1,2} & \dots & \dots & U_{1,n} \\ 0 & U_{2,2} & \dots & \dots & U_{2,n} \\ 0 & 0 & U_{3,3} & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & U_{n,n} \end{pmatrix}$$

- **Etape 1 :**
  - On connait la **première ligne** de $\mathbb{L} \Longrightarrow$ on peut calculer la **première ligne** de $\mathbb{U}$
  - On connait la **première colonne** de $\mathbb{U} \Longrightarrow$ on peut calculer la **première colonne** de $\mathbb{U}$
- **Etape 2 :**
  - On connait la **deuxième ligne** de $\mathbb{L} \Longrightarrow$ on peut calculer la **deuxième ligne** de $\mathbb{U}$ car on connait la première ligne de $\mathbb{U}$
  - On connait la **deuxième colonne** de $\mathbb{U} \Longrightarrow$ on peut calculer la **deuxième colonne** de $\mathbb{L}$ car on connait la première colonne de $\mathbb{L}$
- ...

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ admettant une factorisation $\mathbb{LU}$.

$$\begin{pmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,n} \\ A_{2,1} & A_{2,2} & \dots & A_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n,1} & A_{n,2} & \dots & A_{n,n} \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ L_{2,1} & 1 & 0 & \dots & 0 \\ L_{3,1} & L_{3,2} & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ L_{n,1} & L_{n,2} & \dots & L_{n,n-1} & 1 \end{pmatrix} \begin{pmatrix} U_{1,1} & U_{1,2} & \dots & \dots & U_{1,n} \\ 0 & U_{2,2} & \dots & \dots & U_{2,n} \\ 0 & 0 & U_{3,3} & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & U_{n,n} \end{pmatrix}$$

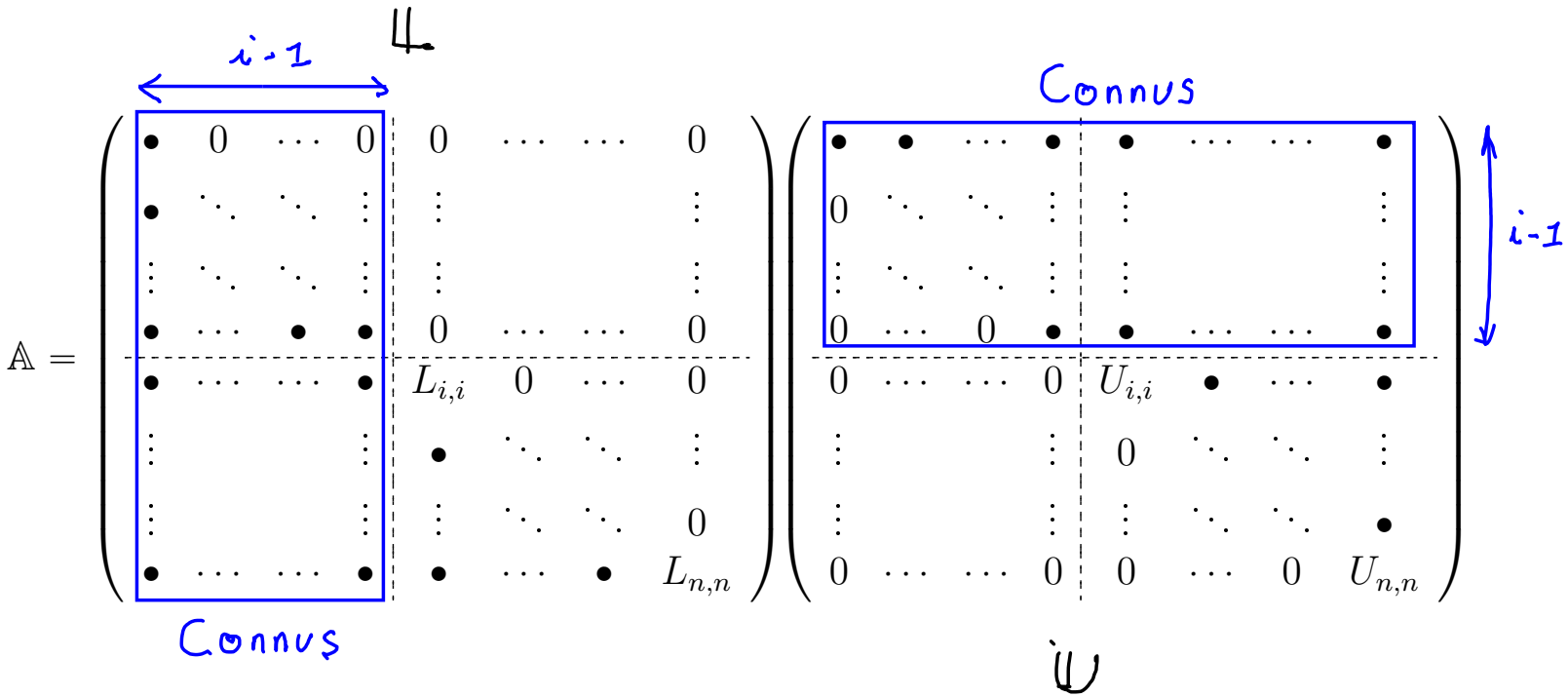
- **Etape i :**
  On connait les $i-1$ premières colonnes de $\mathbb{L}$ et les $i-1$ premières lignes de $\mathbb{U}$.
  Peut-on calculer la colonne i de $\mathbb{L}$ et la ligne i de $\mathbb{U}$?

Par récurrence, en supposant les $i-1$ premières colonnes de $\mathbb{L}$ et les $i-1$ premières lignes de $\mathbb{U}$.

$$\mathbb{A} = \left(\begin{array}{cccc|cccc} \bullet & 0 & \cdots & 0 & 0 & \cdots & \cdots & 0 \\ \bullet & \ddots & \ddots & \vdots & \vdots & & & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots & & & \vdots \\ \bullet & \cdots & \bullet & \bullet & 0 & \cdots & \cdots & 0 \\ \hline \bullet & \cdots & \cdots & \bullet & L_{i,i} & 0 & \cdots & 0 \\ \vdots & & & \vdots & \bullet & \ddots & \ddots & \vdots \\ \vdots & & & \vdots & \vdots & \ddots & \ddots & 0 \\ \bullet & \cdots & \cdots & \bullet & \bullet & \cdots & \bullet & L_{n,n} \end{array}\right) \left(\begin{array}{cccc|cccc} \bullet & \bullet & \cdots & \bullet & \bullet & \cdots & \cdots & \bullet \\ 0 & \ddots & \ddots & \vdots & \vdots & & & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots & & & \vdots \\ 0 & \cdots & 0 & \bullet & \bullet & \cdots & \cdots & \bullet \\ \hline 0 & \cdots & \cdots & 0 & U_{i,i} & \bullet & \cdots & \bullet \\ \vdots & & & \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & & & \vdots & \vdots & \ddots & \ddots & \bullet \\ 0 & \cdots & \cdots & 0 & 0 & \cdots & 0 & U_{n,n} \end{array}\right)$$

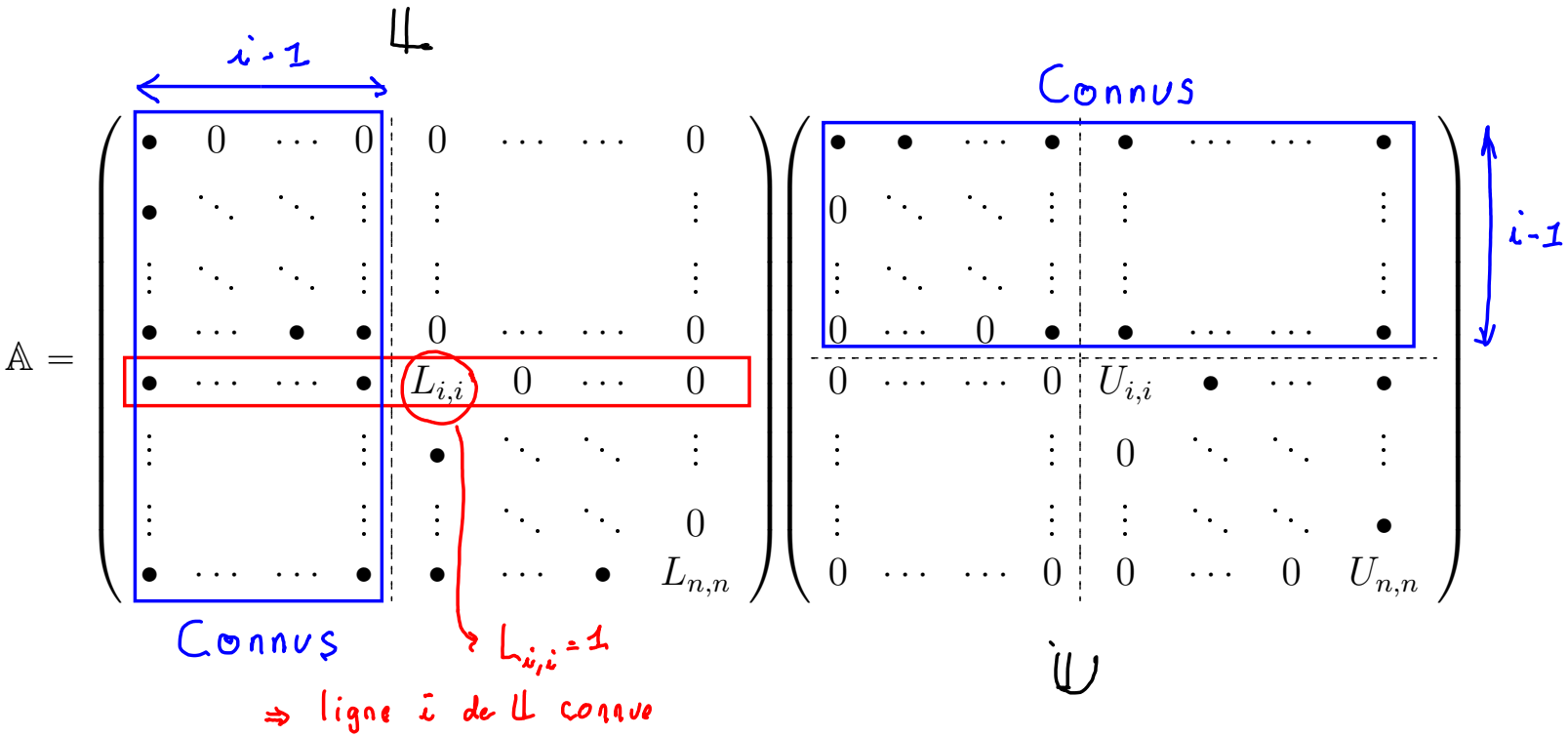
Par récurrence, en supposant les $i-1$ premières colonnes de $\mathbb{L}$ et les $i-1$ premières lignes de $\mathbb{U}$.

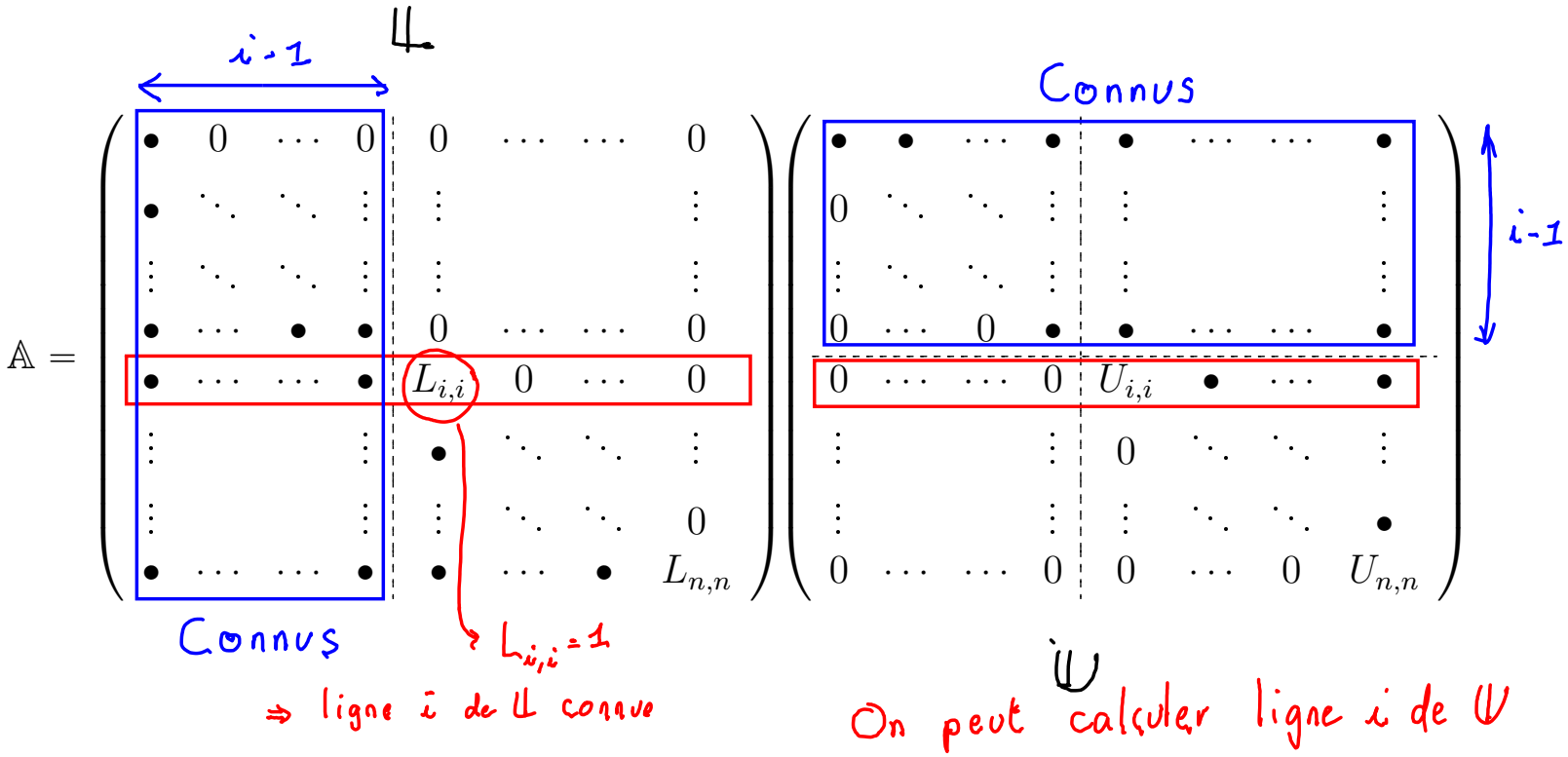
$$\mathbb{A} = \left(\begin{array}{cccc|cccc} \bullet & 0 & \cdots & 0 & 0 & \cdots & \cdots & 0 \\ \bullet & \ddots & \ddots & \vdots & \vdots & & & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots & & & \vdots \\ \bullet & \cdots & \bullet & \bullet & 0 & \cdots & \cdots & 0 \\ \hline \bullet & \cdots & \cdots & \bullet & L_{i,i} & 0 & \cdots & 0 \\ \vdots & & & \vdots & \bullet & \ddots & \ddots & \vdots \\ \vdots & & & \vdots & \vdots & \ddots & \ddots & 0 \\ \bullet & \cdots & \cdots & \bullet & \bullet & \cdots & \bullet & L_{n,n} \end{array}\right) \left(\begin{array}{cccc|cccc} \bullet & \bullet & \cdots & \bullet & \bullet & \cdots & \cdots & \bullet \\ 0 & \ddots & \ddots & \vdots & \vdots & & & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots & & & \vdots \\ 0 & \cdots & 0 & \bullet & \bullet & \cdots & \cdots & \bullet \\ \hline 0 & \cdots & \cdots & 0 & U_{i,i} & \bullet & \cdots & \bullet \\ \vdots & & & \vdots & 0 & \ddots & \ddots & \vdots \\ \vdots & & & \vdots & \vdots & \ddots & \ddots & \bullet \\ 0 & \cdots & \cdots & 0 & 0 & \cdots & 0 & U_{n,n} \end{array}\right)$$



Par récurrence, on suppose connues les $i-1$ premières colonnes de $\mathbb{L}$ et les $i-1$ premières lignes de $\mathbb{U}$.

Peut-on calculer la colonne i de $\mathbb{L}$ et la ligne i de $\mathbb{U}$?

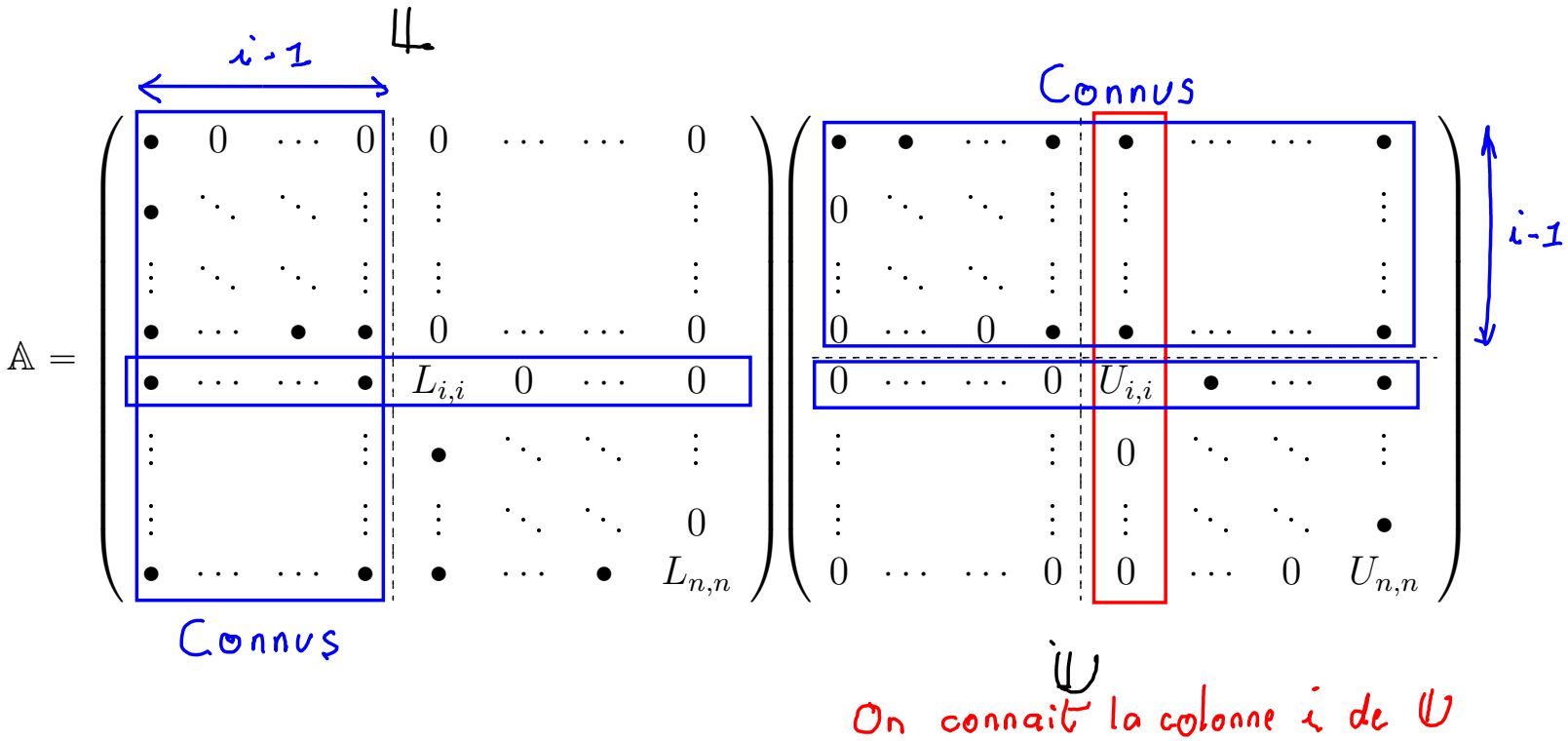

i-1
L
Connus
Connus
i-1
$\mathbb{A} =$
$L_{i,i}$
$U_{i,i}$
$L_{i,i} = 1$
⇒ ligne i de L connue
U

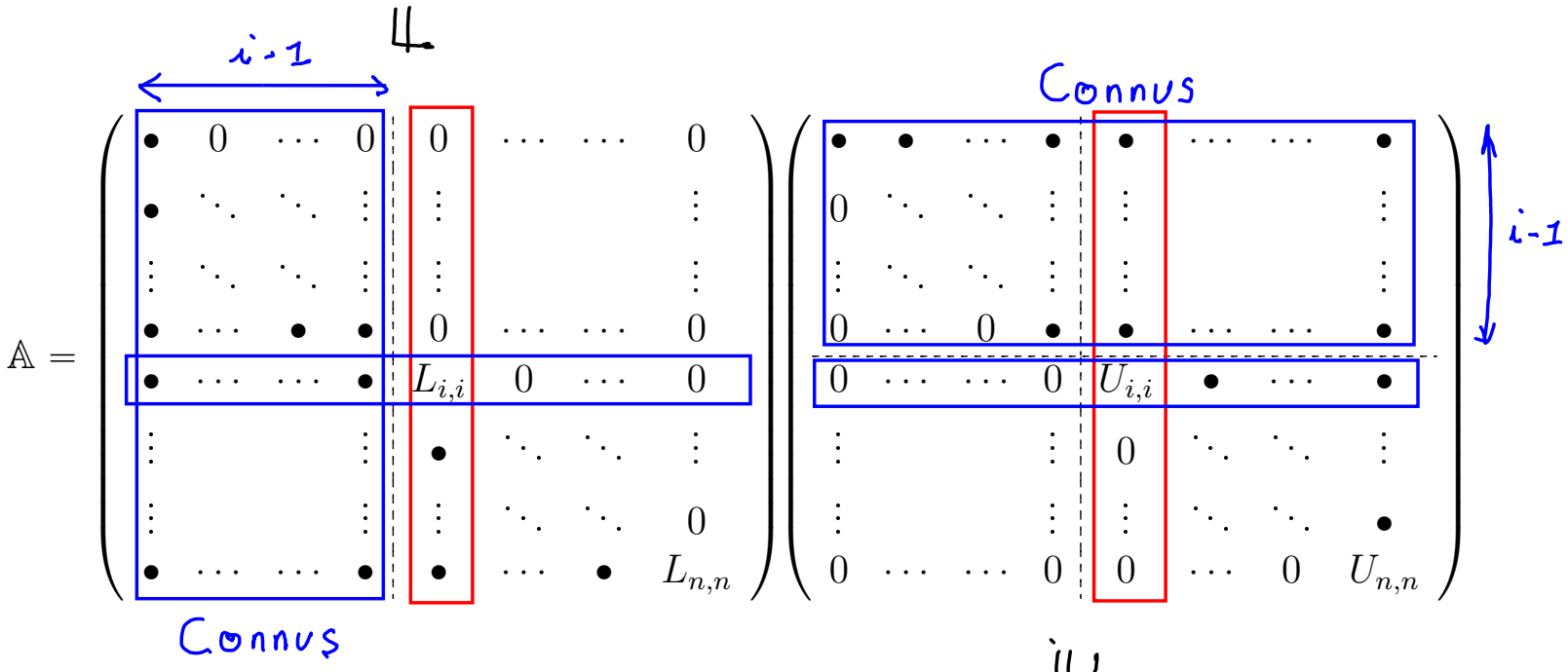


On cherche $U_{i,j}$ $\forall j \in [\![i, n]\!]$.

$$A_{i,j} \overset{\text{def}}{=} \sum_{k=1}^{n} L_{i,k} U_{k,j} = \sum_{k=1}^{i-1} \overbrace{L_{i,k} U_{k,j}}^{\text{connus}} + \overbrace{L_{i,i}}^{=1} U_{i,j} + \sum_{k=1}^{i+1} \overbrace{L_{i,k}}^{=0} U_{k,j}$$

$$U_{i,j} = A_{i,j} - \sum_{k=1}^{i-1} L_{i,k} U_{k,j}, \quad \forall j \in [\![i, n]\!].$$


L
i-1
Connus
$\mathbb{A} =$
$L_{i,i}$
$L_{n,n}$
Connus
U
i-1
$U_{i,i}$
$U_{n,n}$
On connait la colonne i de U



On peut calculer la colonne i de L $\Leftarrow$ On connait la colonne i de U

On cherche $L_{j,i}$ $\forall j \in [\![i+1, n]\!]$, $(L_{i,i} = 1)$

$$A_{j,i} \overset{\text{def}}{=} \sum_{k=1}^{n} L_{j,k} U_{k,i} = \sum_{k=1}^{i-1} \overbrace{L_{j,k} U_{k,i}}^{\text{connus}} + L_{j,i} \overbrace{U_{i,i}}^{\text{connu}} + \sum_{k=1}^{i+1} L_{j,k} \overbrace{U_{k,i}}^{=0}$$

$$L_{j,i} = \left(A_{j,i} - \sum_{k=1}^{i-1} L_{j,k} U_{k,i} \right) / U_{i,i}, \quad \forall j \in [\![i+1, n]\!].$$

Algorithme 9 $\boxed{\mathcal{R}_0}$

1: Calculer les matrices $\mathbb{L}$ et $\mathbb{U}$

Algorithme 9 $\boxed{\mathcal{R}_1}$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: Calculer la ligne i de $\mathbb{U}$.
3: Calculer la colonne i de $\mathbb{L}$.
4: **Fin Pour**

Algorithme 9 $\boxed{\mathcal{R}_1}$

1: **Pour** $i \leftarrow 1$ à n **faire**
2: Calculer la ligne i de $\mathbb{U}$.
3: Calculer la colonne i de $\mathbb{L}$.
4: **Fin Pour**

Algorithme 9 $\boxed{\mathcal{R}_2}$

1: **Pour** $i \leftarrow 1$ à n **faire**
2: **Pour** $j \leftarrow 1$ à $i-1$ **faire**
3: $U(i,j) \leftarrow 0$
4: **Fin Pour**
5: **Pour** $j \leftarrow i$ à n **faire**
6: $U_{i,j} \leftarrow A_{i,j} - \sum_{k=1}^{i-1} L_{i,k} U_{k,j}$
7: **Fin Pour**
8: **Pour** $j \leftarrow 1$ à $i-1$ **faire**
9: $L_{j,i} \leftarrow 0$
10: **Fin Pour**
11: $L_{i,i} \leftarrow 1$
12: **Pour** $j \leftarrow i+1$ à n **faire**
13: $L_{j,i} \leftarrow \frac{1}{U_{i,i}} \left(A_{j,i} - \sum_{k=1}^{i-1} L_{j,k} U_{k,i} \right)$
14: **Fin Pour**
15: **Fin Pour**

Algorithme 9 $\mathcal{R}_2$

1: **Pour** $i \leftarrow 1$ à n **faire**
2: **Pour** $j \leftarrow 1$ à $i-1$ **faire**
3: $U(i,j) \leftarrow 0$
4: **Fin Pour**
5: **Pour** $j \leftarrow i$ à n **faire**
6: $U_{i,j} \leftarrow A_{i,j} - \sum_{k=1}^{i-1} L_{i,k} U_{k,j}$
7: **Fin Pour**
8: **Pour** $j \leftarrow 1$ à $i-1$ **faire**
9: $L_{j,i} \leftarrow 0$
10: **Fin Pour**
11: $L_{i,i} \leftarrow 1$
12: **Pour** $j \leftarrow i+1$ à n **faire**
13: $L_{j,i} \leftarrow \frac{1}{U_{i,i}} \left(A_{j,i} - \sum_{k=1}^{i-1} L_{j,k} U_{k,i} \right)$
14: **Fin Pour**
15: **Fin Pour**

Algorithme 9 $\mathcal{R}_3$

1: **Pour** $i \leftarrow 1$ à n **faire**
2: **Pour** $j \leftarrow 1$ à $i-1$ **faire**
3: $U(i,j) \leftarrow 0$
4: **Fin Pour**
5: **Pour** $j \leftarrow i$ à n **faire**
6: $S_1 \leftarrow \sum_{k=1}^{i-1} L_{i,k} U_{k,j}$
7: $U_{i,j} \leftarrow A_{i,j} - S_1$
8: **Fin Pour**
9: **Pour** $j \leftarrow 1$ à $i-1$ **faire**
10: $L_{j,i} \leftarrow 0$
11: **Fin Pour**
12: $L_{i,i} \leftarrow 1$
13: **Pour** $j \leftarrow i+1$ à n **faire**
14: $S_2 \leftarrow \sum_{k=1}^{i-1} L_{j,k} U_{k,i}$
15: $L_{j,i} \leftarrow \frac{1}{U_{i,i}} (A_{j,i} - S_2)$.
16: **Fin Pour**
17: **Fin Pour**

Algorithme 9 $\mathcal{R}_3$

1: Pour $i \leftarrow 1$ à n faire
2: Pour $j \leftarrow 1$ à $i-1$ faire
3: $U(i,j) \leftarrow 0$
4: Fin Pour
5: Pour $j \leftarrow i$ à n faire
6: $S_1 \leftarrow \sum_{k=1}^{i-1} L_{i,k} U_{k,j}$
7: $U_{i,j} \leftarrow A_{i,j} - S_1$
8: Fin Pour
9: Pour $j \leftarrow 1$ à $i-1$ faire
10: $L_{j,i} \leftarrow 0$
11: Fin Pour
12: $L_{i,i} \leftarrow 1$
13: Pour $j \leftarrow i+1$ à n faire
14: $S_2 \leftarrow \sum_{k=1}^{i-1} L_{j,k} U_{k,i}$
15: $L_{j,i} \leftarrow \frac{1}{U_{i,i}} (A_{j,i} - S_2)$.
16: Fin Pour
17: Fin Pour

Algorithme 9 $\mathcal{R}_4$

1: Pour $i \leftarrow 1$ à n faire
2: Pour $j \leftarrow 1$ à $i-1$ faire
3: $U(i,j) \leftarrow 0$
4: Fin Pour
5: Pour $j \leftarrow i$ à n faire
6: $S_1 \leftarrow 0$
7: Pour $k \leftarrow 1$ à $i-1$ faire
8: $S_1 \leftarrow S_1 + L_{i,k} * U_{k,j}$
9: Fin Pour
10: $U_{i,j} \leftarrow A_{i,j} - S_1$
11: Fin Pour
12: Pour $j \leftarrow 1$ à $i-1$ faire
13: $L_{j,i} \leftarrow 0$
14: Fin Pour
15: $L_{i,i} \leftarrow 1$
16: Pour $j \leftarrow i+1$ à n faire
17: $S_2 \leftarrow 0$
18: Pour $k \leftarrow 1$ à $i-1$ faire
19: $S_2 \leftarrow S_2 + L_{j,k} * U_{k,i}$
20: Fin Pour
21: $L_{j,i} \leftarrow \frac{1}{U_{i,i}} (A_{j,i} - S_2)$.
22: Fin Pour
23: Fin Pour

Algorithme 9 Fonction FactLU permet de calculer les matrices $\mathbb{L}$ et $\mathbb{U}$ dites matrice de factorisation $\mathbb{LU}$ associée à la matrice $\mathbb{A}$, telle que

$$\mathbb{A} = \mathbb{LU}$$

Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{K})$ dont les sous-matrices principales sont inversibles.

Résultat : $\mathbb{L}$: matrice de $\mathcal{M}_n(\mathbb{K})$ triangulaire inférieure avec $L_{i,i} = 1,\ \forall i \in [\![1, n]\!]$

$\mathbb{U}$: matrice de $\mathcal{M}_n(\mathbb{K})$ triangulaire supérieure.

1: **Fonction** $[\mathbb{L}, \mathbb{U}] \leftarrow$ FactLU ($\mathbb{A}$)
2: $\quad \mathbb{U} \leftarrow \mathbb{O}_n$ ▷ $\mathbb{O}_n$ matrice nulle $n \times n$
3: $\quad \mathbb{L} \leftarrow \mathbb{I}_n$ ▷ $\mathbb{I}_n$ matrice identitée $n \times n$
4: $\quad$ **Pour** $i \leftarrow 1$ **à** n **faire**
5: $\quad\quad$ **Pour** $j \leftarrow i$ **à** n **faire** ▷ Calcul de la ligne i de $\mathbb{U}$
6: $\quad\quad\quad S_1 \leftarrow 0$
7: $\quad\quad\quad$ **Pour** $k \leftarrow 1$ **à** $i-1$ **faire**
8: $\quad\quad\quad\quad S_1 \leftarrow S_1 + L(i,k) * U(k,j)$
9: $\quad\quad\quad$ **Fin Pour**
10: $\quad\quad\quad U(i,j) \leftarrow A(i,j) - S_1$
11: $\quad\quad$ **Fin Pour**
12: $\quad\quad$ **Pour** $j \leftarrow i+1$ **à** n **faire** ▷ Calcul de la colonne i de $\mathbb{L}$
13: $\quad\quad\quad S_2 \leftarrow 0$
14: $\quad\quad\quad$ **Pour** $k \leftarrow 1$ **à** $i-1$ **faire**
15: $\quad\quad\quad\quad S_2 \leftarrow S_2 + L(j,k) * U(k,i)$
16: $\quad\quad\quad$ **Fin Pour**
17: $\quad\quad\quad L(j,i) \leftarrow (A_{j,i} - S_2)\,/U(i,i).$
18: $\quad\quad$ **Fin Pour**
19: $\quad$ **Fin Pour**
20: **Fin Fonction**

Plan

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ **hermitienne inversible** admettant une factorisation $\mathbb{LU}$. On pose

$$\mathbb{D} = \operatorname{diag} \mathbb{U} \text{ et } \mathbb{R} = \mathbb{D}^{-1}\mathbb{U}.$$

$\mathbb{R}$ est alors triangulaire supérieure à diagonale unité. On a alors

$$\mathbb{A} = \mathbb{LU} = \mathbb{LDD}^{-1}\mathbb{U} = \mathbb{LDR}.$$

$$\mathbb{A} \text{ hermitienne } \mathbb{A}^* = \mathbb{A} \implies \mathbb{A} = \mathbb{R}^*(\mathbb{D}^*\mathbb{L}^*) = \mathbb{L}(\mathbb{DR})$$

Par unicité de la factorisation $\mathbb{LU}$:

$$\mathbb{R}^* = \mathbb{L} \text{ et} \mathbb{D}^*\mathbb{L}^* = \mathbb{DR} \implies \mathbb{R}^* = \mathbb{L} \text{ et } \mathbb{D}^* = \mathbb{D}$$

Théorème 6: Factorisation $\mathbb{L}\mathbb{D}\mathbb{L}^*$

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice hermitienne inversible admettant une factorisation $\mathbb{L}\mathbb{U}$. Alors $\mathbb{A}$ s'écrit sous la forme

$$\mathbb{A} = \mathbb{L}\mathbb{D}\mathbb{L}^* \tag{9}$$

où $\mathbb{D} = \operatorname{diag} \mathbb{U}$ est une matrice à coefficients réels.

Corollaire 6.1:

Une matrice $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ admet une factorisation $\mathbb{L}\mathbb{D}\mathbb{L}^*$ avec $\mathbb{L} \in \mathcal{M}_n(\mathbb{C})$ matrice triangulaire inférieure à diagonale unité et $\mathbb{D} \in \mathcal{M}_n(\mathbb{R})$ matrice diagonale à coeffcients diagonaux strictement positifs **si et seulement si** la matrice $\mathbb{A}$ est hermitienne définie positive.

Plan

Definition

Une **factorisation régulière de Cholesky** d'une matrice $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ est une factorisation $\mathbb{A} = \mathbb{B}\mathbb{B}^*$ où $\mathbb{B}$ est une matrice triangulaire inférieure inversible.
Si les coefficients diagonaux de $\mathbb{B}$ sont positifs, on parle alors d'une **factorisation positive de Cholesky**.

Théorème: Factorisation de Cholesky

Une **norme** sur un espace vectoriel V est une application $\|\bullet\| : V \to \mathbb{R}^+$ qui vérifie les propriétés suivantes

- $\|\boldsymbol{v}\| = 0 \iff \boldsymbol{v} = 0$,
- $\|\alpha \boldsymbol{v}\| = |\alpha| \, \|\boldsymbol{v}\|, \ \forall \alpha \in \mathbb{K}, \ \forall \boldsymbol{v} \in V$,
- $\|\boldsymbol{u} + \boldsymbol{v}\| \leqslant \|\boldsymbol{u}\| + \|\boldsymbol{v}\|, \ \forall (\boldsymbol{u}, \boldsymbol{v}) \in V^2$ (inégalité triangulaire).

Une norme sur V est également appelée **norme vectorielle** . On appelle **espace vectoriel normé** un espace vectoriel muni d'une norme.

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ hermitienne définie positive et $\boldsymbol{b} \in \mathbb{C}^n$. On note $\mathbb{B}$ la matrice de factorisation positive de Cholesky de $\mathbb{A}$.

Trouver $\boldsymbol{x} \in \mathbb{C}^n$ tel que

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \quad (\Longleftrightarrow \quad \mathbb{B}\mathbb{B}^*\boldsymbol{x} = \boldsymbol{b}) \tag{10}$$

est équivalent à

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ hermitienne définie positive et $\boldsymbol{b} \in \mathbb{C}^n$. On note $\mathbb{B}$ la matrice de factorisation positive de Cholesky de $\mathbb{A}$.

Trouver $\boldsymbol{x} \in \mathbb{C}^n$ tel que

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \quad (\Longleftrightarrow \quad \mathbb{B}\mathbb{B}^*\boldsymbol{x} = \boldsymbol{b}) \tag{10}$$

est équivalent à

Trouver $\boldsymbol{x} \in \mathbb{C}^n$ solution de

$$\mathbb{B}^*\boldsymbol{x} = \boldsymbol{y} \tag{11}$$

avec $\boldsymbol{y} \in \mathbb{C}^n$ solution de

$$\mathbb{B}\boldsymbol{y} = \boldsymbol{b}. \tag{12}$$

Algorithme de résolution de systèmes linéaire par Cholesky

Algorithme 10 Fonction RSLCholesky permettant de résoudre, par une factorisation de Cholesky positive, le système linéaire

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$$

où $\mathbb{A}$ une matrice hermitienne de $\mathcal{M}_n(\mathbb{C})$ définie positive et $\boldsymbol{b} \in \mathbb{C}^n$.

Données : $\mathbb{A}$: matrice de $\mathcal{M}_n(\mathbb{C})$ symétrique définie positive,
$\boldsymbol{b}$: vecteur de $\mathbb{C}^n$.

Résultat : $\boldsymbol{x}$: vecteur de $\mathbb{C}^n$.

1: **Fonction** $\boldsymbol{x} \leftarrow$ RSLCholesky ($\mathbb{A}, \boldsymbol{b}$)
2: $\mathbb{B} \leftarrow$ Cholesky($\mathbb{A}$) ▷ Factorisation positive de Cholesky
3: $\boldsymbol{y} \leftarrow$ RSLTriInf($\mathbb{B}, \boldsymbol{b}$) ▷ Résolution du système $\mathbb{B}\boldsymbol{y} = \boldsymbol{b}$
4: $\mathbb{U} \leftarrow$ MatAdjointe($\mathbb{B}$) ▷ Calcul de la matrice adjointe de $\mathbb{B}$
5: $\boldsymbol{x} \leftarrow$ RSLTriSup($\mathbb{U}, \boldsymbol{y}$) ▷ Résolution du système $\mathbb{B}^*\boldsymbol{x} = \boldsymbol{y}$
6: **Fin Fonction**

Il nous faut donc écrire la fonction RSLCholesky

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ hermitienne définie positive. il existe une unique matrice $\mathbb{B} \in \mathcal{M}_n(\mathbb{C})$ triangulaire inférieure avec $\mathrm{B}_{i,i} \in \mathbb{R}^{+*}$, $\forall i \in [\![1, n]\!]$, telle que

$$\mathbb{A} = \mathbb{B}\mathbb{B}^*$$

c'est à dire

$$\begin{pmatrix} \mathrm{A}_{1,1} & \cdots & \mathrm{A}_{1,n} \\ \vdots & \ddots & \cdots \\ \mathrm{A}_{n,1} & \cdots & \mathrm{A}_{n,n} \end{pmatrix} = \begin{pmatrix} \mathrm{B}_{1,1} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ \mathrm{B}_{n,1} & \cdots & \cdots & \mathrm{B}_{n,n} \end{pmatrix} \begin{pmatrix} \overline{\mathrm{B}_{1,1}} & \cdots & \cdots & \overline{\mathrm{B}_{n,1}} \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \overline{\mathrm{B}_{n,n}} \end{pmatrix}.$$

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ hermitienne définie positive. il existe une unique matrice $\mathbb{B} \in \mathcal{M}_n(\mathbb{C})$ triangulaire inférieure avec $\mathrm{B}_{i,i} \in \mathbb{R}^{+*}$, $\forall i \in [\![1, n]\!]$, telle que

$$\mathbb{A} = \mathbb{B}\mathbb{B}^*$$

c'est à dire

$$\begin{pmatrix} \mathrm{A}_{1,1} & \cdots & \mathrm{A}_{1,n} \\ \vdots & \ddots & \cdots \\ \mathrm{A}_{n,1} & \cdots & \mathrm{A}_{n,n} \end{pmatrix} = \begin{pmatrix} \mathrm{B}_{1,1} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ \mathrm{B}_{n,1} & \cdots & \cdots & \mathrm{B}_{n,n} \end{pmatrix} \begin{pmatrix} \overline{\mathrm{B}_{1,1}} & \cdots & \cdots & \overline{\mathrm{B}_{n,1}} \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \overline{\mathrm{B}_{n,n}} \end{pmatrix}.$$

- Calcul de $\mathrm{B}_{1,1}$ (la 1ère ligne de $\mathbb{B}$ est donc déterminée)
  $\Longrightarrow$ calcul 1ère colonne de $\mathbb{B}$.

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ hermitienne définie positive. il existe une unique matrice $\mathbb{B} \in \mathcal{M}_n(\mathbb{C})$ triangulaire inférieure avec $\mathrm{B}_{i,i} \in \mathbb{R}^{+*}$, $\forall i \in [\![1, n]\!]$, telle que

$$\mathbb{A} = \mathbb{B}\mathbb{B}^*$$

c'est à dire

$$\begin{pmatrix} \mathrm{A}_{1,1} & \cdots & \mathrm{A}_{1,n} \\ \vdots & \ddots & \cdots \\ \mathrm{A}_{n,1} & \cdots & \mathrm{A}_{n,n} \end{pmatrix} = \begin{pmatrix} \mathrm{B}_{1,1} & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ \mathrm{B}_{n,1} & \cdots & \cdots & \mathrm{B}_{n,n} \end{pmatrix} \begin{pmatrix} \overline{\mathrm{B}_{1,1}} & \cdots & \cdots & \overline{\mathrm{B}_{n,1}} \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \overline{\mathrm{B}_{n,n}} \end{pmatrix}.$$

- Calcul de $\mathrm{B}_{1,1}$ (la 1ère ligne de $\mathbb{B}$ est donc déterminée)
  $\Longrightarrow$ calcul 1ère colonne de $\mathbb{B}$.
- Puis calcul de $\mathrm{B}_{2,2}$ (la 2ème ligne de $\mathbb{B}$ est donc déterminée)
  $\Longrightarrow$ calcul 2ème colonne de $\mathbb{B}$.
- Etc...

Soit $i \in [\![1, n]\!]$. On suppose connues les $i-1$ premières colonnes de $\mathbb{B}$.

Peut-on calculer la colonne i de $\mathbb{B}$?

$$\mathbb{A} = \mathbb{B}\mathbb{B}^* \implies \mathrm{A}_{i,i} = \sum_{k=1}^{n} \mathrm{B}_{i,k}(\mathbb{b}^*)_{k,i} = \sum_{k=1}^{n} \mathrm{B}_{i,k}\overline{\mathrm{B}_{i,k}}$$

Or $\mathbb{B}$ triangulaire inférieure (i.e. $\mathrm{B}_{i,j} = 0$ si $j > i$)

$$\mathrm{A}_{i,i} = \sum_{k=1}^{i-1} |\mathrm{B}_{i,k}|^2 + |\mathrm{B}_{i,i}|^2$$

et donc

$$\mathrm{B}_{i,i} = \left(\mathrm{A}_{i,i} - \sum_{j=1}^{i-1} |\mathrm{B}_{i,j}|^2 \right)^{1/2}.$$

Il reste à déterminer $\mathrm{B}_{j,i},\ \forall j \in [\![i+1, n]\!]$.

$$\mathrm{A}_{j,i} = \sum_{k=1}^{n} \mathrm{B}_{j,k} (\mathbb{B}^*)_{k,i} = \sum_{k=1}^{n} \mathrm{B}_{j,k} \overline{\mathrm{B}_{i,k}},\ \forall j \in [\![i+1, n]\!]$$

Comme $\mathbb{L}$ est triangulaire inférieure on obtient

$$\mathrm{A}_{j,i} = \sum_{k=1}^{i} \mathrm{B}_{j,k} \overline{\mathrm{B}_{i,k}} = \sum_{k=1}^{i-1} \mathrm{B}_{j,k} \overline{\mathrm{B}_{i,k}} + \mathrm{B}_{j,i} \overline{\mathrm{B}_{i,i}},\ \forall j \in [\![i+1, n]\!]$$

Or $\mathrm{B}_{i,i} > 0$ connu et les $i-1$ premières colonnes de $\mathbb{B}$ aussi.

$$\begin{aligned} \mathrm{B}_{j,i} &= \frac{1}{\mathrm{B}_{i,i}} \left(\mathrm{A}_{j,i} - \sum_{k=1}^{i-1} \mathrm{B}_{j,k} \overline{\mathrm{B}_{i,k}} \right),\ \forall j \in [\![i+1, n]\!] \\ \mathrm{B}_{j,i} &= 0,\ \forall j \in [\![1, i-1]\!]. \end{aligned}$$

Algorithme 11 $\mathcal{R}_0$

1: Calculer la matrice $\mathbb{B}$

Algorithme 11 $\mathcal{R}_1$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: Calculer $\mathrm{B}_{i,i}$, connaissant les $i-1$ premières colonnes de $\mathbb{B}$.
3: Calculer la $i^{\text{ème}}$ colonne de $\mathbb{B}$.
4: **Fin Pour**

Algorithme 11 $\mathcal{R}_1$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: Calculer $\mathrm{B}_{i,i}$, connaissant les $i-1$ premières colonnes de $\mathbb{B}$.
3: Calculer la $i^{\text{ème}}$ colonne de $\mathbb{B}$.
4: **Fin Pour**

Algorithme 11 $\mathcal{R}_2$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $\mathrm{B}_{i,i} \leftarrow \left(\mathrm{A}_{i,i} - \sum_{j=1}^{i-1} |\mathrm{B}_{i,j}|^2 \right)^{1/2}$
3: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
4: $\mathrm{B}_{j,i} \leftarrow 0$
5: **Fin Pour**
6: **Pour** $j \leftarrow i+1$ **à** n **faire**
7: $\mathrm{B}_{j,i} \leftarrow \frac{1}{\mathrm{B}_{i,i}} \left(\mathrm{A}_{j,i} - \sum_{k=1}^{i-1} \mathrm{B}_{j,k} \overline{\mathrm{B}_{i,k}} \right)$.
8: **Fin Pour**
9: **Fin Pour**

Algorithme 11 $\boxed{\mathcal{R}_2}$

1: **Pour** $i \leftarrow 1$ **à** n **faire**

2: $\mathrm{B}_{i,i} \leftarrow \left(\mathrm{A}_{i,i} - \sum_{j=1}^{i-1} |\mathrm{B}_{i,j}|^2 \right)^{1/2}$

3: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**

4: $\mathrm{B}_{j,i} \leftarrow 0$

5: **Fin Pour**

6: **Pour** $j \leftarrow i+1$ **à** n **faire**

7: $\mathrm{B}_{j,i} \leftarrow \frac{1}{\mathrm{B}_{i,i}} \left(\mathrm{A}_{j,i} - \sum_{k=1}^{i-1} \mathrm{B}_{j,k} \overline{\mathrm{B}_{i,k}} \right).$

8: **Fin Pour**

9: **Fin Pour**

Algorithme 11 $\boxed{\mathcal{R}_3}$

1: **Pour** $i \leftarrow 1$ **à** n **faire**

2: $S_1 \leftarrow \sum_{j=1}^{i-1} |\mathrm{B}_{i,j}|^2$

3: $\mathrm{B}_{i,i} \leftarrow (\mathrm{A}_{i,i} - S_1)^{1/2}$

4: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**

5: $\mathrm{B}_{j,i} \leftarrow 0$

6: **Fin Pour**

7: **Pour** $j \leftarrow i+1$ **à** n **faire**

8: $S_2 \leftarrow \sum_{k=1}^{i-1} \mathrm{B}_{j,k} \overline{\mathrm{B}_{i,k}}$

9: $\mathrm{B}_{j,i} \leftarrow \frac{1}{\mathrm{B}_{i,i}} (\mathrm{A}_{j,i} - S_2).$

10: **Fin Pour**

11: **Fin Pour**

Algorithme 11 $\mathcal{R}_3$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $\quad S_1 \leftarrow \sum_{j=1}^{i-1} |\mathrm{B}_{i,j}|^2$
3: $\quad \mathrm{B}_{i,i} \leftarrow (\mathrm{A}_{i,i} - S_1)^{1/2}$
4: $\quad$ **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
5: $\quad\quad \mathrm{B}_{j,i} \leftarrow 0$
6: $\quad$ **Fin Pour**
7: $\quad$ **Pour** $j \leftarrow i+1$ **à** n **faire**
8: $\quad\quad S_2 \leftarrow \sum_{k=1}^{i-1} \mathrm{B}_{j,k}\overline{\mathrm{B}_{i,k}}$
9: $\quad\quad \mathrm{B}_{j,i} \leftarrow \frac{1}{\mathrm{B}_{i,i}} (\mathrm{A}_{j,i} - S_2)$.
10: $\quad$ **Fin Pour**
11: **Fin Pour**

Algorithme 11 $\mathcal{R}_4$

1: **Pour** $i \leftarrow 1$ **à** n **faire**
2: $\quad S_1 \leftarrow 0$
3: $\quad$ **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
4: $\quad\quad S_1 \leftarrow S_1 + |\mathrm{B}_{i,j}|^2$
5: $\quad$ **Fin Pour**
6: $\quad \mathrm{B}_{i,i} \leftarrow (\mathrm{A}_{i,i} - S_1)^{1/2}$
7: $\quad$ **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
8: $\quad\quad \mathrm{B}_{j,i} \leftarrow 0$
9: $\quad$ **Fin Pour**
10: $\quad$ **Pour** $j \leftarrow i+1$ **à** n **faire**
11: $\quad\quad S_2 \leftarrow 0$
12: $\quad\quad$ **Pour** $k \leftarrow 1$ **à** $i-1$ **faire**
13: $\quad\quad\quad S_2 \leftarrow S_2 + \mathrm{B}_{j,k}\overline{\mathrm{B}_{i,k}}$
14: $\quad\quad$ **Fin Pour**
15: $\quad\quad \mathrm{B}_{j,i} \leftarrow \frac{1}{\mathrm{B}_{i,i}} (\mathrm{A}_{j,i} - S_2)$.
16: $\quad$ **Fin Pour**
17: **Fin Pour**

Algorithme 11 Fonction CHOLESKY permettant de calculer la matrice $\mathbb{B}$, dites matrice de factorisation positive de Cholesky associée à la matrice $\mathbb{A}$, telle que

$$\mathbb{A} = \mathbb{B}\mathbb{B}^*.$$

Données : A : matrice de $\mathcal{M}_n(\mathbb{C})$ hermitienne définie positive.
Résultat : B : matrice de $\mathcal{M}_n(\mathbb{C})$ triangulaire inférieure avec $\mathrm{B}(i,i) > 0,\ \forall i \in [\![1, n]\!]$

1: **Fonction** $\mathrm{B} \leftarrow$ CHOLESKY (A)
2: **Pour** $i \leftarrow 1$ **à** n **faire**
3: $S_1 \leftarrow 0$
4: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
5: $S_1 \leftarrow S_1 + |\mathrm{B}(i,j)|^2$
6: **Fin Pour**
7: $\mathrm{B}(i,i) \leftarrow \mathrm{SQRT}(\mathrm{A}(i,i) - S_1)$
8: **Pour** $j \leftarrow 1$ **à** $i-1$ **faire**
9: $\mathrm{B}(j,i) \leftarrow 0$
10: **Fin Pour**
11: **Pour** $j \leftarrow i+1$ **à** n **faire**
12: $S_2 \leftarrow 0$
13: **Pour** $k \leftarrow 1$ **à** $i-1$ **faire**
14: $S_2 \leftarrow S_2 + \mathrm{B}(j,k) * \overline{\mathrm{B}(i,k)}$
15: **Fin Pour**
16: $\mathrm{B}(j,i) \leftarrow (\mathrm{A}(j,i) - S_2)/\mathrm{B}(i,i).$
17: **Fin Pour**
18: **Fin Pour**
19: **Fin Fonction**

Exercice

Proposer une méthode permettant de tester la fonction CHOLESKY .

Plan

Definition: Matrice élémentaire de Householder

Soit $\boldsymbol{u} \in \mathbb{C}^n$ tel que $\|\boldsymbol{u}\|_2 = 1$. On appelle **matrice élémentaire de Householder** la matrice $\mathbb{H}(\boldsymbol{u}) \in \mathcal{M}_n(\mathbb{C})$ définie par

$$\mathbb{H}(\boldsymbol{u}) = \mathbb{I} - 2\boldsymbol{u}\boldsymbol{u}^*. \tag{13}$$

Propriété:

Toute matrice élémentaire de Householder est hermitienne et unitaire.

Propriété:

Soient $\boldsymbol{x} \in \mathbb{K}^n$ et $\boldsymbol{u} \in \mathbb{K}^n$, $\|\boldsymbol{u}\|_2 = 1$. On note $\boldsymbol{x}_{\|} = \mathrm{proj}_{\boldsymbol{u}}(\boldsymbol{x}) \overset{\text{def}}{=} \langle \boldsymbol{u}, \boldsymbol{x} \rangle \boldsymbol{u}$ et $\boldsymbol{x}_{\perp} = \boldsymbol{x} - \boldsymbol{x}_{\|}$. On a alors

$$\mathbb{H}(\boldsymbol{u})(\boldsymbol{x}_{\perp} + \boldsymbol{x}_{\|}) = \boldsymbol{x}_{\perp} - \boldsymbol{x}_{\|}. \tag{14}$$

et

$$\mathbb{H}(\boldsymbol{u})\boldsymbol{x} = \boldsymbol{x}, \quad \text{si } \langle \boldsymbol{x}, \boldsymbol{u} \rangle = 0. \tag{15}$$

Théorème:

Soient $\boldsymbol{a}$, $\boldsymbol{b}$ deux vecteurs non colinéaires de $\mathbb{C}^n$ avec $\|\boldsymbol{b}\|_2 = 1$. Soit $\alpha \in \mathbb{C}$ tel que $|\alpha| = \|\boldsymbol{a}\|_2$ et $\arg \alpha = -\arg \langle \boldsymbol{a}, \boldsymbol{b} \rangle \ [\pi]$. On a alors

$$\mathbb{H}\left(\frac{\boldsymbol{a} - \alpha \boldsymbol{b}}{\|\boldsymbol{a} - \alpha \boldsymbol{b}\|_2}\right) \boldsymbol{a} = \alpha \boldsymbol{b}. \tag{16}$$

Exercice:

Soient $\boldsymbol{a}$ et $\boldsymbol{b}$ deux vecteurs non nuls et non colinéaires de $\mathbb{C}^n$ avec $\|\boldsymbol{b}\|_2 = 1$.

Q. 1

Ecrire la fonction algorithmique HOUSEHOLDER *permettant de retourner une matrice de Householder* $\mathbb{H}$ *et* $\alpha \in \mathbb{C}$ *tels que* $\mathbb{H}(\boldsymbol{u})\boldsymbol{a} = \alpha\boldsymbol{b}$. *Le choix du* α *est fait par le paramètre* δ *(*0 *ou* 1*) de telle sorte que* $\arg\alpha = -\arg(\langle \boldsymbol{a}, \boldsymbol{b}\rangle) + \delta\pi$ *avec* $|\alpha| = \|\boldsymbol{a}\|_2$.
Des fonctions comme DOT$(\boldsymbol{a}, \boldsymbol{b})$ *(produit scalaire de deux vecteurs),* NORM$(\boldsymbol{a})$ *(norme d'un vecteur),* ARG(z) *(argument d'un nombre complexe),* MATPROD$(\mathbb{A}, \mathbb{B})$ *(produit de deux matrices),* CTRANSPOSE$(\mathbb{A})$ *(adjoint d'une matrice), ... pourront être utilisées*

Q. 2

Proposer un programme permettant de tester cette fonction. On pourra utiliser la fonction VECRAND(n) *retournant un vecteur aléatoire de* $\mathbb{C}^n$ *, les parties réelles et imaginaires de chacune de ses*

Corollaire 6.2:

Soit $\boldsymbol{a} \in \mathbb{C}^n$ avec $a_1 \neq 0$ et $\exists j \in [\![2, n]\!]$ tel que $a_j \neq 0$. Soient $\theta = \arg a_1$ et

$$\boldsymbol{u}_{\pm} = \frac{\boldsymbol{a} \pm \|\boldsymbol{a}\|_2 \, e^{\imath\theta} \boldsymbol{e}_1}{\|\boldsymbol{a} \pm \|\boldsymbol{a}\|_2 \, e^{\imath\theta} \boldsymbol{e}_1\|}$$

Alors

$$\mathbb{H}(\boldsymbol{u}_{\pm})\boldsymbol{a} = \mp \|\boldsymbol{a}\|_2 \, e^{\imath\theta} \boldsymbol{e}_1 \tag{17}$$

où $\boldsymbol{e}_1$ désigne le premier vecteur de la base canonique de $\mathbb{C}^n$.

Théorème 7: 

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$ une matrice. Il existe une matrice unitaire $\mathbb{Q} \in \mathcal{M}_n(\mathbb{C})$ produit d'au plus $n-1$ matrices de Householder et une matrice triangulaire supérieure $\mathbb{R} \in \mathcal{M}_n(\mathbb{C})$ telles que

$$\mathbb{A} = \mathbb{Q}\mathbb{R}. \tag{18}$$

Si $\mathbb{A}$ est réelle alors $\mathbb{Q}$ et $\mathbb{R}$ sont aussi réelles et l'on peut choisir $\mathbb{Q}$ de telle sorte que les coefficients diagonaux de $\mathbb{R}$ soient positifs. De plus, si $\mathbb{A}$ est inversible alors la factorisation est unique.

Exercice: Algorithmique

Q. 1

Ecrire une fonction FactQR *permettant de calculer la factorisation* $\mathbb{QR}$ *d'une matrice* $\mathbb{A} \in \mathcal{M}_n(\mathbb{C})$.
On pourra utiliser la fonction Householder *(voir Exercice 49, page 75).*

Q. 2

Ecrire un programme permettant de tester cette fonction.

Plan

Definition

Une **norme** sur un espace vectoriel V est une application $\|\bullet\| : V \to \mathbb{R}^+$ qui vérifie les propriétés suivantes

- $\|\boldsymbol{v}\| = 0 \iff \boldsymbol{v} = 0$,
- $\|\alpha \boldsymbol{v}\| = |\alpha| \, \|\boldsymbol{v}\|, \ \forall \alpha \in \mathbb{K}, \ \forall \boldsymbol{v} \in V$,
- $\|\boldsymbol{u} + \boldsymbol{v}\| \leqslant \|\boldsymbol{u}\| + \|\boldsymbol{v}\|, \ \forall (\boldsymbol{u}, \boldsymbol{v}) \in V^2$ (inégalité triangulaire).

Une norme sur V est également appelée **norme vectorielle** . On appelle **espace vectoriel normé** un espace vectoriel muni d'une norme.

Proposition

Soit $\boldsymbol{v} \in \mathbb{K}^n$. Pour tout nombre réel $p \geqslant 1$, l'application $\|\bullet\|_p$ définie par

$$\|\boldsymbol{v}\|_p = \left(\sum_{i=1}^{n} |v_i|^p \right)^{1/p}$$

est une norme sur $\mathbb{K}^n$.

Normes usitées :

$$\|\boldsymbol{v}\|_1 = \sum_{i=1}^{n} |v_i|, \quad \|\boldsymbol{v}\|_2 = \left(\sum_{i=1}^{n} |v_i|^2 \right)^{1/2}, \quad \|\boldsymbol{v}\|_\infty = \max_{i \in [\![1,n]\!]} |v_i|.$$

Lemme 8.1: Inégalité de Cauchy-Schwarz

$\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{K}^n$

$$|\langle \boldsymbol{x}, \boldsymbol{y} \rangle| \leqslant \|\boldsymbol{x}\|_2 \, \|\boldsymbol{y}\|_2 \, . \tag{19}$$

Cette inégalité s'appelle l'**inégalité de Cauchy-Schwarz**. On a égalité si et seulement si $\boldsymbol{x}$ et $\boldsymbol{y}$ sont colinéaires.

Lemme 8.2: Inégalité de Hölder

Pour $p > 1$ et $\frac{1}{p} + \frac{1}{q} = 1$, on a $\forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{K}^n$

$$\sum_{i=1}^{n} |u_i v_i| \leqslant \left(\sum_{i=1}^{n} |u_i|^p \right)^{1/p} \left(\sum_{i=1}^{n} |v_i|^q \right)^{1/q} = \|\boldsymbol{u}\|_p \|\boldsymbol{v}\|_q . \qquad (20)$$

Cette inégalité s'appelle l'**inégalité de Hölder**.

Definition 8.3

Deux **normes** $\|\bullet\|$ et $\|\bullet\|'$, définies sur un même espace vectoriel V, sont **équivalentes** s'il exite deux constantes C et C' telles que

$$\|\boldsymbol{v}\|' \leqslant C\,\|\boldsymbol{v}\| \quad \text{et} \quad \|\boldsymbol{v}\| \leqslant C'\,\|\boldsymbol{v}\|' \quad \text{pour tout } \boldsymbol{v} \in V. \tag{21}$$

Proposition

Sur un espace vectoriel de dimension finie toutes les normes sont équivalentes.

Plan

Definition 8.4

Une **norme matricielle** sur $\mathcal{M}_n(\mathbb{K})$ est une application $\|\bullet\| : \mathcal{M}_n(\mathbb{K}) \to \mathbb{R}^+$ vérifiant

1. $\|\mathbb{A}\| = 0 \iff \mathbb{A} = 0$,
2. $\|\alpha\mathbb{A}\| = |\alpha| \|\mathbb{A}\|$, $\forall \alpha \in \mathbb{K}$, $\forall \mathbb{A} \in \mathcal{M}_n(\mathbb{K})$,
3. $\|\mathbb{A} + \mathbb{B}\| \leqslant \|\mathbb{A}\| + \|\mathbb{B}\|$, $\forall (\mathbb{A}, \mathbb{B}) \in \mathcal{M}_n(\mathbb{K})^2$ (inégalité triangulaire)
4. $\|\mathbb{A}\mathbb{B}\| \leqslant \|\mathbb{A}\| \|\mathbb{B}\|$, $\forall (\mathbb{A}, \mathbb{B}) \in \mathcal{M}_n(\mathbb{K})^2$

Peut-on étendre cette définition sur $\mathcal{M}_{m,n}(\mathbb{K})$?

Proposition:

Etant donné une norme vectorielle $\|\bullet\|$ sur $\mathbb{C}^n$, l'application $\|\bullet\|_s : \mathcal{M}_n(\mathbb{C}) \to \mathbb{R}^+$ définie par

$$\|\mathbb{A}\|_s = \sup_{\substack{\boldsymbol{v}\in\mathbb{C}^n \\ \boldsymbol{v}\neq 0}} \frac{\|\mathbb{A}\boldsymbol{v}\|}{\|\boldsymbol{v}\|} = \sup_{\substack{\boldsymbol{v}\in\mathbb{C}^n \\ \|\boldsymbol{v}\|\leqslant 1}} \|\mathbb{A}\boldsymbol{v}\| = \sup_{\substack{\boldsymbol{v}\in\mathbb{C}^n \\ \|\boldsymbol{v}\|= 1}} \|\mathbb{A}\boldsymbol{v}\|, \tag{22}$$

est une norme matricielle, appelée **norme matricielle subordonnée** (à la norme vectorielle donnée).
De plus

$$\|\mathbb{A}\boldsymbol{v}\| \leqslant \|\mathbb{A}\|_s \|\boldsymbol{v}\| \quad \forall \boldsymbol{v} \in \mathbb{C}^n \tag{23}$$

et la norme $\|\mathbb{A}\|$ peut se définir aussi par

$$\|\mathbb{A}\|_s = \inf\{\alpha \in \mathbb{R} : \|\mathbb{A}\boldsymbol{v}\| \leqslant \alpha \|\boldsymbol{v}\|, \ \forall \boldsymbol{v} \in \mathbb{K}^n\}. \tag{24}$$

Il existe au moins un vecteur $\boldsymbol{u} \in \mathbb{C}^n$ tel que

$$\boldsymbol{u} \neq 0 \ \text{ et } \ \|\mathbb{A}\boldsymbol{u}\| = \|\mathbb{A}\|_s \|\boldsymbol{u}\|. \tag{25}$$

Enfin une norme subordonnée vérifie toujours

$$\|\mathbb{I}\|_s = 1 \tag{26}$$

Théorème 9

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$. On a

$$\|\mathbb{A}\|_1 \overset{\text{def}}{=} \sup_{\substack{\boldsymbol{v}\in\mathbb{K}^n \\ \boldsymbol{v}\neq 0}} \frac{\|\mathbb{A}\boldsymbol{v}\|_1}{\|\boldsymbol{v}\|_1} = \max_{j\in[\![1,n]\!]} \sum_{i=1}^{n} |a_{ij}| \tag{27}$$

$$\|\mathbb{A}\|_2 \overset{\text{def}}{=} \sup_{\substack{\boldsymbol{v}\in\mathbb{K}^n \\ \boldsymbol{v}\neq 0}} \frac{\|\mathbb{A}\boldsymbol{v}\|_2}{\|\boldsymbol{v}\|_2} = \sqrt{\rho\left(\mathbb{A}^*\mathbb{A}\right)} = \sqrt{\rho\left(\mathbb{A}\mathbb{A}^*\right)} = \|\mathbb{A}^*\|_2 \tag{28}$$

$$\|\mathbb{A}\|_\infty \overset{\text{def}}{=} \sup_{\substack{\boldsymbol{v}\in\mathbb{K}^n \\ \boldsymbol{v}\neq 0}} \frac{\|\mathbb{A}\boldsymbol{v}\|_\infty}{\|\boldsymbol{v}\|_\infty} = \max_{i\in[\![1,n]\!]} \sum_{j=1}^{n} |a_{ij}| \tag{29}$$

La norme $\|\bullet\|_2$ est invariante par transformation unitaire :

$$\mathbb{U}\mathbb{U}^* = \mathbb{I} \Longrightarrow \|\mathbb{A}\|_2 = \|\mathbb{A}\mathbb{U}\|_2 = \|\mathbb{U}\mathbb{A}\|_2 = \|\mathbb{U}^*\mathbb{A}\mathbb{U}\|_2 \,. \tag{30}$$

Corollaire 9.1

1. Si une matrice $\mathbb{A}$ est hermitienne, ou symétrique (donc normale), on a $\|\mathbb{A}\|_2 = \rho(\mathbb{A})$.
2. Si une matrice $\mathbb{A}$ est unitaire, ou orthogonale (donc normale), on a $\|\mathbb{A}\|_2 = 1$.

Plan

Definition 9.2

Soit V un espace vectoriel muni d'une norme $\|\bullet\|$, on dit qu'une suite $(\boldsymbol{v}_k)$ d'éléments de V **converge vers un élément** $\boldsymbol{v} \in V$, si

$$\lim_{k \to \infty} \|\boldsymbol{v}_k - \boldsymbol{v}\| = 0$$

et on écrit

$$\boldsymbol{v} = \lim_{k \to \infty} \boldsymbol{v}_k.$$

Théorème 10: admis

Soit $\mathbb{B}$ une matrice carrée. Les conditions suivantes sont équivalentes :

1. $\lim_{k\to\infty} \mathbb{B}^k = 0$,
2. $\lim_{k\to\infty} \mathbb{B}^k \boldsymbol{v} = 0$ pour tout vecteur $\boldsymbol{v}$,
3. $\rho(\mathbb{B}) < 1$,
4. $\|\mathbb{B}\| < 1$ pour au moins une norme matricielle subordonnée $\|\bullet\|$.

Théorème 11: admis

Soit $\mathbb{B}$ une matrice carrée, et $\|\bullet\|$ une norme matricielle quelconque. Alors

$$\lim_{k\to\infty} \left\|\mathbb{B}^k\right\|^{1/k} = \rho(\mathbb{B}).$$

Soient $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ inversible et $\boldsymbol{b} \in \mathbb{K}^n$.

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$$

De petites erreurs sur les données engendrent-elles de petites erreurs sur la solution?

Exemple de R.S. Wilson

Soient

$$\mathbb{A} = \begin{pmatrix} 10 & 7 & 8 & 7 \\ 7 & 5 & 6 & 5 \\ 8 & 6 & 10 & 9 \\ 7 & 5 & 9 & 10 \end{pmatrix}, \quad \Delta\mathbb{A} = \begin{pmatrix} 0 & 0 & \frac{1}{10} & \frac{1}{5} \\ \frac{2}{25} & \frac{1}{25} & 0 & 0 \\ 0 & -\frac{1}{50} & -\frac{11}{100} & 0 \\ -\frac{1}{100} & -\frac{1}{100} & 0 & -\frac{1}{50} \end{pmatrix}$$

et $\boldsymbol{b}^{\mathrm{t}} = (32, 23, 33, 31)$, $(\boldsymbol{\Delta b})^{\mathrm{t}} = \left(\frac{1}{100}, -\frac{1}{100}, \frac{1}{100}, -\frac{1}{100}\right)$. Des calculs exacts donnent

$$\begin{aligned}
\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \quad &\Longleftrightarrow \quad \boldsymbol{x}^{\mathrm{t}} = (1, 1, 1, 1) \\
\mathbb{A}\boldsymbol{u} = (\boldsymbol{b} + \boldsymbol{\Delta b}) \quad &\Longleftrightarrow \quad \boldsymbol{u}^{\mathrm{t}} = \left(\frac{91}{50}, -\frac{9}{25}, \frac{27}{20}, \frac{79}{100}\right) \\
&\qquad\quad \approx (1.8, -0.36, 1.3, 0.79) \\
(\mathbb{A} + \Delta\mathbb{A})\boldsymbol{v} = \boldsymbol{b} \quad &\Longleftrightarrow \quad \boldsymbol{v}^{\mathrm{t}} = (-81, 137, -34, 22) \\
(\mathbb{A} + \Delta\mathbb{A})\boldsymbol{y} = (\boldsymbol{b} + \boldsymbol{\Delta b}) \quad &\Longleftrightarrow \quad \boldsymbol{y}^{\mathrm{t}} = \left(-\frac{18283543}{461600}, \frac{31504261}{461600}, -\frac{3741501}{230800}, \frac{5235241}{461600}\right) \\
&\qquad\quad \approx (-39.61, 68.25, -16.21, 11.34)
\end{aligned}$$

Soient $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ inversible et $\boldsymbol{b} \in \mathbb{K}^n$.

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$$

De petites erreurs sur les données engendrent-elles de petites erreurs sur la solution?

Non, pas forcément!

Le système linéaire prédédent est **mal conditionné**.
On dit qu'un système linéaire est **bien conditionné** ou qu'il a un **bon conditionnement** si de petites perturbations des données n'entrainent qu'une variation *raisonnable* de la solution.

Est-il possible de "mesurer" le **conditionnement** d'une matrice?

Definition 12.1

Soit $\|.\|$ une norme matricielle subordonnée, le conditionnement d'une matrice régulière $\mathbb{A}$, associé à cette norme, est le nombre

$$\mathrm{cond}(\mathbb{A}) = \|\mathbb{A}\| \left\|\mathbb{A}^{-1}\right\|.$$

Nous noterons $\mathrm{cond}_p(\mathbb{A}) = \|\mathbb{A}\|_p \left\|\mathbb{A}^{-1}\right\|_p$.

Proposition: 

Soit $\mathbb{A}$ une matrice régulière. On a les propriétés suivantes

1. $\forall \alpha \in \mathbb{K}^*, \operatorname{cond}(\alpha\mathbb{A}) = \operatorname{cond}(\mathbb{A})$.
2. $\operatorname{cond}_p(\mathbb{A}) \geqslant 1, \forall p \in [\![1, +\infty]\!]$.
3. $\operatorname{cond}_2(\mathbb{A}) = 1$ si et seulement si $\mathbb{A} = \alpha\mathbb{Q}$ avec $\alpha \in \mathbb{K}^*$ et $\mathbb{Q}$ matrice unitaire

Théorème 13:

Soit $\mathbb{A}$ une matrice inversible. Soient $\boldsymbol{x}$ et $\boldsymbol{x} + \boldsymbol{\Delta x}$ les solutions respectives de

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \quad \text{et} \quad \mathbb{A}\left(\boldsymbol{x} + \boldsymbol{\Delta x}\right) = \boldsymbol{b} + \boldsymbol{\Delta b}.$$

Supposons $\boldsymbol{b} \neq \boldsymbol{0}$, alors l'inégalité

$$\frac{\|\boldsymbol{\Delta x}\|}{\|\boldsymbol{x}\|} \leqslant \operatorname{cond}(\mathbb{A}) \frac{\|\boldsymbol{\Delta b}\|}{\|\boldsymbol{b}\|}$$

est satisfaite, et c'est la meilleure possible : pour une matrice $\mathbb{A}$ donnée, on peut trouver des vecteurs $\boldsymbol{b} \neq \boldsymbol{0}$ et $\boldsymbol{\Delta b} \neq \boldsymbol{0}$ tels qu'elle devienne une égalité.

Théorème: 

Soient $\mathbb{A}$ et $\mathbb{A} + \Delta\mathbb{A}$ deux matrices inversibles. Soient $\boldsymbol{x}$ et $\boldsymbol{x} + \boldsymbol{\Delta x}$ les solutions respectives de

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \text{ et } (\mathbb{A} + \Delta\mathbb{A})(\boldsymbol{x} + \boldsymbol{\Delta x}) = \boldsymbol{b}.$$

Supposons $\boldsymbol{b} \neq \boldsymbol{0}$, alors on a

$$\frac{\|\boldsymbol{\Delta x}\|}{\|\boldsymbol{x} + \boldsymbol{\Delta x}\|} \leqslant \mathrm{cond}(\mathbb{A}) \frac{\|\Delta\mathbb{A}\|}{\|\mathbb{A}\|}.$$

Remarque 13.1

Une matrice est donc **bien conditionnée** si son conditionnement est proche de 1.

Plan

Méthodes itératives pour la résolution du système linéaire

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}.$$

Trouver une **matrice d'itération** $\mathbb{B}$ et d'un vecteur $\boldsymbol{c}$ telles que

$$\boldsymbol{x}^{[k+1]} = \mathbb{B}\boldsymbol{x}^{[k]} + \boldsymbol{c},\ k \geqslant 0,\ \boldsymbol{x}^{[0]} \text{ arbitraire}$$

vérifie

$$\lim_{k \to \infty} \boldsymbol{x}^{[k]} = \tilde{\boldsymbol{x}} \quad \text{avec } \tilde{\boldsymbol{x}} = \mathbb{A}^{-1}\boldsymbol{b}$$

Plan

Soit $\mathbb{A} \in \mathcal{M}_n(\mathbb{K})$ une matrice régulière, avec $\forall i \in [\![1, n]\!],\ \mathrm{A}_{i,i} \neq 0$

$$\mathbb{A} = \begin{pmatrix} \mathrm{A}_{1,1} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \mathrm{A}_{n,n} \end{pmatrix} + \begin{pmatrix} 0 & \cdots & \cdots & 0 \\ \mathrm{A}_{2,1} & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \mathrm{A}_{n,1} & \cdots & \mathrm{A}_{n,n-1} & 0 \end{pmatrix} + \begin{pmatrix} 0 & \mathrm{A}_{1,2} & \cdots & \mathrm{A}_{1,n} \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \mathrm{A}_{n-1,n} \\ 0 & \cdots & \cdots & 0 \end{pmatrix}$$

$$= \mathbb{D} - \mathbb{E} - \mathbb{F} = \begin{pmatrix} \ddots & & -\mathbb{F} \\ & \mathbb{D} & \\ -\mathbb{E} & & \ddots \end{pmatrix}$$

Plan

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \forall i,\ b_i = \sum_{j=1}^{n} \mathrm{A}_{i,j} x_j = \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j + \mathrm{A}_{i,i} x_i + \sum_{j=i+1}^{n} \mathrm{A}_{i,j} x_j$$

La méthode itérative de **Jacobi** :

$$b_i = \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j^{[k]} + \mathrm{A}_{i,i} x_i^{[k+1]} + \sum_{j=i+1}^{n} \mathrm{A}_{i,j} x_j^{[k]} \quad \forall i \in [\![1, n]\!]$$

ou encore

$$x_i^{[k+1]} = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1, j\neq i}^{n} \mathrm{A}_{ij} x_j^{[k]} \right) \quad \forall i \in [\![1, n]\!]$$

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \forall i,\ b_i = \sum_{j=1}^{n} \mathrm{A}_{i,j} x_j = \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j + \mathrm{A}_{i,i} x_i + \sum_{j=i+1}^{n} \mathrm{A}_{i,j} x_j$$

La méthode itérative de **Jacobi** :

$$b_i = \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j^{[k]} + \mathrm{A}_{i,i} x_i^{[k+1]} + \sum_{j=i+1}^{n} \mathrm{A}_{i,j} x_j^{[k]} \quad \forall i \in [\![1, n]\!]$$

$$\boldsymbol{b} = \mathbb{D}\boldsymbol{x}^{[k+1]} - \mathbb{E}\boldsymbol{x}^{[k]} - \mathbb{F}\boldsymbol{x}^{[k]}$$

ou encore

$$x_i^{[k+1]} = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1, j\neq i}^{n} \mathrm{A}_{ij} x_j^{[k]} \right) \quad \forall i \in [\![1, n]\!]$$

$$\boldsymbol{x}^{[k+1]} = \mathbb{D}^{-1}(\mathbb{E} + \mathbb{F})\boldsymbol{x}^{[k]} + \mathbb{D}^{-1}\boldsymbol{b}$$

Plan

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \forall i,\ b_i = \sum_{j=1}^{n} \mathrm{A}_{i,j} x_j = \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j + \mathrm{A}_{i,i} x_i + \sum_{j=i+1}^{n} \mathrm{A}_{i,j} x_j$$

La méthode itérative de **Gauss-Seidel** :

$$b_i = \mathrm{A}_{i,i} x_i^{[k+1]} + \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j^{[k+1]} + \sum_{j=i+1}^{n} \mathrm{A}_{i,j} x_j^{[k]} \quad \forall i \in [\![1, n]\!]$$

ou encore

$$x_i^{(k+1)} = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1}^{i-1} \mathrm{A}_{ij} x_j^{[k+1]} - \sum_{j=i+1}^{n} \mathrm{A}_{ij} x_j^{[k]} \right) \quad \forall i \in [\![1, n]\!]$$

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b} \iff \forall i,\ b_i = \sum_{j=1}^{n} \mathrm{A}_{i,j} x_j = \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j + \mathrm{A}_{i,i} x_i + \sum_{j=i+1}^{n} \mathrm{A}_{i,j} x_j$$

La méthode itérative de **Gauss-Seidel** :

$$b_i = \mathrm{A}_{i,i} x_i^{[k+1]} + \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j^{[k+1]} + \sum_{j=i+1}^{n} \mathrm{A}_{i,j} x_j^{[k]} \quad \forall i \in [\![1, n]\!]$$

$$\boldsymbol{b} = \mathbb{D}\boldsymbol{x}^{[k+1]} - \mathbb{E}\boldsymbol{x}^{[k+1]} - \mathbb{F}\boldsymbol{x}^{[k]}$$

ou encore

$$x_i^{(k+1)} = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1}^{i-1} \mathrm{A}_{ij} x_j^{[k+1]} - \sum_{j=i+1}^{n} \mathrm{A}_{ij} x_j^{[k]} \right) \quad \forall i \in [\![1, n]\!]$$

$$\boldsymbol{x}^{[k+1]} = (\mathbb{D} - \mathbb{E})^{-1} \mathbb{F}\boldsymbol{x}^{[k]} + (\mathbb{D} - \mathbb{E})^{-1} \boldsymbol{b}$$

Plan

Soit $w \in \mathbb{R}^*$.

$$x_i^{[k+1]} = w\hat{x}_i^{[k+1]} + (1-w)x_i^{[k]}$$

où $\hat{x}_i^{[k+1]}$ est obtenu à partir de l'une des deux méthodes précédentes.
Avec la méthode de Gauss-Seidel : méthode S.O.R. (successive over relaxation)

$$x_i^{[k+1]} = \frac{w}{\mathrm{A}_{ii}}\left(b_i - \sum_{j=1}^{i-1} \mathrm{A}_{ij}x_j^{[k+1]} - \sum_{j=i+1}^{n} \mathrm{A}_{ij}x_j^{[k]}\right) + (1-w)x_i^{[k]} \quad \forall i \in [\![1, n]\!]$$

Exercice 14.1:

Déterminer la matrice d'itération $\mathbb{B}$ et le vecteur $\boldsymbol{c}$ tels que

$$\boldsymbol{x}^{[k+1]} = \mathbb{B}\boldsymbol{x}^{[k]} + \boldsymbol{c}$$

en fonction de $\mathbb{D}$, $\mathbb{E}$, $\mathbb{F}$, et $\boldsymbol{b}$.

Proposition:

On pose $\mathbb{L} = \mathbb{D}^{-1}\mathbb{E}$ et $\mathbb{U} = \mathbb{D}^{-1}\mathbb{F}$.
La matrice d'itération de la méthode de Jacobi, notée $\mathbb{J}$, est donnée par

$$\mathbb{J} = \mathbb{D}^{-1}(\mathbb{E} + \mathbb{F}) = \mathbb{L} + \mathbb{U}, \tag{31}$$

La matrice d'itération de la méthode S.O.R., notée $\mathcal{L}_w$, est donnée par

$$\mathcal{L}_w = \left(\frac{\mathbb{D}}{w} - \mathbb{E}\right)^{-1}\left(\frac{1-w}{w}\mathbb{D} + \mathbb{F}\right) = (\mathbb{I} - w\mathbb{L})^{-1}\left((1-w)\mathbb{I} + w\mathbb{U}\right). \tag{32}$$

et elle vérifie

$$\rho(\mathcal{L}_w) \geqslant |w-1|. \tag{33}$$

La matrice d'itération de Gauss-Seidel est $\mathcal{L}_1$ et elle correspond à

$$\mathcal{L}_1 = (\mathbb{D} - \mathbb{E})^{-1}\mathbb{F} = (\mathbb{I} - \mathbb{L})^{-1}\mathbb{U}. \tag{34}$$

Plan

 **Théorème:** 

Soit $\mathbb{A}$ une matrice régulière décomposée sous la forme $\mathbb{A} = \mathbb{M} - \mathbb{N}$ avec $\mathbb{M}$ régulière. On pose

$$\mathbb{B} = \mathbb{M}^{-1}\mathbb{N} \quad \text{et} \quad \boldsymbol{c} = \mathbb{M}^{-1}\boldsymbol{b}.$$

Alors la suite définie par

$$\boldsymbol{x}^{[0]} \in \mathbb{K}^n \quad \text{et} \quad \boldsymbol{x}^{[k+1]} = \mathbb{B}\boldsymbol{x}^{[k]} + \boldsymbol{c}$$

converge vers $\bar{\boldsymbol{x}} = \mathbb{A}^{-1}\boldsymbol{b}$ quelque soit $\boldsymbol{x}^{[0]}$ si et seulement si $\rho(\mathbb{B}) < 1$.

Corollaire: 

Soit $\mathbb{A}$ une matrice vérifiant $\mathrm{A}_{i,i} \neq 0 \ \forall i$. Une condition nécessaire de convergence pour la méthode S.O.R. est que $0 < w < 2$.

Théorème: voir Lascaux-Théodor, vol.2, Théorème 19 et 20, pages 346 à 349

Soit $\mathbb{A}$ une matrice à diagonale strictement dominante ou une matrice inversible à diagonale fortement dominante alors

- la méthode de Jacobi est convergente,
- si $w \in]0, 1]$ la méthode de Relaxation est convergente.

Théorème

Soit $\mathbb{A}$ une matrice hermitienne inversible en décomposée en $\mathbb{A} = \mathbb{M} - \mathbb{N}$ où $\mathbb{M}$ est inversible. Soit $\mathbb{B} = \mathbb{I} - \mathbb{M}^{-1}\mathbb{A}$, la matrice de l'itération. Supposons que $\mathbb{M}^* + \mathbb{N}$ (qui est hermitienne) soit définie positive. Alors $\rho(\mathbb{B}) < 1$ si et seulement si $\mathbb{A}$ est définie positive.

Théorème

Soit $\mathbb{A}$ une matrice hermitienne définie positive, alors la méthode de relaxation converge si et seulement si $w \in]0, 2[$.

Plan

Principe de base

Résoudre :

$$\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$$

Méthodes itératives :

$$\boldsymbol{x}^{[0]} \in \mathbb{K}^n \quad \text{et} \quad \boldsymbol{x}^{[k+1]} = \mathbb{B}\boldsymbol{x}^{[k]} + \boldsymbol{c}$$

Algorithme :

$\boldsymbol{x}^{[0]}$ donné
Pour $k = 0,\ 1, \cdots$ **faire**
  $\boldsymbol{x}^{[k+1]} \leftarrow \mathbb{B}\boldsymbol{x}^{[k]} + \boldsymbol{c}$
Fin Pour

Critère d'arrêt? Stockage de tous les $\boldsymbol{x}^{[k]}$?

La convergence de ces méthodes n'est pas assurées et si il y a convergence le nombre d'itération nécessaire n'est (à priori) pas connu.

$$\Longrightarrow \text{ boucle } \textbf{Tantque}$$

Critères d'arrêt :

- nombre maximum d'itérations
- $\varepsilon > 0$ permet l'arrêt des calculs si $\boldsymbol{x}^{[k]}$ suffisament proche de $\bar{\boldsymbol{x}} = \mathbb{A}^{-1}\boldsymbol{b}$

Comment choisir le critère d'arrêt pour la convergence?

La convergence de ces méthodes n'est pas assurées et si il y a convergence le nombre d'itération nécessaire n'est (à priori) pas connu.

$$\Longrightarrow \text{ boucle } \textbf{Tantque}$$

Critères d'arrêt :

- nombre maximum d'itérations
- $\varepsilon > 0$ permet l'arrêt des calculs si $\boldsymbol{x}^{[k]}$ suffisament proche de $\bar{\boldsymbol{x}} = \mathbb{A}^{-1}\boldsymbol{b}$

Comment choisir le critère d'arrêt pour la convergence?

Exemple de critère d'arrêt pour la convergence :
Soit $\boldsymbol{r}^{[k]} = \boldsymbol{b} - \mathbb{A}\boldsymbol{x}^{[k]}$ le résidu.

$$\frac{\|\boldsymbol{r}^{[k]}\|}{\|\boldsymbol{b}\|} \leqslant \varepsilon$$

La convergence de ces méthodes n'est pas assurées et si il y a convergence le nombre d'itération nécessaire n'est (à priori) pas connu.

$$\Longrightarrow \text{ boucle } \textbf{Tantque}$$

Critères d'arrêt :

- nombre maximum d'itérations
- $\varepsilon > 0$ permet l'arrêt des calculs si $\boldsymbol{x}^{[k]}$ suffisament proche de $\bar{\boldsymbol{x}} = \mathbb{A}^{-1}\boldsymbol{b}$

Comment choisir le critère d'arrêt pour la convergence?

Exemple de critère d'arrêt pour la convergence :
Soit $\boldsymbol{r}^{[k]} = \boldsymbol{b} - \mathbb{A}\boldsymbol{x}^{[k]}$ le résidu.

$$\frac{\left\|\boldsymbol{r}^{[k]}\right\|}{\|\boldsymbol{b}\|} \leqslant \varepsilon$$

Car dans ce cas, on a avec $\boldsymbol{e}^{[k]} = \bar{\boldsymbol{x}} - \boldsymbol{x}^{[k]}$

$$\frac{\left\|\boldsymbol{e}^{[k]}\right\|}{\|\bar{\boldsymbol{x}}\|} \leqslant \varepsilon \operatorname{cond}(\mathbb{A})$$

Algorithme 12 Méthode itérative pour la résolution d'un système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$

Données :

$\mathbb{A}$	:	matrice de $\mathcal{M}_n(\mathbb{K})$,
$\boldsymbol{b}$	:	vecteur de $\mathbb{K}^n$,
$\boldsymbol{x}^0$	:	vecteur initial de $\mathbb{K}^n$,
ε	:	la tolérence, $\varepsilon \in \mathbb{R}^+$,
kmax	:	nombre maximum d'itérations, $\mathrm{kmax} \in \mathbb{N}^*$

Résultat :

$\boldsymbol{x}^{\mathrm{tol}}$: un vecteur de $\mathbb{K}^n$ si convergence, sinon $\varnothing$

1: $k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
2: $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
3: $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
4: **Tantque** $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
5: $k \leftarrow k + 1$
6: $\boldsymbol{p} \leftarrow \boldsymbol{x}$ ▷ $\boldsymbol{p}$ contient le vecteur précédent
7: $\boldsymbol{x} \leftarrow$ calcul de l'itérée suivante en fonction de $\boldsymbol{p}$, $\mathbb{A}$, $\boldsymbol{b}$, ...
8: $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
9: **Fin Tantque**
10: **Si** $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors** ▷ Convergence
11: $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
12: **Fin Si**

Pour Jacobi , la suite des itérées est définie par

$$x_i^{[k+1]} = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1, j\neq i}^{n} \mathrm{A}_{ij} x_j^{[k]} \right), \quad \forall i \in [\![1, n]\!].$$

Algorithme 13 $\mathcal{R}_0$

1: $k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
2: $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
3: $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
4: **Tantque** $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
5: $k \leftarrow k + 1$
6: $\boldsymbol{p} \leftarrow \boldsymbol{x}$
7:
8: $\boldsymbol{x} \leftarrow$ calcul par Jacobi
9: $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
10: **Fin Tantque**
11: **Si** $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors** ▷ Convergence
12: $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
13: **Fin Si**

Algorithme 13 $\mathcal{R}_1$

1: $k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
2: $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
3: $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
4: **Tantque** $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
5: $k \leftarrow k + 1$
6: $\boldsymbol{p} \leftarrow \boldsymbol{x}$
7: **Pour** $i \leftarrow 1$ à n **faire**
8: $x_i \leftarrow \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1, j\neq i}^{n} \mathrm{A}_{ij} \boldsymbol{p}_j \right)$
9: **Fin Pour**
10: $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
11: **Fin Tantque**
12: **Si** $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors** ▷ Convergence
13: $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
14: **Fin Si**

Algorithme 13 $\boxed{\mathcal{R}_1}$

1: $k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
2: $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
3: $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
4: **Tantque** $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
5: $\quad k \leftarrow k + 1$
6: $\quad \boldsymbol{p} \leftarrow \boldsymbol{x}$
7: $\quad$ **Pour** $i \leftarrow 1$ à n **faire**
8:
9: $$x_i \leftarrow \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1, j \neq i}^{n} \mathrm{A}_{ij} \boldsymbol{p}_j \right)$$
10: $\quad$ **Fin Pour**
11: $\quad \boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
12: **Fin Tantque**
13: **Si** $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
14: $\quad \boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
15: **Fin Si**

Algorithme 13 $\boxed{\mathcal{R}_2}$

1: $k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
2: $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
3: $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
4: **Tantque** $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
5: $\quad k \leftarrow k + 1$
6: $\quad \boldsymbol{p} \leftarrow \boldsymbol{x}$
7: $\quad$ **Pour** $i \leftarrow 1$ à n **faire**
8: $\quad\quad \mathrm{S} \leftarrow 0$
9: $\quad\quad$ **Pour** $j \leftarrow 1$ à n $(j \neq i)$ **faire**
10: $\quad\quad\quad \mathrm{S} \leftarrow \mathrm{S} + \mathrm{A}_{i,j} p_j$
11: $\quad\quad$ **Fin Pour**
12: $\quad\quad x_i \leftarrow \frac{1}{\mathrm{A}_{ii}} (b_i - \mathrm{S})$
13: $\quad$ **Fin Pour**
14: $\quad \boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
15: **Fin Tantque**
16: **Si** $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
17: $\quad \boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
18: **Fin Si**

Algorithme 13 Méthode itérative de Jacobi pour la résolution d'un système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$

Données :

$\mathbb{A}$	:	matrice de $\mathcal{M}_n(\mathbb{K})$,
$\boldsymbol{b}$	:	vecteur de $\mathbb{K}^n$,
$\boldsymbol{x}^0$	:	vecteur initial de $\mathbb{K}^n$,
ε	:	la tolérence, $\varepsilon \in \mathbb{R}^+$,
kmax	:	nombre maximum d'itérations, $\mathrm{kmax} \in \mathbb{N}^*$

Résultat :

$\boldsymbol{X}$	:	un vecteur de $\mathbb{K}^n$

1: **Fonction** $\boldsymbol{X} \leftarrow$ RSLJacobi $(\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \mathrm{kmax})$
2: $k \leftarrow 0$, $\boldsymbol{X} \leftarrow \varnothing$
3: $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
4: $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
5: **Tantque** $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
6: $k \leftarrow k + 1$
7: $\boldsymbol{p} \leftarrow \boldsymbol{x}$
8: **Pour** $i \leftarrow 1$ **à** n **faire**
9: $\mathrm{S} \leftarrow 0$
10: **Pour** $j \leftarrow 1$ **à** n $(j \neq i)$ **faire**
11: $\mathrm{S} \leftarrow \mathrm{S} + A(i,j) * p(j)$
12: **Fin Pour**
13: $x(i) \leftarrow (b(i) - \mathrm{S})/A(i,i)$
14: **Fin Pour**
15: $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
16: **Fin Tantque**
17: **Si** $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
18: $\boldsymbol{X} \leftarrow \boldsymbol{x}$
19: **Fin Si**
20: **Fin Fonction**

Pour Gauss-Seidel , la suite des itérées est définie par

$$x_i^{(k+1)} = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1}^{i-1} \mathrm{A}_{ij} x_j^{[k+1]} - \sum_{j=i+1}^{n} \mathrm{A}_{ij} x_j^{[k]} \right) \quad \forall i \in [\![1, n]\!]$$

Algorithme 14 $\mathcal{R}_0$

1: $k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
2: $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
3: $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
4: **Tantque** $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
5: $k \leftarrow k + 1$
6: $\boldsymbol{p} \leftarrow \boldsymbol{x}$
7:
8: $\boldsymbol{x} \leftarrow$ calcul par Gauss-Seidel
9: $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
10: **Fin Tantque**
11: **Si** $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
12: $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
13: **Fin Si**

Algorithme 14 $\mathcal{R}_1$

1: $k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
2: $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
3: $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
4: **Tantque** $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
5: $k \leftarrow k + 1$
6: $\boldsymbol{p} \leftarrow \boldsymbol{x}$
7: **Pour** $i \leftarrow 1$ à n **faire**
8: $x_i \leftarrow \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1}^{i-1} \mathrm{A}_{ij} x_j - \sum_{j=i+1}^{n} \mathrm{A}_{ij} p_j \right)$
9: **Fin Pour**
10: $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
11: **Fin Tantque**
12: **Si** $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
13: $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
14: **Fin Si**

Algorithme 14 $\boxed{\mathcal{R}_1}$

1: $k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
2: $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
3: $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
4: **Tantque** $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
5: $k \leftarrow k + 1$
6: $\boldsymbol{p} \leftarrow \boldsymbol{x}$
7: **Pour** $i \leftarrow 1$ à n **faire**
8:
9: $$x_i \leftarrow \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1}^{i-1} \mathrm{A}_{ij} x_j - \sum_{j=i+1}^{n} \mathrm{A}_{ij} p_j \right)$$
10: **Fin Pour**
11: $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
12: **Fin Tantque**
13: **Si** $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
14: $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
15: **Fin Si**

Algorithme 14 $\boxed{\mathcal{R}_2}$

1: $k \leftarrow 0$, $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \varnothing$
2: $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
3: $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
4: **Tantque** $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
5: $k \leftarrow k + 1$
6: $\boldsymbol{p} \leftarrow \boldsymbol{x}$
7: **Pour** $i \leftarrow 1$ à n **faire**
8: $\mathrm{S} \leftarrow 0$
9: **Pour** $j \leftarrow 1$ à $i - 1$ **faire**
10: $\mathrm{S} \leftarrow \mathrm{S} + \mathrm{A}_{i,j} x_j$
11: **Fin Pour**
12: **Pour** $j \leftarrow i + 1$ à n **faire**
13: $\mathrm{S} \leftarrow \mathrm{S} + \mathrm{A}_{i,j} p_j$
14: **Fin Pour**
15: $x_i \leftarrow \frac{1}{\mathrm{A}_{ii}} (b_i - \mathrm{S})$
16: **Fin Pour**
17: $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
18: **Fin Tantque**
19: **Si** $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
20: $\boldsymbol{x}^{\mathrm{tol}} \leftarrow \boldsymbol{x}$
21: **Fin Si**

Algorithme 14 Méthode itérative de Gauss-Seidel pour la résolution d'un système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$

Données :

$\mathbb{A}$	:	matrice de $\mathcal{M}_n(\mathbb{K})$,
$\boldsymbol{b}$	:	vecteur de $\mathbb{K}^n$,
$\boldsymbol{x}^0$	:	vecteur initial de $\mathbb{K}^n$,
ε	:	la tolérence, $\varepsilon \in \mathbb{R}^+$,
kmax	:	nombre maximum d'itérations, $\text{kmax} \in \mathbb{N}^*$

Résultat :

$\boldsymbol{X}$: un vecteur de $\mathbb{K}^n$

1: **Fonction** $\boldsymbol{X} \leftarrow$ RSLGaussSeidel $(\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \text{kmax})$
2: $k \leftarrow 0$, $\boldsymbol{X} \leftarrow \varnothing$
3: $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
4: $\text{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
5: **Tantque** $\|\boldsymbol{r}\| > \text{tol}$ et $k \leqslant \text{kmax}$ **faire**
6: $k \leftarrow k + 1$
7: $\boldsymbol{p} \leftarrow \boldsymbol{x}$
8: **Pour** $i \leftarrow 1$ **à** n **faire**
9: $\text{S} \leftarrow 0$
10: **Pour** $j \leftarrow 1$ **à** $i - 1$ **faire**
11: $\text{S} \leftarrow \text{S} + A(i,j) * x(j)$
12: **Fin Pour**
13: **Pour** $j \leftarrow i + 1$ **à** n **faire**
14: $\text{S} \leftarrow \text{S} + A(i,j) * p(j)$
15: **Fin Pour**
16: $x(i) \leftarrow (b(i) - \text{S})/A(i,i)$
17: **Fin Pour**
18: $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
19: **Fin Tantque**
20: **Si** $\|\boldsymbol{r}\| \leqslant \text{tol}$ **alors**
21: $\boldsymbol{X} \leftarrow \boldsymbol{x}$
22: **Fin Si**
23: **Fin Fonction**

Fonction $\boldsymbol{X} \leftarrow$ RSLJacobi $(\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \text{kmax})$
  $k \leftarrow 0$, $\boldsymbol{X} \leftarrow \varnothing$
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  $\text{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
  Tantque $\|\boldsymbol{r}\| > \text{tol}$ et $k \leqslant \text{kmax}$ **faire**
    $k \leftarrow k + 1$
    $\boldsymbol{p} \leftarrow \boldsymbol{x}$
    Pour $i \leftarrow 1$ **à** n **faire**
      $\text{S} \leftarrow 0$
      Pour $j \leftarrow 1$ **à** n $(j \neq i)$ **faire**
        $\text{S} \leftarrow \text{S} + A(i,j) * p(j)$
      Fin Pour
      $x(i) \leftarrow (b(i) - \text{S})/A(i,i)$
    Fin Pour
    $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  Fin Tantque
  Si $\|\boldsymbol{r}\| \leqslant \text{tol}$ **alors**
    $\boldsymbol{X} \leftarrow \boldsymbol{x}$
  Fin Si
Fin Fonction

Fonction $\boldsymbol{X} \leftarrow$ RSLGaussSeidel $(\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \text{kmax})$
  $k \leftarrow 0$, $\boldsymbol{X} \leftarrow \varnothing$
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  $\text{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
  Tantque $\|\boldsymbol{r}\| > \text{tol}$ et $k \leqslant \text{kmax}$ **faire**
    $k \leftarrow k + 1$
    $\boldsymbol{p} \leftarrow \boldsymbol{x}$
    Pour $i \leftarrow 1$ **à** n **faire**
      $\text{S} \leftarrow 0$
      Pour $j \leftarrow 1$ **à** $i - 1$ **faire**
        $\text{S} \leftarrow \text{S} + A(i,j) * x(j)$
      Fin Pour
      Pour $j \leftarrow i + 1$ **à** n **faire**
        $\text{S} \leftarrow \text{S} + A(i,j) * p(j)$
      Fin Pour
      $x(i) \leftarrow (b(i) - \text{S})/A(i,i)$
    Fin Pour
    $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  Fin Tantque
  Si $\|\boldsymbol{r}\| \leqslant \text{tol}$ **alors**
    $\boldsymbol{X} \leftarrow \boldsymbol{x}$
  Fin Si
Fin Fonction

Fonction $X \leftarrow$ RSLJacobi $(\mathbb{A}, b, x^0, \varepsilon, \text{kmax})$
  $k \leftarrow 0$, $X \leftarrow \varnothing$
  $x \leftarrow x^0$, $r \leftarrow b - \mathbb{A} * x$,
  $\text{tol} \leftarrow \varepsilon(\|b\| + 1)$
  Tantque $\|r\| > \text{tol}$ et $k \leqslant \text{kmax}$ **faire**
    $k \leftarrow k + 1$
    $p \leftarrow x$
    Pour $i \leftarrow 1$ à n **faire**
      $S \leftarrow 0$
      Pour $j \leftarrow 1$ à n $(j \neq i)$ **faire**
        $S \leftarrow S + A(i,j) * p(j)$
      Fin Pour
      $x(i) \leftarrow (b(i) - S)/A(i,i)$
    Fin Pour
    $r \leftarrow b - \mathbb{A} * x$,
  Fin Tantque
  Si $\|r\| \leqslant \text{tol}$ **alors**
    $X \leftarrow x$
  Fin Si
Fin Fonction

Fonction $X \leftarrow$ RSLGaussSeidel $(\mathbb{A}, b, x^0, \varepsilon, \text{kmax})$
  $k \leftarrow 0$, $X \leftarrow \varnothing$
  $x \leftarrow x^0$, $r \leftarrow b - \mathbb{A} * x$,
  $\text{tol} \leftarrow \varepsilon(\|b\| + 1)$
  Tantque $\|r\| > \text{tol}$ et $k \leqslant \text{kmax}$ **faire**
    $k \leftarrow k + 1$
    $p \leftarrow x$
    Pour $i \leftarrow 1$ à n **faire**
      $S \leftarrow 0$
      Pour $j \leftarrow 1$ à $i - 1$ **faire**
        $S \leftarrow S + A(i,j) * x(j)$
      Fin Pour
      Pour $j \leftarrow i + 1$ à n **faire**
        $S \leftarrow S + A(i,j) * p(j)$
      Fin Pour
      $x(i) \leftarrow (b(i) - S)/A(i,i)$
    Fin Pour
    $r \leftarrow b - \mathbb{A} * x$,
  Fin Tantque
  Si $\|r\| \leqslant \text{tol}$ **alors**
    $X \leftarrow x$
  Fin Si
Fin Fonction

Même ossature puisque toutes deux basées sur l'Algorithme générique

Peut-on simplifier, clarifier et racourcir les codes?

Fonction $\boldsymbol{X} \leftarrow$ RSLJacobi $(\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \mathrm{kmax})$
  $k \leftarrow 0$, $\boldsymbol{X} \leftarrow \varnothing$
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
  Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
    $k \leftarrow k + 1$
    $\boldsymbol{p} \leftarrow \boldsymbol{x}$
    Pour $i \leftarrow 1$ **à** n **faire**
      $\mathrm{S} \leftarrow 0$
      Pour $j \leftarrow 1$ **à** n $(j \neq i)$ **faire**
        $\mathrm{S} \leftarrow \mathrm{S} + A(i,j) * p(j)$
      Fin Pour
      $x(i) \leftarrow (b(i) - \mathrm{S})/A(i,i)$
    Fin Pour
    $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  Fin Tantque
  Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
    $\boldsymbol{X} \leftarrow \boldsymbol{x}$
  Fin Si
Fin Fonction

Algorithme 15 Itération de Jacobi : calcul de $\boldsymbol{x}$ tel que

$$x_i = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1, j \neq i}^{n} \mathrm{A}_{ij} y_j \right), \quad \forall i \in [\![1, n]\!].$$

Données :
A : matrice de $\mathcal{M}_n(\mathbb{K})$,
b : vecteur de $\mathbb{K}^n$,
y : vecteur de $\mathbb{K}^n$,
Résultat :
x : un vecteur de $\mathbb{K}^n$

1: **Fonction** $\mathrm{x} \leftarrow$ IterJacobi $(\mathrm{A}, \mathrm{b}, \mathrm{y})$
2: **Pour** $i \leftarrow 1$ **à** n **faire**
3: $S \leftarrow 0$
4: **Pour** $j \leftarrow 1$ **à** n $(j \neq i)$ **faire**
5: $S \leftarrow S + \mathrm{A}(i,j) * \mathrm{y}(j)$
6: **Fin Pour**
7: $\mathrm{x}(i) \leftarrow (\mathrm{b}(i) - S)/\mathrm{A}(i,i)$
8: **Fin Pour**
9: **Fin Fonction**

Fonction $\boldsymbol{X} \leftarrow$ RSLJacobi2 $(\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \mathrm{kmax})$
  $k \leftarrow 0$, $\boldsymbol{X} \leftarrow \varnothing$
  $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  $\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
  Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
    $k \leftarrow k + 1$
    $\boldsymbol{p} \leftarrow \boldsymbol{x}$
    $\boldsymbol{x} \leftarrow$ IterJacobi$(\mathbb{A}, \boldsymbol{b}, \boldsymbol{p})$
    $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
  Fin Tantque
  Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
    $\boldsymbol{X} \leftarrow \boldsymbol{x}$
  Fin Si
Fin Fonction

Algorithme 16 Itération de Jacobi : calcul de $\boldsymbol{x}$ tel que

$$x_i = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1, j \neq i}^{n} \mathrm{A}_{ij} y_j \right), \quad \forall i \in [\![1, n]\!].$$

Données :
A : matrice de $\mathcal{M}_n(\mathbb{K})$,
b : vecteur de $\mathbb{K}^n$,
y : vecteur de $\mathbb{K}^n$,
Résultat :
x : un vecteur de $\mathbb{K}^n$

1: **Fonction** x $\leftarrow$ IterJacobi (A, b, y)
2: **Pour** $i \leftarrow 1$ à n **faire**
3: $S \leftarrow 0$
4: **Pour** $j \leftarrow 1$ à n $(j \neq i)$ **faire**
5: $S \leftarrow S + \mathrm{A}(i, j) * \mathrm{y}(j)$
6: **Fin Pour**
7: $\mathrm{x}(i) \leftarrow (\mathrm{b}(i) - S)/\mathrm{A}(i, i)$
8: **Fin Pour**
9: **Fin Fonction**

Fonction $\boldsymbol{X} \leftarrow$ RSLGaussSeidel $(\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \mathrm{kmax})$

$k \leftarrow 0$, $\boldsymbol{X} \leftarrow \varnothing$
$\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
$\mathrm{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
Tantque $\|\boldsymbol{r}\| > \mathrm{tol}$ et $k \leqslant \mathrm{kmax}$ **faire**
$k \leftarrow k + 1$
$\boldsymbol{p} \leftarrow \boldsymbol{x}$
Pour $i \leftarrow 1$ **à** n **faire**
$\mathrm{S} \leftarrow 0$
Pour $j \leftarrow 1$ **à** $i - 1$ **faire**
$\mathrm{S} \leftarrow \mathrm{S} + A(i,j) * x(j)$
Fin Pour
Pour $j \leftarrow i + 1$ **à** n **faire**
$\mathrm{S} \leftarrow \mathrm{S} + A(i,j) * p(j)$
Fin Pour
$x(i) \leftarrow (b(i) - \mathrm{S})/A(i,i)$
Fin Pour
$\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
Fin Tantque
Si $\|\boldsymbol{r}\| \leqslant \mathrm{tol}$ **alors**
$\boldsymbol{X} \leftarrow \boldsymbol{x}$
Fin Si
Fin Fonction

Algorithme 17 Itération de Gauss-Seidel : calcul de $\boldsymbol{x}$ tel que

$$x_i = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j - \sum_{j=i+1}^{n} \mathrm{A}_{i,j} y_j \right), \quad \forall i \in [\![1, n]\!].$$

Données :
A : matrice de $\mathcal{M}_n(\mathbb{K})$,
b : vecteur de $\mathbb{K}^n$,
y : vecteur de $\mathbb{K}^n$,
Résultat :
x : un vecteur de $\mathbb{K}^n$

1: **Fonction** x $\leftarrow$ IterGaussSeidel (A, b, y)
2: **Pour** $i \leftarrow 1$ **à** n **faire**
3: $S \leftarrow 0$
4: **Pour** $j \leftarrow 1$ **à** $i - 1$ **faire**
5: $S \leftarrow S + \mathrm{A}(i,j) * \mathrm{x}(j)$
6: **Fin Pour**
7: **Pour** $j \leftarrow i + 1$ **à** n **faire**
8: $S \leftarrow S + \mathrm{A}(i,j) * \mathrm{y}(j)$
9: **Fin Pour**
10: $\mathrm{x}(i) \leftarrow (\mathrm{b}(i) - S)/\mathrm{A}(i,i)$
11: **Fin Pour**
12: **Fin Fonction**

```
Fonction X ← RSLGaussSeidel2 ( 𝔸, b, x^0, ε, kmax )
  k ← 0, X ← ∅
  x ← x^0, r ← b − 𝔸 ∗ x,
  tol ← ε(‖b‖ + 1)
  Tantque ‖r‖ > tol et k ⩽ kmax faire
    k ← k + 1
    p ← x
    x ← IterGaussSeidel(𝔸, b, p)
    r ← b − 𝔸 ∗ x,
  Fin Tantque
  Si ‖r‖ ⩽ tol alors
    X ← x
  Fin Si
Fin Fonction
```

Algorithme 18 Itération de Gauss-Seidel : calcul de x tel que

$$x_i = \frac{1}{\mathrm{A}_{ii}} \left(b_i - \sum_{j=1}^{i-1} \mathrm{A}_{i,j} x_j - \sum_{j=i+1}^{n} \mathrm{A}_{i,j} y_j \right), \quad \forall i \in [\![1, n]\!].$$

Données :

A : matrice de $\mathcal{M}_n(\mathbb{K})$,
b : vecteur de $\mathbb{K}^n$,
y : vecteur de $\mathbb{K}^n$,

Résultat :

x : un vecteur de $\mathbb{K}^n$

```
1:  Fonction x ← IterGaussSeidel ( A, b, y )
2:    Pour i ← 1 à n faire
3:      S ← 0
4:      Pour j ← 1 à i − 1 faire
5:        S ← S + A(i, j) ∗ x(j)
6:      Fin Pour
7:      Pour j ← i + 1 à n faire
8:        S ← S + A(i, j) ∗ y(j)
9:      Fin Pour
10:     x(i) ← (b(i) − S)/A(i, i)
11:   Fin Pour
12: Fin Fonction
```

Fonction $X \leftarrow$ RSLGaussSeidel2 $(\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \text{kmax})$

$k \leftarrow 0$, $\boldsymbol{X} \leftarrow \varnothing$
$\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
$\text{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
Tantque $\|\boldsymbol{r}\| > \text{tol}$ et $k \leqslant \text{kmax}$ **faire**
$k \leftarrow k + 1$
$\boldsymbol{p} \leftarrow \boldsymbol{x}$
$\boldsymbol{x} \leftarrow$ IterGaussSeidel$(\mathbb{A}, \boldsymbol{b}, \boldsymbol{p})$
$\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
Fin Tantque
Si $\|\boldsymbol{r}\| \leqslant \text{tol}$ **alors**
$\boldsymbol{X} \leftarrow \boldsymbol{x}$
Fin Si
Fin Fonction

Fonction $X \leftarrow$ RSLJacobi2 $(\mathbb{A}, \boldsymbol{b}, \boldsymbol{x}^0, \varepsilon, \text{kmax})$

$k \leftarrow 0$, $\boldsymbol{X} \leftarrow \varnothing$
$\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
$\text{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
Tantque $\|\boldsymbol{r}\| > \text{tol}$ et $k \leqslant \text{kmax}$ **faire**
$k \leftarrow k + 1$
$\boldsymbol{p} \leftarrow \boldsymbol{x}$
$\boldsymbol{x} \leftarrow$ IterJacobi$(\mathbb{A}, \boldsymbol{b}, \boldsymbol{p})$
$\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A} * \boldsymbol{x}$,
Fin Tantque
Si $\|\boldsymbol{r}\| \leqslant \text{tol}$ **alors**
$\boldsymbol{X} \leftarrow \boldsymbol{x}$
Fin Si
Fin Fonction

Les deux codes sont fortement similaires!

Peut-on éviter les copier/coller et gagner encore en lisibilité?

Ecriture Algorithme générique sous forme d'une fonction et on ajoute aux paramètres d'entrées une fonction formelle ITERFONC calculant une itérée :

$$\boldsymbol{x} \leftarrow \text{ITERFONC}(\mathbb{A}, \boldsymbol{b}, \boldsymbol{y}).$$

Algorithme 18 Méthode itérative pour la résolution d'un système linéaire $\mathbb{A}\boldsymbol{x} = \boldsymbol{b}$

Données :

$\mathbb{A}$	:	matrice de $\mathcal{M}_n(\mathbb{K})$,
$\boldsymbol{b}$	:	vecteur de $\mathbb{K}^n$,
ITERFONC	:	fonction de paramètres une matrice d'ordre n, et deux vecteurs de $\mathbb{K}^n$. retourne un vecteur de $\mathbb{K}^n$.
$\boldsymbol{x}^0$	:	vecteur initial de $\mathbb{K}^n$,
ε	:	la tolérence, $\varepsilon \in \mathbb{R}^+$,
kmax	:	nombre maximum d'itérations, $\text{kmax} \in \mathbb{N}^*$

Résultat :

$\boldsymbol{x}^{\text{tol}}$: un vecteur de $\mathbb{K}^n$ si convergence, sinon $\varnothing$

1: **Fonction** $\boldsymbol{X} \leftarrow$ RSLMETHITER $(\mathbb{A}, \boldsymbol{b}, \text{ITERFONC}, \boldsymbol{x}^0, \varepsilon, \text{kmax})$
2: $k \leftarrow 0$, $\boldsymbol{x}^{\text{tol}} \leftarrow \varnothing$
3: $\boldsymbol{x} \leftarrow \boldsymbol{x}^0$, $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
4: $\text{tol} \leftarrow \varepsilon(\|\boldsymbol{b}\| + 1)$
5: **Tantque** $\|\boldsymbol{r}\| > \text{tol}$ et $k \leqslant \text{kmax}$ **faire**
6: $k \leftarrow k + 1$
7: $\boldsymbol{p} \leftarrow \boldsymbol{x}$
8: $\boldsymbol{x} \leftarrow \text{ITERFONC}(\mathbb{A}, \boldsymbol{b}, \boldsymbol{p})$
9: $\boldsymbol{r} \leftarrow \boldsymbol{b} - \mathbb{A}\boldsymbol{x}$,
10: **Fin Tantque**
11: **Si** $\|\boldsymbol{r}\| \leqslant \text{tol}$ **alors**
12: $\boldsymbol{x}^{\text{tol}} \leftarrow \boldsymbol{x}$
13: **Fin Si**
14: **Fin Fonction**

Fonction $X \leftarrow$ RSLJacobi3 $(\mathbb{A}, b, x^0, \varepsilon, \text{kmax})$
  $X \leftarrow$ RSLMethIter$(\mathbb{A}, b,$ IterJacobi$, x^0, \varepsilon, \text{kmax})$
Fin Fonction

Fonction $X \leftarrow$ RSLGaussSeidel3 $(\mathbb{A}, b, x^0, \varepsilon, \text{kmax})$
  $X \leftarrow$ RSLMethIter$(\mathbb{A}, b,$ IterGaussSeidel$, x^0, \varepsilon, \text{kmax})$
Fin Fonction

Algorithme 18 Méthode itérative pour la résolution d'un système linéaire $\mathbb{A}x = b$

Données :

$\mathbb{A}$	:	matrice de $\mathcal{M}_n(\mathbb{K})$,
b	:	vecteur de $\mathbb{K}^n$,
IterFonc	:	fonction de paramètres une matrice d'ordre n, et deux vecteurs de $\mathbb{K}^n$. retourne un vecteur de $\mathbb{K}^n$.
x^0	:	vecteur initial de $\mathbb{K}^n$,
ε	:	la tolérence, $\varepsilon \in \mathbb{R}^+$,
kmax	:	nombre maximum d'itérations, $\text{kmax} \in \mathbb{N}^*$

Résultat :

x^{tol} : un vecteur de $\mathbb{K}^n$ si convergence, sinon $\varnothing$

1: **Fonction** $X \leftarrow$ RSLMethIter $(\mathbb{A}, b,$ IterFonc$, x^0, \varepsilon, \text{kmax})$
2: $k \leftarrow 0$, $x^{\text{tol}} \leftarrow \varnothing$
3: $x \leftarrow x^0$, $r \leftarrow b - \mathbb{A}x$,
4: $\text{tol} \leftarrow \varepsilon(\|b\| + 1)$
5: **Tantque** $\|r\| > \text{tol}$ et $k \leqslant \text{kmax}$ **faire**
6: $k \leftarrow k + 1$
7: $p \leftarrow x$
8: $x \leftarrow$ IterFonc$(\mathbb{A}, b, p)$
9: $r \leftarrow b - \mathbb{A}x$,
10: **Fin Tantque**
11: **Si** $\|r\| \leqslant \text{tol}$ **alors**
12: $x^{\text{tol}} \leftarrow x$
13: **Fin Si**
14: **Fin Fonction**

Méthode de relaxation utilisant Gauss-Seidel,avec $w \in \mathbb{R}^*$,

$$x_i^{[k+1]} = \frac{w}{A_{ii}} \left(b_i - \sum_{j=1}^{i-1} A_{ij} x_j^{[k+1]} - \sum_{j=i+1}^{n} A_{ij} x_j^{[k]} \right) + (1-w) x_i^{[k]} \quad \forall i \in [\![1, n]\!]$$

Algorithme 19 Itération S.O.R.

Données :

A : matrice de $\mathcal{M}_n(\mathbb{K})$,
b : vecteur de $\mathbb{K}^n$,
y : vecteur de $\mathbb{K}^n$,
w : réel non nul.

Résultat :

x : un vecteur de $\mathbb{K}^n$

```
1:  Fonction x ← IterSOR ( A, b, y, w )
2:    Pour i ← 1 à n faire
3:      S ← 0
4:      Pour j ← 1 à i − 1 faire
5:        S ← S − A(i, j) ∗ x(j)
6:      Fin Pour
7:      Pour j ← i + 1 à n faire
8:        S ← S − A(i, j) ∗ y(j)
9:      Fin Pour
10:     x(i) ← w ∗ (b(i) − S)/A(i, i) + (1 − w) ∗ x(i)
11:   Fin Pour
12: Fin Fonction
```

Paramètre w "en trop" dans l'appel de la fonction IterSOR pour pouvoir utiliser la fonction générique RSLMethIter !

Méthode de relaxation utilisant Gauss-Seidel,avec $w \in \mathbb{R}^*$,

$$x_i^{[k+1]} = \frac{w}{A_{ii}} \left(b_i - \sum_{j=1}^{i-1} A_{ij} x_j^{[k+1]} - \sum_{j=i+1}^{n} A_{ij} x_j^{[k]} \right) + (1-w) x_i^{[k]} \quad \forall i \in [\![1, n]\!]$$

Algorithme 20 Itération S.O.R.

Données :

A : matrice de $\mathcal{M}_n(\mathbb{K})$,
b : vecteur de $\mathbb{K}^n$,
y : vecteur de $\mathbb{K}^n$,
w : réel non nul.

Résultat :

x : un vecteur de $\mathbb{K}^n$

```
1:  Fonction x ← IterSOR ( A, b, y, w )
2:    Pour i ← 1 à n faire
3:      S ← 0
4:      Pour j ← 1 à i − 1 faire
5:        S ← S − A(i, j) ∗ x(j)
6:      Fin Pour
7:      Pour j ← i + 1 à n faire
8:        S ← S − A(i, j) ∗ y(j)
9:      Fin Pour
10:     x(i) ← w ∗ (b(i) − S)/A(i, i) + (1 − w) ∗ x(i)
11:   Fin Pour
12: Fin Fonction
```

Paramètre w "en trop" dans l'appel de la fonction IterSOR pour pouvoir utiliser la fonction générique RSLMethIter !

Fonction $X \leftarrow$ RSLSOR3 $(\mathbb{A}, \boldsymbol{b}, w, \boldsymbol{x}^0, \varepsilon, \text{kmax})$
  IterFun $\leftarrow ((\mathbb{M}, \boldsymbol{r}, \boldsymbol{s}) \mapsto \text{IterSOR}(\mathbb{M}, \boldsymbol{r}, \boldsymbol{s}, w))$
  $\boldsymbol{X} \leftarrow$ RSLMethIter$(\mathbb{A}, \boldsymbol{b}, \text{IterFun}, \boldsymbol{x}^0, \varepsilon, \text{kmax})$
Fin Fonction