

Analyse Numérique I

Sup'Galilée, Ingénieurs MACS, 1ère année / L3 MIM

François Cuvelier

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

2023/10/10

Chapitre IV

Résolution de systèmes linéaires

1 Conditionnement

2 Méthodes directes

- Matrices diagonales
- Matrices triangulaires inférieures
- Matrices triangulaires supérieures
- Ecriture algébrique

- Résultats théoriques
- Utilisation pratique
- Résultats théoriques
- Résolution d'un système linéaire
- Algorithme : Factorisation positive de Cholesky
- La transformation de Householder

3 Méthodes itératives

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

Résoudre

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

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 $\mathbb{M}\mathbb{A}$ *facilement* inversible

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

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

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

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

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

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

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

$$\mathbb{A}\mathbf{x} = \mathbf{b} \quad \Longleftrightarrow \quad \mathbf{x}^t = (1, 1, 1, 1)$$

$$\begin{aligned} \mathbb{A}\mathbf{u} = (\mathbf{b} + \Delta\mathbf{b}) \quad &\Longleftrightarrow \quad \mathbf{u}^t = \left(\frac{91}{50}, -\frac{9}{25}, \frac{27}{20}, \frac{79}{100} \right) \\ &\approx (1.8, -0.36, 1.3, 0.79) \end{aligned}$$

$$(\mathbb{A} + \Delta\mathbb{A})\mathbf{v} = \mathbf{b} \quad \Longleftrightarrow \quad \mathbf{v}^t = (-81, 137, -34, 22)$$

$$\begin{aligned} (\mathbb{A} + \Delta\mathbb{A})\mathbf{y} = (\mathbf{b} + \Delta\mathbf{b}) \quad &\Longleftrightarrow \quad \mathbf{y}^t = \left(-\frac{18283543}{461600}, \frac{31504261}{461600}, -\frac{3741501}{230800}, \frac{5235241}{461600} \right) \\ &\approx (-39.61, 68.25, -16.21, 11.34) \end{aligned}$$

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

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

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

Non, pas forcément!

Le système linéaire précédent 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'entraînent qu'une variation *raisonnable* de la solution.

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

♥ Definition 1.1

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

$$\text{cond}(A) = \|A\| \|A^{-1}\|.$$

Nous noterons $\text{cond}_p(A) = \|A\|_p \|A^{-1}\|_p$.



Proposition:



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

- ① $\forall \alpha \in \mathbb{K}^*, \text{cond}(\alpha \mathbb{A}) = \text{cond}(\mathbb{A})$.
- ② $\text{cond}_p(\mathbb{A}) \geq 1, \forall p \in \llbracket 1, +\infty \rrbracket$.
- ③ $\text{cond}_2(\mathbb{A}) = 1$ si et seulement si $\mathbb{A} = \alpha \mathbb{Q}$ avec $\alpha \in \mathbb{K}^*$ et \mathbb{Q} matrice unitaire



Théorème 2:



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

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

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

$$\frac{\|\Delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \text{cond}(\mathbb{A}) \frac{\|\Delta\mathbf{b}\|}{\|\mathbf{b}\|}$$

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



Théorème:



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

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

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

$$\frac{\|\Delta\mathbf{x}\|}{\|\mathbf{x} + \Delta\mathbf{x}\|} \leq \text{cond}(\mathbb{A}) \frac{\|\Delta\mathbb{A}\|}{\|\mathbb{A}\|}.$$

Remarque 1

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

1 Conditionnement

2 Méthodes directes

• Matrices particulières

- Matrices diagonales
- Matrices triangulaires inférieures
- Matrices triangulaires supérieures

• Exercices et résultats préliminaires

• Méthode de Gauss-Jordan

- Ecriture algébrique

• Factorisation LU

- Résultats théoriques

- Utilisation pratique

• Factorisation LDL*

• Factorisation de Cholesky

- Résultats théoriques
- Résolution d'un système linéaire
- Algorithme : Factorisation positive de Cholesky

• Factorisation QR

- La transformation de Householder

3 Méthodes itératives

Système diagonal

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

$$x_i = b_i/A_{i,i}, \quad \forall i \in \llbracket 1, n \rrbracket. \quad (1)$$

Algorithm Fonction **RSLMatDIAG** permettant de résoudre le système linéaire à matrice diagonale inversible

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

Données : \mathbb{A} : matrice diagonale de $\mathcal{M}_n(\mathbb{R})$ inversible.

\mathbf{b} : vecteur de \mathbb{R}^n .

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

```
1: Fonction  $\mathbf{x} \leftarrow \text{RSLMatDIAG} (\mathbb{A}, \mathbf{b})$ 
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 $i < j$)

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

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

Système triangulaire inférieur

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

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

$$\mathbb{A} \text{ inversible} \iff A_{i,i} \neq 0, \forall i \in \llbracket 1, n \rrbracket$$

Système triangulaire inférieur

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

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

$$\mathbb{A} \text{ inversible} \iff A_{i,i} \neq 0, \forall i \in \llbracket 1, n \rrbracket$$

$$\text{Soit } i \in \llbracket 1, n \rrbracket, \quad (\mathbb{A}\mathbf{x})_i = b_i, \iff \sum_{j=1}^n A_{i,j}x_j = b_i.$$

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

$$x_i = \frac{1}{A_{i,i}} \left(b_i - \sum_{j=1}^{i-1} A_{i,j}x_j \right), \quad \forall i \in \llbracket 1, n \rrbracket. \quad (2)$$

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

Algorithme 2 \mathcal{R}_0

1: Résoudre $\mathbb{A}\mathbf{x} = \mathbf{b}$ en calculant successivement x_1, x_2, \dots, 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**

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

Algorithme 2 \mathcal{R}_1

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

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

2:

3: **Fin Pour**

Algorithme 2 \mathcal{R}_2

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

$$\left. \begin{array}{l} 2: \quad S \leftarrow \sum_{j=1}^{i-1} A_{i,j} x_j \\ 3: \quad x_i \leftarrow (b_i - S)/A_{i,i} \end{array} \right\}$$

4: **Fin Pour**

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

Algorithme 2 \mathcal{R}_2

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

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

2:

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

4: **Fin Pour**

Algorithme 2 \mathcal{R}_3

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**

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

Algorithm Fonction **RSLTRIINF** permettant de résoudre le système linéaire triangulaire inférieur inversible

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

Données : \mathbb{A} : matrice triangulaire de $\mathcal{M}_n(\mathbb{K})$ inférieure inversible.

\mathbf{b} : vecteur de \mathbb{K}^n .

Résultat : \mathbf{x} : vecteur de \mathbb{K}^n .

```

1: Fonction  $\mathbf{x} \leftarrow \text{RSLTRIINF} (\mathbb{A}, \mathbf{b})$ 
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 $i > j$)

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



Exercice

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

1 Conditionnement

2 Méthodes directes

- Matrices particulières
 - Matrices diagonales
 - Matrices triangulaires inférieures
 - Matrices triangulaires supérieures
- Exercices et résultats préliminaires
- Méthode de Gauss-Jordan
 - Ecriture algébrique
- Factorisation LU

- Résultats théoriques

- Utilisation pratique

- Factorisation LDL*

- Factorisation de Cholesky

- Résultats théoriques

- Résolution d'un système linéaire

- Algorithme : Factorisation positive de Cholesky

- Factorisation QR

- La transformation de Householder

3 Méthodes itératives



Lemme 3.1:



Soit $(i, j) \in \llbracket 1, n \rrbracket^2$. On note $\mathbb{P}_n^{[i,j]} \in \mathcal{M}_n(\mathbb{R})$ la matrice identité 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 3.2:



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

$$\mathbb{E} \mathbb{A} \mathbf{e}_1 = A_{1,1} \mathbf{e}_1 \quad (3)$$

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



Théorème 4: Décomposition de Schur



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

$$A = UTU^* \quad (4)$$

1 Conditionnement

2 Méthodes directes

- Matrices particulières
 - Matrices diagonales
 - Matrices triangulaires inférieures
 - Matrices triangulaires supérieures
- Exercices et résultats préliminaires
- **Méthode de Gauss-Jordan**
 - Ecriture algébrique
- Factorisation LU

- Résultats théoriques

- Utilisation pratique

• Factorisation LDL*

• Factorisation de Cholesky

- Résultats théoriques

- Résolution d'un système linéaire

- Algorithme : Factorisation positive de Cholesky

• Factorisation QR

- La transformation de Householder

3 Méthodes itératives

Algorithme de Gauss-Jordan

$$Ax = b \iff Ux = f$$

où 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.

- **Etape j :** on va *s'arranger* pour annuler les termes sous-diagonaux de la colonne j de la matrice sans modifier les $j - 1$ premières colonnes.

$$\begin{pmatrix} \bullet & \bullet & \dots & \bullet & \vdots & \bullet & \dots & \bullet \\ 0 & \ddots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \ddots & \ddots & \bullet & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & \bullet & \vdots & \vdots & \vdots & \vdots \\ \hline 0 & \dots & \dots & 0 & \bullet & \dots & \bullet & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & \dots & 0 & \bullet & \dots & \bullet & \vdots \end{pmatrix} x = \begin{pmatrix} \bullet \\ \vdots \\ \bullet \\ \vdots \\ \bullet \\ \vdots \\ \bullet \end{pmatrix}$$

$\xleftarrow{j-1}$

- **Etape j :** on va *s'arranger* pour annuler les termes sous-diagonaux de la colonne j de la matrice sans modifier les $j - 1$ premières colonnes.

$$\left(\begin{array}{cccc|cccc} \bullet & \bullet & \dots & \bullet & \bullet & \dots & \bullet & \\ 0 & \ddots & \ddots & \vdots & \vdots & & \vdots & \\ \vdots & \ddots & \ddots & \bullet & \vdots & & \vdots & \\ 0 & \dots & 0 & \bullet & \bullet & \dots & \bullet & \\ \hline 0 & \dots & \dots & 0 & \bullet & \dots & \bullet & \\ \vdots & & & \vdots & \vdots & & \vdots & \\ 0 & \dots & \dots & 0 & \bullet & \dots & \bullet & \end{array} \right) \mathbf{x} = \left(\begin{array}{c} \bullet \\ \vdots \\ \bullet \\ \vdots \\ \bullet \\ \vdots \\ \bullet \end{array} \right)$$

$\xleftarrow{j-1}$

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

- 1: **Pour** $j \leftarrow 1$ à $n - 1$ **faire**
- 2: Rechercher l'indice k de la ligne du pivot (sur la colonne j , $k \in \llbracket j, n \rrbracket$)
- 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.

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

Données : \mathbb{A} : matrice de $\mathcal{M}_n(\mathbb{K})$ inversible.

\mathbf{b} : vecteur de \mathbb{K}^n .

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

```
1: Fonction  $\mathbf{x} \leftarrow \text{RSLGAUSS} (\mathbb{A}, \mathbf{b})$ 
2:   Pour  $j \leftarrow 1$  à  $n - 1$  faire
3:      $k \leftarrow \boxed{\text{CHERCHEINDPIVOT}}(\mathbb{A}, j)$  ▷ à écrire
4:      $[\mathbb{A}, \mathbf{b}] \leftarrow \boxed{\text{PERMLIGNESYS}}(\mathbb{A}, \mathbf{b}, j, k)$  ▷ à écrire
5:     Pour  $i \leftarrow j + 1$  à  $n$  faire
6:        $[\mathbb{A}, \mathbf{b}] \leftarrow \boxed{\text{COMBLIGNESYS}}(\mathbb{A}, \mathbf{b}, j, i, -A(i, j)/A(j, j))$  ▷ à écrire
7:     Fin Pour
8:   Fin Pour
9:    $\mathbf{x} \leftarrow \text{RSLTriSup}(\mathbb{A}, \mathbf{b})$  ▷ déjà écrite
10: Fin Fonction
```



Exercice 1: 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 $\mathbb{G}\mathbb{A}\mathbf{e}_1 = \alpha\mathbf{e}_1$ avec $\alpha \neq 0$ et \mathbf{e}_1 premier vecteur de la base canonique de \mathbb{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 $\mathbf{b} \in \mathbb{C}^n$. En supposant connue la décomposition précédente $\mathbb{S}_n\mathbb{A}_n = \mathbb{U}_n$, expliquer comment résoudre le système $\mathbb{A}_n\mathbf{x} = \mathbf{b}$.

Q.3 Que peut-on dire si \mathbb{A} est non inversible?

Indication : utiliser les Lemmes 3.1 et 3.2.

On a donc démontré le théorème suivant



Théorème 5

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

1 Conditionnement

2 Méthodes directes

- Matrices particulières
 - Matrices diagonales
 - Matrices triangulaires inférieures
 - Matrices triangulaires supérieures
- Exercices et résultats préliminaires
- Méthode de Gauss-Jordan
 - Ecriture algébrique
- Factorisation LU

- Résultats théoriques

- Utilisation pratique

• Factorisation LDL*

• Factorisation de Cholesky

- Résultats théoriques

- Résolution d'un système linéaire

- Algorithme : Factorisation positive de Cholesky

• Factorisation QR

- La transformation de Householder

3 Méthodes itératives



Exercice: Factorisation LU



Soit $A \in \mathcal{M}_n(\mathbb{C})$ une matrice dont les sous-matrices principales d'ordre i , notées Δ_i , $i \in \llbracket 1, n \rrbracket$ (voir Définition B.48, page 99) sont inversibles.

Montrer qu'il existe des matrices $E^{[k]} \in \mathcal{M}_n(\mathbb{C})$, $k \in \llbracket 1, n-1 \rrbracket$, triangulaires inférieures à diagonale unité telles que la matrice U définie par

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

soit triangulaire supérieure avec $U_{i,i} = \det \Delta_i / (U_{1,1} \times \dots \times U_{i-1,i-1})$, $\forall i \in \llbracket 1, n \rrbracket$.



Théorème 6: Factorisation LU



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

$$A = LU.$$

preuve :

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

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

- **Unicité** : $A = L_1 U_1 = L_2 U_2 \dots$



Exercice 2

Montrer que si A inversible admet une factorisation LU alors toutes ses sous-matrices principales sont inversibles.



Corollaire 6.1:



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

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

Remarque 2

Si la matrice $A \in \mathcal{M}_n(\mathbb{C})$ 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 7: Factorisation LU avec permutations



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

$$PA = LU. \quad (5)$$

Utilisation pratique de la factorisation LU

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

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

$$A\mathbf{x} = \mathbf{b} \iff LU\mathbf{x} = \mathbf{b} \quad (6)$$

est équivalent à

Utilisation pratique de la factorisation LU

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

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

$$Ax = b \iff LUx = b \quad (6)$$

est équivalent à

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

$$Ux = y \quad (7)$$

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

$$Ly = b. \quad (8)$$

Algorithme de résolution de systèmes linéaire par LU

Algorithm Fonction RSLFactLU permettant de résoudre, par une factorisation LU, le système linéaire $\mathbb{A}\mathbf{x} = \mathbf{b}$ où \mathbb{A} est une matrice de $\mathcal{M}_n(\mathbb{K})$, dont toutes les sous-matrices principales sont inversibles, et $\mathbf{b} \in \mathbb{K}^n$.

, **Données** : \mathbb{A} : matrice de $\mathcal{M}_n(\mathbb{K})$ dont les sous-matrices principales sont inversibles;
 \mathbf{b} : vecteur de \mathbb{K}^n .

Résultat : \mathbf{x} : vecteur de \mathbb{K}^n .

```
1: Fonction  $\mathbf{x} \leftarrow \text{RSLFactLU} (\mathbb{A}, \mathbf{b})$   
2:    $[\mathbb{L}, \mathbb{U}] \leftarrow \text{FactLU}(\mathbb{A})$   
3:    $\mathbf{y} \leftarrow \text{RSLTriInf}(\mathbb{L}, \mathbf{b})$   
4:    $\mathbf{x} \leftarrow \text{RSLTriSup}(\mathbb{U}, \mathbf{y})$   
5: Fin Fonction
```

▷ Factorisation LU
▷ Résolution du système $\mathbb{L}\mathbf{y} = \mathbf{b}$
▷ Résolution du système $\mathbb{U}\mathbf{x} = \mathbf{y}$

Il nous faut donc écrire la fonction **FactLU**

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

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

On connaît A , on cherche L et U

Soit $A \in \mathcal{M}_n(\mathbb{K})$ admettant une factorisation 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 connaît la **première ligne** de $L \implies$ on peut calculer la **première ligne** de U

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

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

• **Etape 1 :**

- ▶ On connaît la **première ligne** de $L \implies$ on peut calculer la **première ligne** de U
- ▶ On connaît la **première colonne** de $U \implies$ on peut calculer la **première colonne** de L

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

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

- **Etape 1 :**

- ▶ On connaît la **première ligne** de $L \implies$ on peut calculer la **première ligne** de U
- ▶ On connaît la **première colonne** de $U \implies$ on peut calculer la **première colonne** de L

- **Etape 2 :**

- ▶ On connaît la **deuxième ligne** de $L \implies$ on peut calculer la **deuxième ligne** de U car on connaît la première ligne de U

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

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

• Etape 1 :

- ▶ On connaît la **première ligne** de $L \implies$ on peut calculer la **première ligne** de U
- ▶ On connaît la **première colonne** de $U \implies$ on peut calculer la **première colonne** de L

• Etape 2 :

- ▶ On connaît la **deuxième ligne** de $L \implies$ on peut calculer la **deuxième ligne** de U car on connaît la première ligne de U
- ▶ On connaît la **deuxième colonne** de $U \implies$ on peut calculer la **deuxième colonne** de L car on connaît la première colonne de L

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

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

- **Etape 1 :**

- ▶ On connaît la **première ligne** de $L \implies$ on peut calculer la **première ligne** de U
- ▶ On connaît la **première colonne** de $U \implies$ on peut calculer la **première colonne** de L

- **Etape 2 :**

- ▶ On connaît la **deuxième ligne** de $L \implies$ on peut calculer la **deuxième ligne** de U car on connaît la première ligne de U
- ▶ On connaît la **deuxième colonne** de $U \implies$ on peut calculer la **deuxième colonne** de L car on connaît la première colonne de L

- ...

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

$$\mathbb{A} = \left(\begin{array}{c|cccc} & \xrightarrow{i-1} & & & \\ \hline \begin{array}{c} \bullet \quad 0 \quad \cdots \quad 0 \\ \bullet \quad \diagdown \quad \diagup \quad \vdots \\ \vdots \quad \diagdown \quad \diagup \quad \bullet \\ \bullet \quad \cdots \quad \bullet \end{array} & \begin{array}{c} 0 \quad \cdots \quad \cdots \quad 0 \\ \vdots \quad \vdots \quad \vdots \quad \vdots \\ 0 \quad \cdots \quad \cdots \quad 0 \end{array} & & & \\ \hline \bullet \quad \cdots \quad \cdots \quad \bullet & L_{i,i} \quad 0 \quad \cdots \quad 0 & & & \\ \vdots & \bullet \quad \diagdown \quad \diagup \quad \vdots & & & \\ \vdots & \vdots \quad \vdots \quad \vdots \quad 0 & & & \\ \bullet \quad \cdots \quad \cdots \quad \bullet & \bullet \quad \cdots \quad \bullet \quad L_{n,n} \end{array} \right) \left(\begin{array}{c|cccc} & \xrightarrow{\text{Connus}} & & & \\ \hline \begin{array}{c} \bullet \quad \bullet \quad \cdots \quad \bullet \\ 0 \quad \diagdown \quad \diagup \quad \vdots \\ \vdots \quad \diagdown \quad \diagup \quad \bullet \\ 0 \quad \cdots \quad 0 \end{array} & \begin{array}{c} \bullet \quad \cdots \quad \cdots \quad \bullet \\ \vdots \quad \vdots \quad \vdots \quad \vdots \\ \bullet \quad \cdots \quad \cdots \quad \bullet \end{array} & & & \\ \hline 0 \quad \cdots \quad \cdots \quad 0 & U_{i,i} \quad \bullet \quad \cdots \quad \bullet & & & \\ \vdots & 0 \quad \diagdown \quad \diagup \quad \vdots & & & \\ \vdots & \vdots \quad \vdots \quad \vdots \quad \bullet & & & \\ 0 \quad \cdots \quad \cdots \quad 0 & 0 \quad \cdots \quad 0 \quad U_{n,n} \end{array} \right) \begin{array}{c} \uparrow \\ i-1 \\ \downarrow \end{array}$$

Connus

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

$$\mathbb{A} = \left(\begin{array}{c|c} \begin{array}{c} \xrightarrow{i-1} \\ \begin{array}{ccc} \bullet & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \bullet & \dots & \bullet & 0 \\ \vdots & \ddots & \vdots & \vdots \\ \bullet & \dots & \bullet & 0 \\ \vdots & \ddots & \vdots & \vdots \\ \bullet & \dots & \bullet & 0 \end{array} \\ \text{Connus} \end{array} & \begin{array}{c} L_{n,n} \end{array} \end{array} \right) \left(\begin{array}{c|c} \begin{array}{c} \xrightarrow{i-1} \\ \begin{array}{ccc} \bullet & \bullet & \dots & \bullet \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \vdots \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & \vdots \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \dots & 0 & \vdots \end{array} \\ \text{Connus} \end{array} & \begin{array}{c} U_{n,n} \end{array} \end{array} \right)$$

$\Rightarrow L_{i,i} = 1$

\Rightarrow ligne i de \mathbb{L} connue :

on peut calculer la ligne i de \mathbb{U} !

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

The diagram illustrates the structure of the matrix A , which is a block matrix with two main parts, L and U , separated by a vertical dashed line. The left part, labeled L , contains a blue box labeled "Connus" and a red box labeled $L_{i,i}$. The right part, labeled U , contains a blue box labeled "Connus" and a red box labeled $U_{i,i}$. A red arrow points from the red box in L to the red box in U , labeled $L_{i,i} = 1$.

⇒ ligne i de \mathbb{L} connue :

on peut calculer la ligne i de U !

On cherche $U_{i,j} \forall j \in \llbracket i, n \rrbracket$.

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

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

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

$$\mathbb{A} = \left(\begin{array}{c|ccc} & \xrightarrow{i-1} & & & \\ \hline \begin{array}{c} \bullet \quad 0 \quad \cdots \quad 0 \\ \bullet \quad \ddots \quad \ddots \quad \vdots \\ \vdots \quad \ddots \quad \ddots \quad \bullet \\ \bullet \quad \cdots \quad \bullet \quad 0 \end{array} & \begin{array}{c} 0 \\ \vdots \\ 0 \end{array} & \cdots & \cdots & 0 \\ \hline \begin{array}{c} \bullet \quad \cdots \quad \bullet \\ \vdots \\ \bullet \end{array} & \begin{array}{c} L_{i,i} \\ \vdots \\ \bullet \end{array} & 0 & \cdots & 0 \\ \hline & & & & L_{n,n} \end{array} \right) \left(\begin{array}{c|ccc} & \xrightarrow{i-1} & & & \\ \hline \begin{array}{c} \bullet \quad \bullet \quad \cdots \quad \bullet \\ 0 \quad \ddots \quad \ddots \quad \vdots \\ 0 \quad \cdots \quad 0 \quad \bullet \end{array} & \begin{array}{c} \bullet \\ \vdots \\ 0 \end{array} & \cdots & \cdots & \bullet \\ \hline \begin{array}{c} 0 \quad \cdots \quad 0 \\ \vdots \\ 0 \end{array} & \begin{array}{c} U_{i,i} \\ \vdots \\ 0 \end{array} & \bullet & \cdots & \bullet \\ \hline & & & & U_{n,n} \end{array} \right)$$

Connus
Connus

\Rightarrow colonne i de \mathbb{U} connue : on peut calculer la colonne i de \mathbb{L} !

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

$$A = \left(\begin{array}{c|ccc} \mathbb{L} & & & \\ \hline \begin{array}{ccc} \xrightarrow{i-1} & & \\ \bullet & 0 & \cdots & 0 \\ \bullet & & \ddots & \\ \vdots & & & \\ \bullet & \cdots & \bullet & \bullet \\ \hline \bullet & & \bullet & 0 \\ \vdots & & & \\ \vdots & & & \\ \bullet & \cdots & \bullet & \end{array} & \begin{array}{c} 0 \\ \vdots \\ 0 \\ L_{i,i} \\ \bullet \\ \vdots \\ \bullet \end{array} & \cdots & \begin{array}{c} 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \\ L_{n,n} \end{array} \\ \hline \begin{array}{ccc} & & \\ \hline \bullet & & \bullet \\ \vdots & & \\ \vdots & & \\ \bullet & \cdots & \bullet \end{array} & \begin{array}{c} 0 \\ \vdots \\ 0 \\ U_{i,i} \\ \bullet \\ \vdots \\ 0 \end{array} & \cdots & \begin{array}{c} 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \\ U_{n,n} \end{array} \\ \hline \text{Connus} & & & \end{array} \right)$$

⇒ colonne i de \mathbb{U} connue : on peut calculer la colonne i de \mathbb{L} !

On cherche $L_{j,i} \forall j \in \llbracket i+1, n \rrbracket, (L_{i,i} = 1)$

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

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

En résumé, à l'étape i :

- Calcul de la ligne i de \mathbb{U}

$$U_{i,j} = 0, \quad \forall j \in \llbracket 1, i-1 \rrbracket,$$

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

- Calcul de la colonne i de \mathbb{L}

$$L_{j,i} = 0, \quad \forall j \in \llbracket 1, i-1 \rrbracket,$$

$$L_{i,i} = 1,$$

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

Algorithme 9 \mathcal{R}_0

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

Algorithme 9 \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 \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 \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

$$U_{i,j} \leftarrow A_{i,j} - \sum_{k=1}^{i-1} L_{i,k} U_{k,j}$$

- 6:
- 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

$$L_{j,i} \leftarrow \frac{1}{U_{i,i}} \left(A_{j,i} - \sum_{k=1}^{i-1} L_{j,k} U_{k,i} \right)$$

- 13:
- 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

$$\left\{ \begin{array}{l} S_1 \leftarrow \sum_{k=1}^{i-1} L_{i,k} U_{k,j} \\ U_{i,j} \leftarrow A_{i,j} - S_1 \end{array} \right.$$

- 6:
- 7:
- 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

$$\left\{ \begin{array}{l} S_2 \leftarrow \sum_{k=1}^{i-1} L_{j,k} U_{k,i} \\ L_{j,i} \leftarrow \frac{1}{U_{i,i}} (A_{j,i} - S_2) \end{array} \right.$$

- 14:
- 15:
- 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 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

$$\mathbb{A} = \mathbb{L}\mathbb{U}$$

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 $L_{i,i} = 1, \forall i \in \llbracket 1, n \rrbracket$

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

1: **Fonction** $[\mathbb{L}, \mathbb{U}] \leftarrow \text{FactLU}(\mathbb{A})$

2: $\mathbb{U} \leftarrow \mathbb{O}_n$

▷ \mathbb{O}_n matrice nulle $n \times n$

3: $\mathbb{L} \leftarrow \mathbb{I}_n$

▷ \mathbb{I}_n matrice identité $n \times n$

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

5: **Pour** $j \leftarrow i$ à n faire

▷ Calcul de la ligne i de \mathbb{U}

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** $j \leftarrow i + 1$ à n faire

▷ Calcul de la colonne i de \mathbb{L}

13: $S_2 \leftarrow 0$

14: **Pour** $k \leftarrow 1$ à $i - 1$ faire

15: $S_2 \leftarrow S_2 + L(j, k) * U(k, i)$

16: **Fin Pour**

17: $L(j, i) \leftarrow (A_{j,i} - S_2) / U(i, i).$

18: **Fin Pour**

19: **Fin Pour**

20: **Fin Fonction**

1 Conditionnement

2 Méthodes directes

- Matrices particulières
 - Matrices diagonales
 - Matrices triangulaires inférieures
 - Matrices triangulaires supérieures
- Exercices et résultats préliminaires
- Méthode de Gauss-Jordan
 - Ecriture algébrique
- Factorisation LU

- Résultats théoriques

- Utilisation pratique

• Factorisation LDL*

• Factorisation de Cholesky

- Résultats théoriques
 - Résolution d'un système linéaire
 - Algorithme : Factorisation positive de Cholesky
- ## • Factorisation QR
- La transformation de Householder

3 Méthodes itératives

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

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

R est alors triangulaire supérieure à diagonale unité. On a alors

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

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

Par unicité de la factorisation LU :

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



Théorème 8: 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

$$\mathbb{A} = \mathbb{L}\mathbb{D}\mathbb{L}^* \quad (9)$$

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



Corollaire 8.1:



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 coefficients diagonaux strictement positifs **si et seulement si** la matrice \mathbb{A} est hermitienne définie positive.

1 Conditionnement

2 Méthodes directes

- Matrices particulières
 - Matrices diagonales
 - Matrices triangulaires inférieures
 - Matrices triangulaires supérieures
- Exercices et résultats préliminaires
- Méthode de Gauss-Jordan
 - Ecriture algébrique
- Factorisation LU

- Résultats théoriques

- Utilisation pratique

• Factorisation LDL*

• Factorisation de Cholesky

- Résultats théoriques

- Résolution d'un système linéaire

- Algorithme : Factorisation positive de Cholesky

• Factorisation QR

- La transformation de Householder

3 Méthodes itératives

♥ Definition

Une **factorisation régulière de Cholesky** d'une matrice $A \in \mathcal{M}_n(\mathbb{C})$ est une factorisation $A = BB^*$ où B est une matrice triangulaire inférieure inversible.

Si les coefficients diagonaux de B sont positifs, on parle alors d'une **factorisation positive de Cholesky**.



Théorème: Factorisation de Cholesky



La matrice $A \in \mathcal{M}_n(\mathbb{C})$ admet une factorisation régulière de Cholesky **si et seulement si** la matrice A est hermitienne définie positive. Dans ce cas, elle admet une unique factorisation positive.

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

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

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

est équivalent à

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

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

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

est équivalent à

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

$$\mathbb{B}^*\mathbf{x} = \mathbf{y} \quad (11)$$

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

$$\mathbb{B}\mathbf{y} = \mathbf{b}. \quad (12)$$

Algorithme de résolution de systèmes linéaire par Cholesky

Algorithm Fonction **RSLCHOLESKY** permettant de résoudre, par une factorisation de Cholesky positive, le système linéaire

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

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

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

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

```
1: Fonction  $\mathbf{x} \leftarrow \text{RSLCHOLESKY}(\mathbb{A}, \mathbf{b})$ 
2:    $\mathbb{B} \leftarrow \text{CHOLESKY}(\mathbb{A})$ 
3:    $\mathbf{y} \leftarrow \text{RSLTRIINF}(\mathbb{B}, \mathbf{b})$ 
4:    $\mathbb{U} \leftarrow \text{MATADJOINTE}(\mathbb{B})$ 
5:    $\mathbf{x} \leftarrow \text{RSLTRISUP}(\mathbb{U}, \mathbf{y})$ 
6: Fin Fonction
```

- ▷ Factorisation positive de Cholesky
 - ▷ Résolution du système $\mathbb{B}\mathbf{y} = \mathbf{b}$
- ▷ Calcul de la matrice adjointe de \mathbb{B}
 - ▷ Résolution du système $\mathbb{B}^*\mathbf{x} = \mathbf{y}$

Il nous faut donc écrire la fonction **CHOLESKY**

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

$$A = BB^*$$

c'est à dire

$$\begin{pmatrix} A_{1,1} & \dots & A_{1,n} \\ \vdots & \ddots & \vdots \\ A_{n,1} & \dots & A_{n,n} \end{pmatrix} = \begin{pmatrix} B_{1,1} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ B_{n,1} & \dots & \dots & B_{n,n} \end{pmatrix} \begin{pmatrix} \overline{B_{1,1}} & \dots & \dots & \overline{B_{n,1}} \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \overline{B_{n,n}} \end{pmatrix}.$$

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

$$A = BB^*$$

c'est à dire

$$\begin{pmatrix} A_{1,1} & \dots & A_{1,n} \\ \vdots & \ddots & \dots \\ A_{n,1} & \dots & A_{n,n} \end{pmatrix} = \begin{pmatrix} B_{1,1} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ B_{n,1} & \dots & \dots & B_{n,n} \end{pmatrix} \begin{pmatrix} \overline{B_{1,1}} & \dots & \dots & \overline{B_{n,1}} \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \overline{B_{n,n}} \end{pmatrix}.$$

- Calcul de $B_{1,1}$ (la 1ère ligne de B est donc déterminée)
 \implies calcul 1ère colonne de B .

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

$$A = BB^*$$

c'est à dire

$$\begin{pmatrix} A_{1,1} & \dots & A_{1,n} \\ \vdots & \ddots & \dots \\ A_{n,1} & \dots & A_{n,n} \end{pmatrix} = \begin{pmatrix} B_{1,1} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ B_{n,1} & \dots & \dots & B_{n,n} \end{pmatrix} \begin{pmatrix} \overline{B_{1,1}} & \dots & \dots & \overline{B_{n,1}} \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \overline{B_{n,n}} \end{pmatrix}.$$

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

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

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

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

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

$$A_{i,i} = \sum_{k=1}^{i-1} |B_{i,k}|^2 + |B_{i,i}|^2$$

et donc

$$B_{i,i} = \left(A_{i,i} - \sum_{j=1}^{i-1} |B_{i,j}|^2 \right)^{1/2}.$$

Il reste à déterminer $B_{j,i}$, $\forall j \in \llbracket i+1, n \rrbracket$.

$$A_{j,i} = \sum_{k=1}^n B_{j,k} (\mathbb{B}^*)_{k,i} = \sum_{k=1}^n B_{j,k} \overline{B_{i,k}}, \quad \forall j \in \llbracket i+1, n \rrbracket$$

Comme \mathbb{L} est triangulaire inférieure on obtient

$$A_{j,i} = \sum_{k=1}^i B_{j,k} \overline{B_{i,k}} = \sum_{k=1}^{i-1} B_{j,k} \overline{B_{i,k}} + B_{j,i} \overline{B_{i,i}}, \quad \forall j \in \llbracket i+1, n \rrbracket$$

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

$$\begin{aligned} B_{j,i} &= \frac{1}{B_{i,i}} \left(A_{j,i} - \sum_{k=1}^{i-1} B_{j,k} \overline{B_{i,k}} \right), \quad \forall j \in \llbracket i+1, n \rrbracket \\ B_{j,i} &= 0, \quad \forall j \in \llbracket 1, i-1 \rrbracket. \end{aligned}$$

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 $B_{i,i}$, 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**

Calculer $B_{i,i}$, connaissant les
 $i - 1$ premières colonnes de \mathbb{B} .

2:

Calculer la $i^{\text{ème}}$ colonne de \mathbb{B} .

3:

4: **Fin Pour**

Algorithme 11 \mathcal{R}_2

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

2: $B_{i,i} \leftarrow \left(A_{i,i} - \sum_{j=1}^{i-1} |B_{i,j}|^2 \right)^{1/2}$

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

4: $B_{j,i} \leftarrow 0$

5: **Fin Pour**

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

7: $B_{j,i} \leftarrow \frac{1}{B_{i,i}} \left(A_{j,i} - \sum_{k=1}^{i-1} B_{j,k} \overline{B_{i,k}} \right)$

8: **Fin Pour**

9: **Fin Pour**

Algorithme 11 \mathcal{R}_2

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

$$B_{i,i} \leftarrow \left(A_{i,i} - \sum_{j=1}^{i-1} |B_{i,j}|^2 \right)^{1/2}$$

2:

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

4: $B_{j,i} \leftarrow 0$

5: **Fin Pour**

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

$$B_{j,i} \leftarrow \frac{1}{B_{i,i}} \left(A_{j,i} - \sum_{k=1}^{i-1} B_{j,k} \overline{B_{i,k}} \right).$$

7:

8: **Fin Pour**

9: **Fin Pour**

Algorithme 11 \mathcal{R}_3

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

$$2: \left\{ \begin{array}{l} S_1 \leftarrow \sum_{j=1}^{i-1} |B_{i,j}|^2 \end{array} \right.$$

$$3: B_{i,i} \leftarrow (A_{i,i} - S_1)^{1/2}$$

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

5: $B_{j,i} \leftarrow 0$

6: **Fin Pour**

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

$$8: \left\{ \begin{array}{l} S_2 \leftarrow \sum_{k=1}^{i-1} B_{j,k} \overline{B_{i,k}} \end{array} \right.$$

$$9: B_{j,i} \leftarrow \frac{1}{B_{i,i}} (A_{j,i} - S_2).$$

10: **Fin Pour**

11: **Fin Pour**

Algorithme 11 \mathcal{R}_3

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

$$S_1 \leftarrow \sum_{j=1}^{i-1} |B_{i,j}|^2$$

2:

$$B_{i,i} \leftarrow (A_{i,i} - S_1)^{1/2}$$

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

$$B_{j,i} \leftarrow 0$$

6: **Fin Pour**

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

$$S_2 \leftarrow \sum_{k=1}^{i-1} B_{j,k} \overline{B_{i,k}}$$

8:

$$B_{j,i} \leftarrow \frac{1}{B_{i,i}} (A_{j,i} - S_2).$$

10: **Fin Pour**

11: **Fin Pour**

Algorithme 11 \mathcal{R}_4

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

$$S_1 \leftarrow 0$$

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

$$S_1 \leftarrow S_1 + |B_{i,j}|^2$$

5: **Fin Pour**

$$B_{i,i} \leftarrow (A_{i,i} - S_1)^{1/2}$$

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

$$B_{j,i} \leftarrow 0$$

9: **Fin Pour**

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

$$S_2 \leftarrow 0$$

12: **Pour** $k \leftarrow 1$ à $i - 1$ faire

$$S_2 \leftarrow S_2 + B_{j,k} \overline{B_{i,k}}$$

14: **Fin Pour**

$$B_{j,i} \leftarrow \frac{1}{B_{i,i}} (A_{j,i} - S_2).$$

16: **Fin Pour**

17: **Fin Pour**

Algorithm 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 : \mathbb{A} : matrice de $\mathcal{M}_n(\mathbb{C})$ hermitienne définie positive.

Résultat : \mathbb{B} : matrice de $\mathcal{M}_n(\mathbb{C})$ triangulaire inférieure
avec $B(i, i) > 0, \forall i \in \llbracket 1, n \rrbracket$

```
1: Fonction B ← CHOLESKY ( A )
2:   Pour i ← 1 à n faire
3:     S1 ← 0
4:     Pour j ← 1 à i - 1 faire
5:       S1 ← S1 + |B(i, j)|2
6:     Fin Pour
7:     B(i, i) ← SQRT(A(i, i) - S1)
8:     Pour j ← 1 à i - 1 faire
9:       B(j, i) ← 0
10:    Fin Pour
11:    Pour j ← i + 1 à n faire
12:      S2 ← 0
13:      Pour k ← 1 à i - 1 faire
14:        S2 ← S2 + B(j, k) *  $\overline{B(i, k)}$ 
15:      Fin Pour
16:      B(j, i) ← (A(j, i) - S2)/B(i, i).
17:    Fin Pour
18:  Fin Pour
19: Fin Fonction
```



Exercice 3

Proposer une méthode permettant de tester la fonction **CHOLESKY** .

1 Conditionnement

2 Méthodes directes

- Matrices particulières
 - Matrices diagonales
 - Matrices triangulaires inférieures
 - Matrices triangulaires supérieures
- Exercices et résultats préliminaires
- Méthode de Gauss-Jordan
 - Ecriture algébrique
- Factorisation LU

- Résultats théoriques

- Utilisation pratique

• Factorisation LDL*

• Factorisation de Cholesky

- Résultats théoriques

- Résolution d'un système linéaire

- Algorithme : Factorisation positive de Cholesky

• Factorisation QR

- La transformation de Householder

3 Méthodes itératives

Definition: Matrice élémentaire de Householder

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

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

Propriété:



Toute matrice élémentaire de Householder est hermitienne et unitaire.



Propriété:



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

$$\mathbb{H}(\mathbf{u})(\mathbf{x}_{\perp} + \mathbf{x}_{\parallel}) = \mathbf{x}_{\perp} - \mathbf{x}_{\parallel}. \quad (14)$$

et

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



Théorème:



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

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



Exercice:



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

Q.1 Ecrire la fonction algorithmique **HOUSEHOLDER** permettant de retourner une matrice de Householder \mathbb{H} et $\alpha \in \mathbb{C}$ tels que $\mathbb{H}(\mathbf{u})\mathbf{a} = \alpha\mathbf{b}$. Le choix du α est fait par le paramètre δ (0 ou 1) de telle sorte que $\arg \alpha = -\arg(\langle \mathbf{a}, \mathbf{b} \rangle) + \delta\pi$ avec $|\alpha| = \|\mathbf{a}\|_2$.

Des fonctions comme **DOT**(\mathbf{a}, \mathbf{b}) (produit scalaire de deux vecteurs), **NORM**(\mathbf{a}) (norme 2 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 composantes étant dans $]0, 1[$ (loi uniforme).

Q.3 Proposer un programme permettant de vérifier que $\delta = 1$ est le "meilleur" choix.



Corollaire 8.2:



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

$$\mathbf{u}_{\pm} = \frac{\mathbf{a} \pm \|\mathbf{a}\|_2 e^{i\theta} \mathbf{e}_1}{\|\mathbf{a} \pm \|\mathbf{a}\|_2 e^{i\theta} \mathbf{e}_1\|}$$

Alors

$$\mathbb{H}(\mathbf{u}_{\pm})\mathbf{a} = \mp \|\mathbf{a}\|_2 e^{i\theta} \mathbf{e}_1 \quad (17)$$

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



Théorème 9:



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

$$A = QR. \quad (18)$$

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



Exercice: Algorithmique



Q.1 *Ecrire une fonction **FACTQR** permettant de calculer la factorisation QR d'une matrice $A \in \mathcal{M}_n(\mathbb{C})$.*

*On pourra utiliser la fonction **HOUSEHOLDER** (voir Exercice 54, page 78).*

Q.2 *Ecrire un programme permettant de tester cette fonction.*

1 Conditionnement

2 Méthodes directes

- Matrices particulières
 - Matrices diagonales
 - Matrices triangulaires inférieures
 - Matrices triangulaires supérieures
- Exercices et résultats préliminaires
- Méthode de Gauss-Jordan
 - Ecriture algébrique
- Factorisation LU

- Résultats théoriques

- Utilisation pratique

• Factorisation LDL*

• Factorisation de Cholesky

- Résultats théoriques

- Résolution d'un système linéaire

- Algorithme : Factorisation positive de Cholesky

• Factorisation QR

- La transformation de Householder

3 Méthodes itératives