

Systemes linéaires

Résolution de systèmes $n \times n$



A la fin du chapitre, l'étudiant doit **être capable de** :

1. Décrire l'algorithme de Gauss pour la résolution des systèmes linéaires
2. Justifier et décrire l'algorithme de Cholesky pour la résolution des systèmes SDP
3. Faire la distinction entre problème mal conditionné et méthode peu robuste aux erreurs de troncature
4. Donner l'ordre de grandeur du nombre d'opérations nécessaire à la résolution d'un système de grande taille, à l'inversion d'une matrice de grande taille
5. Décrire les algorithmes de Jacobi et de Gauss-Seidel
6. Faire un choix éclairé entre les méthodes du cours pour la résolution d'un problème donné



II - METHODES DIRECTES

Direct s'oppose à **itératif** ; on calcule « la » solution en une « fois ».

II.1- Méthode de Gauss

a- Conditions d'utilisation : méthode applicable pour tout système $n \times n$

b-Principe : triangularisation de la matrice A + balayage

« On ne change pas la solution d'un système linéaire si :

- on multiplie tous les termes d'une équation par une constante non nulle.

- si l'on remplace une équation par la somme, membre à membre de cette équation et d'une autre équation du système. »

On utilise ces propriétés pour triangulariser !

$$\begin{array}{|c|} \hline a_{ij} \\ \hline \end{array} \begin{array}{|c|} \hline x_j \\ \hline \end{array} = \begin{array}{|c|} \hline b_i \\ \hline \end{array} \longrightarrow \begin{array}{|c|} \hline 0 & a_{ij}^* \\ \hline \end{array} \begin{array}{|c|} \hline x_j \\ \hline \end{array} = \begin{array}{|c|} \hline b_i^* \\ \hline \end{array}$$

c- Application de la méthode à un exemple 3×3 :



**Triangularisation
étape 0**

$$\begin{cases} E_1^{(0)} & a_{11}^{(0)}x_1 + a_{12}^{(0)}x_2 + a_{13}^{(0)}x_3 = b_1^{(0)} \\ E_2^{(0)} & a_{21}^{(0)}x_1 + a_{22}^{(0)}x_2 + a_{23}^{(0)}x_3 = b_2^{(0)} \\ E_3^{(0)} & a_{31}^{(0)}x_1 + a_{32}^{(0)}x_2 + a_{33}^{(0)}x_3 = b_3^{(0)} \end{cases}$$

Elimination des termes en x_1 dans E_2 et E_3 ...

$$-\frac{a_{21}^{(0)}}{a_{11}^{(0)}} \times E_1^{(0)} + E_2^{(0)} = E_2^{(1)}$$

étape 1

$$\begin{cases} E_1^{(0)} & a_{11}^{(0)}x_1 + a_{12}^{(0)}x_2 + a_{13}^{(0)}x_3 = b_1^{(0)} \\ E_2^{(1)} & a_{22}^{(1)}x_2 + a_{23}^{(1)}x_3 = b_2^{(1)} \\ E_3^{(1)} & a_{32}^{(1)}x_2 + a_{33}^{(1)}x_3 = b_3^{(1)} \end{cases}$$

$$-\frac{a_{31}^{(0)}}{a_{11}^{(0)}} \times E_1^{(0)} + E_3^{(0)} = E_3^{(1)}$$

Elimination du terme en x_2 dans E_3 ...

$$-\frac{a_{32}^{(1)}}{a_{22}^{(1)}} \times E_2^{(1)} + E_3^{(1)} = E_3^{(2)} \quad \text{étape 2}$$

$$\begin{cases} E_1^{(0)} & a_{11}^{(0)}x_1 + a_{12}^{(0)}x_2 + a_{13}^{(0)}x_3 = b_1^{(0)} \\ E_2^{(1)} & a_{22}^{(1)}x_2 + a_{23}^{(1)}x_3 = b_2^{(1)} \\ E_3^{(2)} & a_{33}^{(2)}x_3 = b_3^{(2)} \end{cases}$$

Balayage :

$$\left\{ \begin{array}{l} x_1 = (b_1^{(0)} - a_{12}^{(0)}x_2 - a_{13}^{(0)}x_3) / a_{11}^{(0)} \\ x_2 = (b_2^{(1)} - a_{23}^{(1)}x_3) / a_{22}^{(1)} \\ x_3 = b_3^{(2)} / a_{33}^{(2)} \end{array} \right.$$



d- Définition importante

Les termes diagonaux par lesquels il est nécessaire de diviser lors de l'étape de balayage ($a_{11}^{(0)}, a_{22}^{(1)}, a_{33}^{(2)}$) sont appelés **pivots**

On verra plus tard qu'ils jouent un rôle très particulier ...

e- cas général



Étape $(r-1)$: $A^{(r-1)} X = B^{(r-1)}$

$$A^{(r-1)} = \begin{bmatrix} a_{11}^{(0)} & a_{12}^{(0)} & \dots & a_{1r-1}^{(0)} & a_{1r}^{(0)} & \dots & a_{1n}^{(0)} \\ 0 & a_{22}^{(1)} & \dots & a_{2r-1}^{(1)} & a_{2r}^{(1)} & \dots & a_{2n}^{(1)} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & a_{r-1r-1}^{(r-2)} & a_{r-1r}^{(r-2)} & \dots & a_{r-1n}^{(r-2)} \\ 0 & 0 & \dots & 0 & a_{rr}^{(r-1)} & \dots & a_{rn}^{(r-1)} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & 0 & a_{nr}^{(r-1)} & \dots & a_{nn}^{(r-1)} \end{bmatrix} \quad B^{(r-1)} = \begin{bmatrix} b_1^{(0)} \\ b_2^{(1)} \\ \vdots \\ b_{r-1}^{(r-2)} \\ b_r^{(r-1)} \\ \vdots \\ b_n^{(r-1)} \end{bmatrix}$$

Formules de Gauss :

étape $r-1$

$$a_{ij}^{(r)} = a_{ij}^{(r-1)} - \frac{a_{ir}^{(r-1)}}{a_{rr}^{(r-1)}} a_{rj}^{(r-1)} \quad (r+1 \leq i, j \leq n)$$

$$b_i^{(r)} = b_i^{(r-1)} - \frac{a_{ir}^{(r-1)}}{a_{rr}^{(r-1)}} b_r^{(r-1)} \quad (r+1 \leq i \leq n)$$



Étape $(n-1)$: $A^{(n-1)} X = B^{(n-1)}$

$$A^{(n-1)} = \begin{bmatrix} a_{11}^{(0)} & a_{12}^{(0)} & \dots & a_{1r-1}^{(0)} & a_{1r}^{(0)} & \dots & a_{1n}^{(0)} \\ 0 & a_{22}^{(1)} & \dots & a_{2r-1}^{(1)} & a_{2r}^{(1)} & \dots & a_{2n}^{(1)} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & a_{r-1r-1}^{(r-2)} & a_{r-1r}^{(r-2)} & \dots & a_{r-1n}^{(r-2)} \\ 0 & 0 & \dots & 0 & a_{rr}^{(r-1)} & \dots & a_{rn}^{(r-1)} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & 0 & 0 & \dots & a_{nn}^{(n-1)} \end{bmatrix} \quad B^{(n-1)} = \begin{bmatrix} b_1^{(0)} \\ b_2^{(1)} \\ \vdots \\ b_{r-1}^{(r-2)} \\ b_r^{(r-1)} \\ \vdots \\ b_n^{(n-1)} \end{bmatrix}$$

Balayage :

$$x_n = \frac{1}{a_{nn}^{(n-1)}} b_n^{(n-1)}$$
$$x_r = \frac{1}{a_{rr}^{(r-1)}} \left(b_r^{(r-1)} - \sum_{j=r+1}^n a_{rj}^{(r-1)} x_j \right) \quad 1 \leq r \leq n-1$$



Vue matricielle de la méthode de Gauss: un exemple

La matrice $\mathbf{E} = \begin{bmatrix} 1 & & \\ -2 & 1 & \\ & & 1 \end{bmatrix}$ est une matrice élémentaire qui remplace la ligne L_2 par $L_2 - 2L_1$

De même la matrice $\mathbf{F} = \begin{bmatrix} 1 & & \\ & 1 & \\ 1 & & 1 \end{bmatrix}$ est une matrice élémentaire qui remplace la ligne L_3 par $L_3 + L_1$

et la matrice $\mathbf{G} = \begin{bmatrix} 1 & & \\ & 1 & \\ & 1 & 1 \end{bmatrix}$ est une matrice élémentaire qui remplace la ligne L_3 par $L_3 + L_2$

Au final on passe de \mathbf{A} à \mathbf{U} en combinant les opérations élémentaires dans le bon ordre:

$$\mathbf{U} = \mathbf{GFEA}$$

Les mêmes opérations sont appliquées sur le membre de droite donc:

$$\mathbf{c} = \mathbf{GFEb}$$



Vue matricielle de la méthode de Gauss: un exemple

Après avoir écrit le système à partir de la matrice \mathbf{U} , est-on capable de revenir à la forme initiale écrite à partir de \mathbf{A} ? Oui bien sûr, en appliquant les transformations élémentaires inverses.

Par exemple, l'opération associée à la matrice \mathbf{E} est annulée par l'opération qui consiste à remplacer L_2 par $L_2 + 2 L_1$. La matrice élémentaire correspondante est simplement:

$$\begin{bmatrix} 1 & & \\ +2 & 1 & \\ & & 1 \end{bmatrix} = \begin{bmatrix} 1 & & \\ -2 & 1 & \\ & & 1 \end{bmatrix}^{-1} = \mathbf{E}^{-1}$$

En procédant de même pour les matrices élémentaires \mathbf{F} et \mathbf{G} , on obtient au final:

$$\mathbf{E}^{-1}\mathbf{F}^{-1}\mathbf{G}^{-1}\mathbf{U} = \mathbf{A}$$

$$\mathbf{E}^{-1}\mathbf{F}^{-1}\mathbf{G}^{-1}\mathbf{c} = \mathbf{b}$$



Vue matricielle de la méthode de Gauss: un exemple

Le produit des matrices élémentaires inverses a deux propriétés très importantes:

1. Elle est triangulaire inférieure
2. Ses coefficients contiennent l'information sur les transformations élémentaires permettant de passer de **U** à **A**

$$\mathbf{E}^{-1}\mathbf{F}^{-1}\mathbf{G}^{-1} = \begin{bmatrix} 1 & & & \\ +2 & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix} = \begin{bmatrix} 1 & & & \\ +2 & 1 & & \\ -1 & -1 & 1 & \end{bmatrix}$$

$L_2 \leftarrow L_2 + 2L_1$

$L_3 \leftarrow L_3 - L_1$

$L_3 \leftarrow L_3 - L_2$

Le produit des matrices élémentaires inverses est souvent noté **L** de sorte que:

$$\mathbf{A} = \mathbf{LU}$$

$$\mathbf{Lc} = \mathbf{b}$$



Factorisation LU et résolution de systèmes linéaires

La réécriture de toute matrice carrée \mathbf{A} comme le produit d'une matrice triangulaire inférieure \mathbf{L} et d'une matrice triangulaire supérieure \mathbf{U} a de très nombreuses applications en algèbre linéaire. Cette opération porte le nom de **factorisation LU**

Partant d'un système linéaire $\mathbf{AX}=\mathbf{b}$, elle permet d'obtenir deux systèmes:

$$\mathbf{LY}=\mathbf{b}$$

$$\mathbf{UX}=\mathbf{Y}$$

L'intérêt réside dans le fait que les matrices \mathbf{L} et \mathbf{U} étant triangulaires, ces deux systèmes peuvent être résolus très simplement par substitution directe (pour $\mathbf{LY}=\mathbf{b}$) ou inverse (pour $\mathbf{UX}=\mathbf{Y}$)

La résolution de $\mathbf{AX}=\mathbf{b}$ se fait alors en deux étapes:

1. Résolution de $\mathbf{LY}=\mathbf{b}$ qui donne \mathbf{Y} ,
2. Résolution de $\mathbf{UX}=\mathbf{Y}$ qui donne \mathbf{X} une fois \mathbf{Y} connu



Factorisation LU ou LDU

La factorisation **LU** est **non symétrique** dans le sens où:

1. **L**, triangulaire inférieure, contient des « 1 » sur sa diagonale,
2. **U**, triangulaire supérieure, contient les pivots sur sa diagonale

C'est la raison pour laquelle on préfère parfois écrire la matrice **A** sous la forme:

$$\mathbf{A} = \mathbf{LDU}$$

où les diagonales de **L** et **U** ne contiennent que des 1 et **D** est une matrice diagonale dont les éléments sont les pivots.

Evidemment cette écriture affecte les coefficients de la matrice **U** dont la $i^{\text{ème}}$ ligne doit être divisée par le $i^{\text{ème}}$ pivot (lien entre **U** et **U'**):

$$\mathbf{A} = \begin{bmatrix} 3 & 2 & 1 \\ 1 & 2 & 3 \\ 1 & 2 & 2 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & & \\ 1/3 & 1 & \\ 1/3 & 1 & 1 \end{bmatrix}}_L \underbrace{\begin{bmatrix} 3 & 2 & 1 \\ & 4/3 & 8/3 \\ & & -1 \end{bmatrix}}_U = \underbrace{\begin{bmatrix} 1 & & \\ 1/3 & 1 & \\ 1/3 & 1 & 1 \end{bmatrix}}_L \underbrace{\begin{bmatrix} 3 & & \\ & 4/3 & \\ & & -1 \end{bmatrix}}_D \underbrace{\begin{bmatrix} 1 & 2/3 & 1/3 \\ & 1 & 2 \\ & & 1_{13} \end{bmatrix}}_{U'}$$



Matrice singulière

Les factorisations **LU** et **LDU** sont équivalentes (au sens de l'information qu'elles contiennent) **SAUF** lorsque la matrice **A** est **singulière**

Dans ce cas, un **pivot** (au moins) **est égal à 0** et la factorisation **LDU** n'existe plus alors que **LU est toujours bien définie** (rien n'empêche **U** d'avoir un zéro sur sa diagonale ...)

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 1 & 2 & 3 \end{bmatrix} = \begin{bmatrix} 1 & & \\ 1/3 & 1 & \\ 1/3 & 1 & 1 \end{bmatrix} \begin{bmatrix} 3 & 2 & 1 \\ 4/3 & 8/3 & \\ 0 & & \end{bmatrix} = \mathbf{LU}$$

Dans ce cas, la factorisation **LU** ne peut évidemment pas être utilisée pour résoudre le système linéaire **AX=b** puisque le système **UX=Y** **n'admet pas de solution unique**.

Ce n'est **pas une faiblesse** de la méthode, c'est que le problème posé (résoudre **AX=b**) n'admet **pas de solution unique**



Qualité numérique - Précision

- Même lorsque la matrice **A** n'est pas singulière, la méthode de substitution de Gauss (ou factorisation **LU**, c'est la même chose) peut ne pas donner de bons résultats en **précision finie**
- Exemple: On suppose que l'on calcule avec une **précision de 10^{-3}** et que l'on veut résoudre le système suivant:

$$\begin{bmatrix} 10^{-4} & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \text{ dont la solution théorique est : } \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1/(1-10^{-4}) \\ (1-2 \times 10^{-4})/(1-10^{-4}) \end{bmatrix} \approx \begin{bmatrix} 1.0001 \\ 0.9999 \end{bmatrix}$$

- Appliquer la méthode directement conduit au système triangulaire:

$$L_2 \leftarrow L_2 - 10^4 L_1 \text{ d'où } \begin{bmatrix} 10^{-4} & 1 \\ 0 & -9999 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 \\ -9998 \end{bmatrix}$$



Qualité numérique - Précision

- Avec une erreur de **troncature de 10^{-3}** , on obtient en fait:

$$0 - 10000y = -10000 \text{ soit } y = 1, \text{ résultat valable à } 10^{-3} \text{ près}$$

- Avec $y=1$ (proche de 0.9999), le processus de substitution inverse donne alors pour x :

$$10^{-4}x + 1 = 1 \text{ soit } x = 0, \text{ très éloigné de } 1.0001 \quad !!!$$

- Autrement dit, le fait d'utiliser **un pivot « petit »** conduit à une très grande **sensibilité** du résultat aux erreurs de troncature
- Remarques:
 - 10^{-3} est énorme pour une troncature, mais pas pour une donnée expérimentale
 - Pour des problèmes de grande taille, une erreur de 10^{-15} peut conduire à une grande sensibilité des résultats
 - Ce problème a une solution: le « **partial pivoting** »



Qualité numérique - Précision

- Lorsqu'il s'agit de résoudre un système linéaire, l'ordre d'écriture des équations n'influence pas le résultat ... sauf en précision finie !!
- Exemple: On suppose que l'on calcule avec une **précision de 10^{-3}** et que l'on veut résoudre le système suivant:

$$\begin{bmatrix} 1 & 1 \\ 10^{-4} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \text{ dont la solution théorique est : } \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1/(1-10^{-4}) \\ (1-2 \times 10^{-4})/(1-10^{-4}) \end{bmatrix} \approx \begin{bmatrix} 1.0001 \\ 0.9999 \end{bmatrix}$$

- Appliquer la méthode directement conduit au système triangulaire:

$$L_2 \leftarrow L_2 - 10^{-4} L_1 \text{ d'où } \begin{bmatrix} 1 & 1 \\ 0 & 1-10^{-4} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2 \\ 1-10^{-4} \end{bmatrix}$$



Qualité numérique - Précision

- Avec une erreur de **troncature de 10^{-3}** , on obtient en fait:

$$0 + y = +1 \text{ soit } y = 1, \text{ résultat valable à } 10^{-3} \text{ près}$$

- Avec $y=1$ (proche de 0.9999), le processus de substitution inverse donne alors pour x :

$$x + 1 = 2 \text{ soit } x = 1, \text{ proche de } 1.0001$$

- Autrement dit, le fait d'utiliser **un pivot « grand »** conduit à une très faible **sensibilité** du résultat aux erreurs de troncature
- En termes de matrices, échanger les lignes du système linéaire revient à multiplier à gauche par une matrice de permutation

$$\begin{bmatrix} 1 & 1 \\ 10^{-4} & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 10^{-4} & 1 \\ 1 & 1 \end{bmatrix}$$



Factorisation LU avec pivotage partiel

- A chaque étape de substitution, on cherche dans les lignes à traiter le plus grand pivot en valeur absolue
- Autrement dit, les formules de Gauss données précédemment pour l'étape $r-1$ sont appliquées après avoir placé en position r la ligne i_{\max} dont le premier coefficient est tel que

$$\left| a_{i_{\max} r}^{(r-1)} \right| = \max_{r \leq i \leq n} \left| a_{ir}^{(r-1)} \right|$$

- En termes matriciel, chaque permutation entre les lignes r et i_{\max} revient à multiplier à gauche le système linéaire par une matrice de permutation fabriquée à partir de la matrice identité en échangeant les lignes r et i_{\max} :

$$\mathbf{AX} = \mathbf{b} \rightarrow \mathbf{PAX} = \mathbf{Pb} \text{ avec } \mathbf{P} \text{ telle que } \begin{cases} P_{r, i_{\max}} = P_{i_{\max}, r} = 1 \\ P_{i, i} = 1 \text{ si } i \neq r \text{ et } i \neq i_{\max} \\ P_{ij} = 0 \text{ sinon} \end{cases}$$



Nombre d'opérations pour résoudre un système linéaire

- On compte le nombre **d'opérations arithmétiques**
- Factorisation LU (ou élimination de Gauss) d'une matrice de taille n :
 - À l'étape k , il faut annuler $n-k$ coefficients en dessous du pivot
Pour chacune des lignes entre $k+1$ et n cela nécessite :
 - 1 division par le pivot pour calculer a_{ik} / a_{kk}
 - $n-k$ multiplications/soustractions pour effectuer $L_i \leftarrow L_i - a_{ir}^{(r-1)} / a_{rr}^{(r-1)} L_r$

Pour les $n-k$ lignes on obtient donc $(n-k)(n-k+1)$ opérations

- Pour la totalité des étapes on arrive donc à :

$$\begin{aligned} \sum_{k=1}^n (n-k)(n-k+1) &= \sum_{k=1}^n k^2 - (2n+1) \sum_{k=1}^n k + n(n+1) \sum_{k=1}^n 1 \\ &= \frac{n(n+1)(2n+1)}{6} - (2n+1) \frac{n(n+1)}{2} + n^2(n+1) \\ &= \frac{n^3}{3} - \frac{n}{3} \underset{n \rightarrow \infty}{\approx} \frac{n^3}{3} \end{aligned}$$



Nombre d'opérations pour résoudre un système linéaire

- Le terme de droite doit également être manipulé à chaque étape de la substitution. Un calcul similaire conduit à

$$\sum_{k=1}^n (n-k) = n \sum_{k=1}^n 1 - \sum_{k=1}^n k = n^2 - \frac{n(n+1)}{2} \underset{n \rightarrow \infty}{\approx} \frac{n^2}{2}$$

- Une fois le système triangularisé, il faut utiliser une substitution inverse pour obtenir la solution

Pour chacune des lignes k entre n et 1 cela nécessite :

- $n - k$ multiplications/soustractions
- 1 division par le pivot

Pour l'ensemble des lignes (des inconnues) on obtient :

$$\sum_{k=1}^n n - k + 1 = n^2 - \frac{n(n+1)}{2} + n = \frac{n^2}{2} + \frac{n}{2} \underset{n \rightarrow \infty}{\approx} \frac{n^2}{2}$$

Au final, **résoudre** un gros système linéaire « coûte »
environ **$n^3/3$** opérations



Nombre d'opérations pour inverser une matrice

- Inverser une matrice revient à résoudre n fois le même système linéaire avec n membres de droite différents (les n vecteurs de la base canonique):

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & : & 1 & 0 & 0 \\ 1 & 2 & 3 & : & 0 & 1 & 0 \\ 1 & 1 & 3 & : & 0 & 0 & 1 \end{bmatrix}}_A \rightarrow \rightarrow \rightarrow \underbrace{\begin{bmatrix} 1 & 0 & 0 & : & 3/2 & -1 & 1/2 \\ 0 & 1 & 0 & : & 0 & 1 & -1 \\ 0 & 0 & 1 & : & -1/2 & 0 & 1/2 \end{bmatrix}}_{I \quad A^{-1}}$$

- C'est la méthode de Gauss-Jordan
- Le compte des opérations est le suivant:

$$\underbrace{\frac{n^3}{3}}_{\text{élimination}} + \underbrace{n \times \frac{n^2}{2}}_{n \text{ termes de droite}} + \underbrace{n \times \frac{n^2}{2}}_{n \text{ substitutions inverses}} \underset{n \rightarrow \infty}{\approx} \frac{4}{3} n^3$$

qui peut être ramené à n^3 opérations en prenant en compte la présence de zéros. En tous les cas il est clair que d'un point de vue numérique:

Inverser une matrice pour **résoudre** un gros système linéaire n'est donc **pas efficace !!**

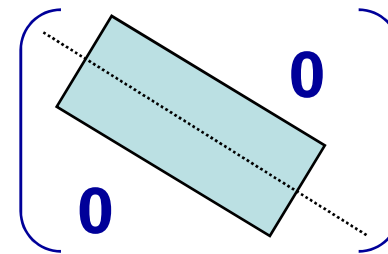


e- remarques

1 – un **+** de la méthode de GAUSS (ou LU): le calcul du déterminant de A est immédiat :

$$\det(A) = a_{11}^{(0)} \cdot a_{22}^{(1)} \cdots a_{nn}^{(n-1)}$$

3 – la méthode de GAUSS (ou LU) peut être optimisée pour les matrices bandes. En effet, la matrice est déjà partiellement triangularisée. Par exemple, pour une **matrice tri-diagonale**, on montre que le nombre d'opérations nécessaire pour résoudre le système peut être **réduit à $O(n)$** (au lieu de $O(n^3)$ dans le cas général ...)





Méthode de Cholesky

a- conditions d'utilisation

Il faut que la matrice A soit symétrique définie positive (SDP).

A symétrique : $a_{ij} = a_{ji}, \forall i, j$, (ou $A^T = A$)

A définie : $AX = 0 \Rightarrow X = 0$ ($\det A \neq 0$)

A positive : $\forall X \neq 0, X^T AX > 0$

En fait, ces conditions ne sont pas si restrictives car pour pouvoir résoudre $AX=B$, il faut que A^{-1} existe et donc que $\det A$ soit non nul. Si A n'est pas symétrique et positive, il suffit de réécrire le problème sous la forme :

$$\underbrace{A^T A}_{A^*} X = \underbrace{A^T B}_{B^*}$$

A^* est symétrique: $A^{*T} = (A^T A)^T = A^T A = A^*$

A^* est positive: $X^T A^* X = X^T A^T AX = (AX)^T (AX) = \|AX\|^2 > 0$



Méthode de Cholesky

- Lorsque \mathbf{A} est **symétrique** la factorisation **LDU** peut prendre une forme particulière : $\mathbf{A} = \mathbf{LDU} = \mathbf{A}^t = \mathbf{U}^t\mathbf{D}\mathbf{L}^t$ d'où $\mathbf{U}^t = \mathbf{L}$ et $\mathbf{L}^t = \mathbf{U}$ et donc :

$$\mathbf{A} = \mathbf{LDL}^t$$

- Par théorème, si \mathbf{A} est définie positive alors toutes ses **valeurs propres sont positives** et tous ses **pivots** sont **positifs**. On en déduit que \mathbf{D} peut s'écrire $\mathbf{D} = \Delta\Delta = \Delta\Delta^t$
- La décomposition $\mathbf{A} = \mathbf{LDL}^t$ donne alors: $\mathbf{A} = \mathbf{L}\Delta\Delta^t\mathbf{L}^t = \mathbf{L}\Delta(\mathbf{L}\Delta)^t$
- Au final, si les conditions SDP sont respectées alors on obtient la factorisation de Cholesky:

Si \mathbf{A} est SDP, alors il existe une matrice \mathbf{L} telle que : $\mathbf{A} = \mathbf{LL}^t$



- Une fois la factorisation LL^T effectuée, la méthode de résolution est identique à celle utilisée avec la factorisation LU :

$$A = LL^T \quad \boxed{A} = \boxed{L} \boxed{L^T}$$

L'équation $AX = B$ peut être réécrite sous la forme :

$$LL^T X = B$$

Si on pose : $Y = L^T X$ alors on peut résoudre en deux balayages successifs (c'est le fameux double balayage de Choleski !!) :



$$LY = B \quad \boxed{L} \boxed{Y} \downarrow = \boxed{B}$$

Le 1^{er} balayage donne Y

$$L^T X = Y \quad \boxed{L^T} \boxed{X} \uparrow = \boxed{Y}$$

Le 2^{ème} balayage donne X





- Comme pour la méthode d'élimination de Gauss (factorisation LU), la principale difficulté de la méthode est de déterminer les coefficients l_{ij} de la matrice \mathbf{L} à partir des coefficients a_{ij} de \mathbf{A} .
- Exemple : dans le cas d' un système 3×3 , on doit avoir :

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = \begin{pmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{pmatrix} \begin{pmatrix} l_{11} & l_{21} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{pmatrix}$$



Soit à résoudre le système non linéaire suivant :

$$\begin{cases} l_{11}^2 = a_{11} \\ l_{21}l_{11} = a_{21} \\ l_{31}l_{11} = a_{31} \\ l_{21}^2 + l_{22}^2 = a_{22} \\ l_{31}l_{21} + l_{32}l_{22} = a_{32} \\ l_{31}^2 + l_{32}^2 + l_{33}^2 = a_{33} \end{cases}$$

6 équations à 6 inconnues

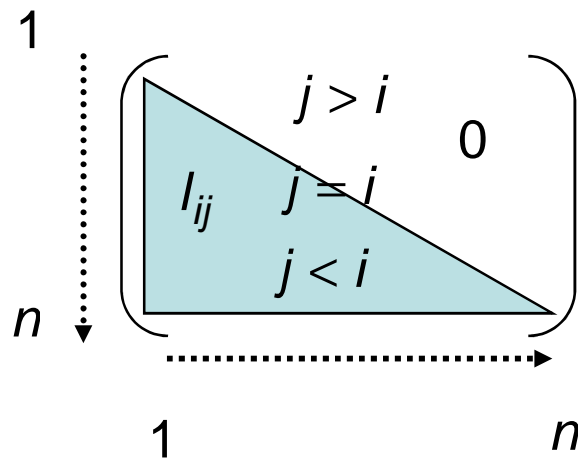
$$\begin{cases} l_{11} = \sqrt{a_{11}} \\ l_{21} = a_{21} / \sqrt{a_{11}} \\ l_{31} = a_{31} / \sqrt{a_{11}} \\ l_{22} = \sqrt{a_{22} - \frac{a_{21}^2}{a_{11}}} \\ l_{32} = \frac{a_{32} - \frac{a_{21}a_{31}}{a_{11}}}{\sqrt{a_{22} - \frac{a_{21}^2}{a_{11}}}} \\ l_{33} = \sqrt{a_{33} - \frac{a_{21}^2}{a_{11}} - \frac{a_{31}^2}{a_{11}}} \end{cases}$$

Pas de problème pour calculer les l_{ij}
 $i = 1, 2, 3$ car on sait que la diagonale
de L est positive...



Cas général :

$$\text{on a : } a_{ij} = \sum_{k=1}^n l_{ik} l_{kj}^T = \sum_{k=1}^n l_{ik} l_{jk}$$



or $l_{pq} = 0$ dès que $q > p$

donc, $l_{ik} = 0$ dès que $k > i$ et $l_{jk} = 0$ dès que $k > j$

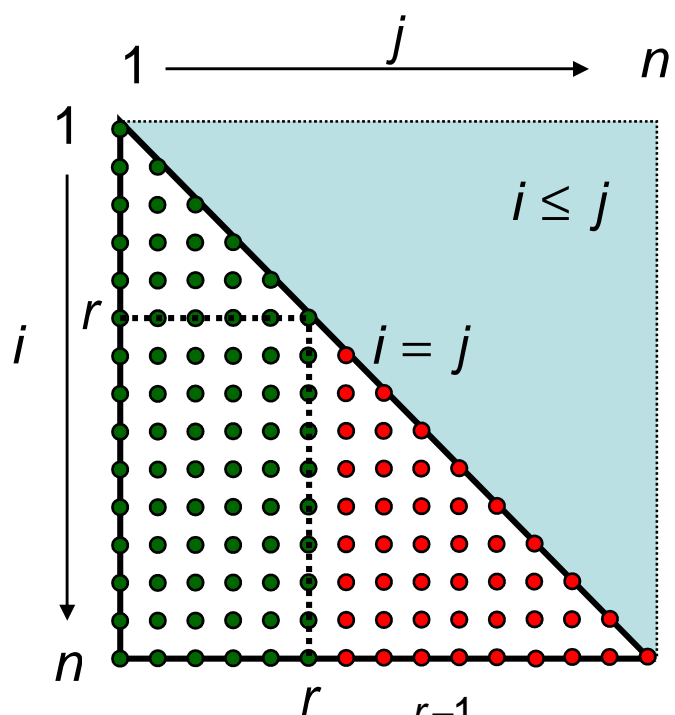
d'où
$$a_{ij} = \sum_{k=1}^{\min(i,j)} l_{ik} l_{jk}$$

Les indices i et j concernés correspondent au triangle inférieur :

$$1 \leq j \leq n$$

$$j \leq i \leq n$$

donc le $\min(i, j) = j$



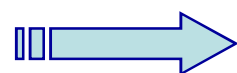
En suivant la méthode de résolution mise en place sur l'exemple 3×3, si l'on se place en $j = r$: tous les l_{ij} ($1 \leq j \leq r - 1$ et $j \leq i \leq n$) sont connus.

En fait les l_{ij} où $i < j$ sont nuls !!

$$\text{Or } a_{ij} = \sum_{k=1}^j l_{ik} l_{jk}$$

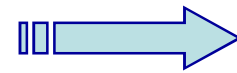
Si $j = r$, $a_{ir} = \sum_{k=1}^{r-1} l_{ik} l_{rk} + l_{ir} l_{rr}$, $j \leq i \leq n$

*si $i = j = r$, $a_{rr} = \sum_{k=1}^{r-1} l_{rk} l_{rk} + l_{rr}^2$



$$l_{rr} = \sqrt{\left(a_{rr} - \sum_{k=1}^{r-1} l_{rk}^2 \right)}$$

*si $i > r$, $a_{ir} = \sum_{k=1}^{r-1} l_{ik} l_{rk} + l_{ir} l_{rr}$



$$l_{ir} = \frac{1}{l_{rr}} \left(a_{ir} - \sum_{k=1}^{r-1} l_{ik} l_{rk} \right)$$



Quelques remarques:

1. La méthode de Cholesky est **moins générale** que la factorisation LU puisqu'elle n'est applicable qu'aux matrices SDP.
2. On montre que le nombre d'opérations nécessaires à la résolution d'un système linéaire de taille n est de l'ordre de $n^3/6$, soit environ **2 fois moins** que pour la méthode générale LU
3. Un autre avantage: la méthode de Cholesky est très robuste numériquement, **peu sensible aux erreurs** de troncature. Le pivotage est en pratique beaucoup moins nécessaire pour qu'il ne l'est pour l'élimination de Gauss.
4. Même divisé par 2, le nombre d'opérations reste proportionnel au **cube** de la taille du système à résoudre. Cela devient très problématique lorsque n est de l'ordre de 10^6 , ..., 10^9 comme on le rencontre en calcul intensif ... **Il faut trouver autre chose !**

b- généralisation aux systèmes $n \times n$



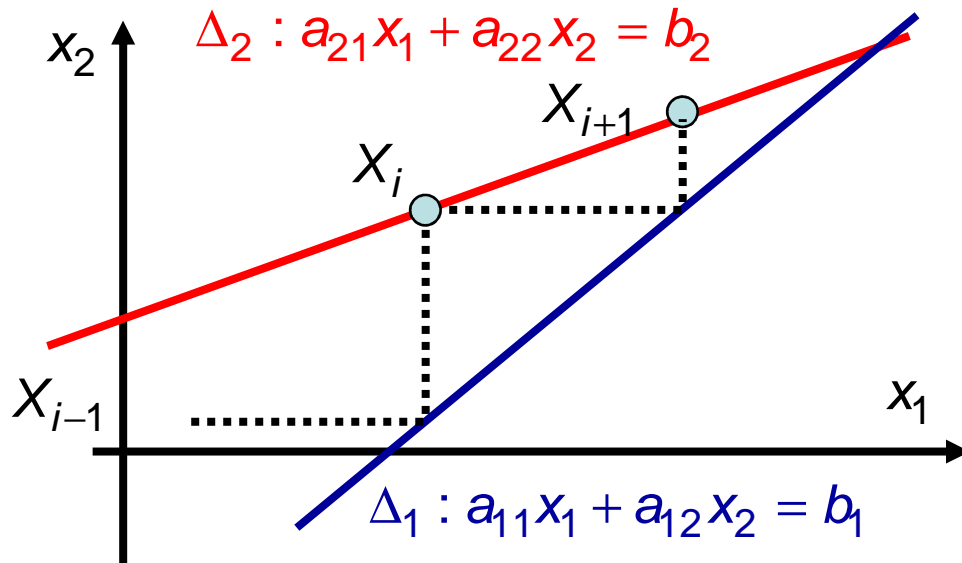
Convergence : $\|X^{(k+1)} - X^{(k)}\| < \varepsilon$

Do it yourself !!

Méthode lente, mais très facile à programmer



II.2- Méthode de Gauss-Seidel



a- Cas d'un système 2×2

$$\begin{cases} a_{11}x_1 + a_{12}x_2 = b_1 \\ a_{21}x_1 + a_{22}x_2 = b_2 \end{cases}$$



$$\begin{cases} x_1^{(i+1)} = \frac{1}{a_{11}} (b_1 - a_{12} x_2^{(i)}) \\ x_2^{(i+1)} = \frac{1}{a_{22}} (b_2 - a_{21} x_1^{(i+1)}) \end{cases}$$

On utilise, pour calculer x_2 , la valeur de x_1 réactualisée.

b- mise en œuvre dans le cas de systèmes $n \times n$

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \\ \dots\dots\dots \\ a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n = b_n \end{cases}$$

$$AX = B$$

Passage de $X^{(k)} \rightarrow X^{(k+1)}$



$$\left\{ \begin{array}{l} x_1^{(k+1)} = \frac{1}{a_{11}} \left(b_1 - \sum_{j=2}^n a_{1j} x_j^{(k)} \right) \\ \dots \text{ligne } i = 2 \dots \\ x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^n a_{ij} x_j^{(k)} \right) \\ \dots \text{ligne } i = n - 1 \dots \\ x_n^{(k+1)} = \frac{1}{a_{nn}} \left(b_n - \sum_{j=1}^{n-1} a_{nj} x_j^{(k+1)} \right) \end{array} \right.$$

$i = 1$



$i = n$

c- conditions de convergence

On va chercher à montrer que : $X^{(k+1)} = \mathcal{L} X^{(k)} + H$

On pose pour cela : $A = D - E - F$



$$A = \underbrace{\begin{bmatrix} a_{11} & 0 & \dots & 0 \\ 0 & a_{22} & \dots & \vdots \\ \vdots & \dots & \dots & 0 \\ 0 & \dots & 0 & a_{nn} \end{bmatrix}}_D + \underbrace{\begin{bmatrix} 0 & 0 & \dots & 0 \\ a_{21} & \dots & \dots & \vdots \\ \vdots & \dots & \dots & 0 \\ a_{n1} & \dots & a_{nn-1} & 0 \end{bmatrix}}_{-E} + \underbrace{\begin{bmatrix} 0 & a_{12} & \dots & a_{1n} \\ 0 & \dots & \dots & \vdots \\ \vdots & \dots & \dots & a_{n-1n} \\ 0 & \dots & 0 & 0 \end{bmatrix}}_{-F}$$

La méthode de Gauss-Seidel s'écrit avec ces notations :

$$\left\{ \begin{array}{l} a_{11}x_1^{(k+1)} = b_1 - \sum_{j=2}^n a_{1j}x_j^{(k)} \\ \dots\dots\dots \text{ligne } i = 2 \\ \sum_{j=1}^i a_{ij}x_j^{(k+1)} = b_i - \sum_{j=i+1}^n a_{ij}x_j^{(k)} \\ \dots\dots\dots \text{ligne } i = n-1 \\ \sum_{j=1}^n a_{nj}x_j^{(k+1)} = b_n \end{array} \right.$$

$$(D - E)X^{(k+1)} = FX^{(k)} + B$$

$$X^{(k+1)} = \underbrace{(D - E)^{-1}FX^{(k)}}_{\mathcal{L}} + \underbrace{(D - E)^{-1}B}_{\mathcal{H}}$$



Prenons alors comme vecteur X initial

$$X^{(0)} = \begin{bmatrix} 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix} \quad \text{alors :} \quad \left\{ \begin{array}{l} X^{(1)} = H \\ X^{(2)} = [1 + \mathcal{L}]H \\ \vdots \\ X^{(k+1)} = [1 + \mathcal{L} + \dots + \mathcal{L}^k]H \end{array} \right.$$

La suite des $X^{(k)}$ sera convergente si la suite matricielle converge

Théorème 1: *une condition **nécessaire et suffisante** de convergence d'une méthode itérative (ici G.-S.) est que le rayon spectral de la matrice \mathcal{L} associée soit inférieur à 1.*

$$\rho(\mathcal{L}) < 1 \quad \text{avec } \rho = \sup_{i=1, \dots, n} |\lambda_i| \quad \lambda_i \text{ valeur propre}$$

Remarque : peu d'intérêt en pratique car la recherche des valeurs propres est un problème plus complexe que la résolution du système



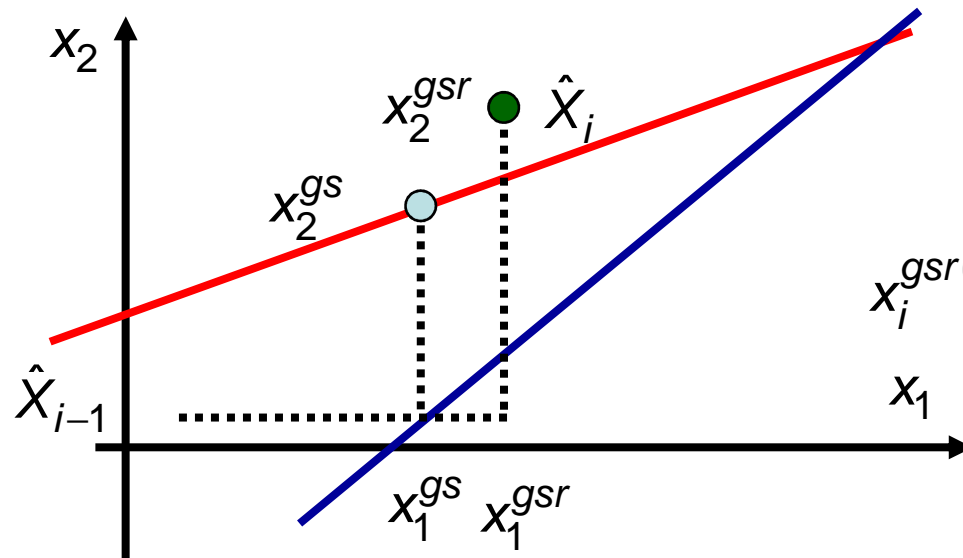
Théorème 2: une condition **suffisante** de convergence d'une méthode itérative (ici G.S.) est que la matrice **A** soit à « diagonale dominante ».

$$|a_{ii}| \geq \sum_{\substack{j=1 \\ j \neq i}}^N |a_{ij}| \text{ pour } 1 \leq i \leq n$$

II.3- Relaxation

$$\Delta_2 : a_{21}x_1 + a_{22}x_2 = b_2$$

$$\Delta_1 : a_{11}x_1 + a_{12}x_2 = b_1$$



Avec Gauss-Seidel ...

$$\begin{aligned} x_i^{gsr(k+1)} &= x_i^{gsr(k)} + \omega \left(x_i^{gs(k+1)} - x_i^{gsr(k)} \right) \\ &= \omega x_i^{gs(k+1)} + (1 - \omega) x_i^{gsr(k)} \end{aligned}$$



c- condition de convergence

Pour toute matrice \mathbf{A} , le rayon spectral de la matrice de la méthode de relaxation \mathcal{L}_ω est supérieur ou égal à $|1-\omega|$.

En d'autres termes, une condition **nécessaire** de convergence de la méthode de relaxation est que :

$$0 < \omega < 2$$

$\omega > 1$: sur relaxation

$\omega < 1$: sous relaxation

Attention !! Une condition nécessaire de convergence n'est pas une condition suffisante !

Le fait que ω vérifie la condition n'assure en rien la convergence de la méthode.

Par contre si ω ne vérifie pas la condition, alors là, on est sûr de diverger...

$\omega_{opt} ?$

III- Comparaison des méthodes directes (MD) et itératives (MI)



Rapidité d'exécution	Le compte des opérations est en général largement inférieur pour les MI que pour les MD, au moins pour les systèmes de grande taille [$O(n)$ contre $O(n^3)$].	
Facilité de mise en oeuvre	Bien que les MD ne soient pas très difficiles à mettre en oeuvre, les MI sont particulièrement faciles à programmer.	
Validité des résultats	Systèmes dégénérés	Les MI permettent, sous certaines conditions, d'obtenir une solution y compris pour les systèmes singuliers, ce qui n'est pas le cas pour les MD ($\det A = 0$)
	Convergence	La définition du critère d'arrêt est un problème complexe qui ne se pose que pour les MI.
	Précision	Les effets dévastateurs de la propagation des erreurs peuvent être évités que ce soit avec les MD ou avec les MI. Evidemment, pour les systèmes mal conditionnés, le problème peut ne pas avoir de bonne solution.
Conclusion	En général, les MD sont à préférer aux MI lorsque le choix est possible (systèmes de taille modérée ($O(10^5)$)): elles sont plus fiables et l'on sait que l'on obtiendra la solution à coup sûr. Toutefois, les MI sont souvent les seules possibles pour la résolution de systèmes issu de la discrétisation d'EDP Ces systèmes sont très creux et de taille potentiellement énorme ($O(10^8)$).	

Quelques références à consulter sans modération



1. *Analyse numérique matricielle appliquée à l'art de l'ingénieur*
Patrick Lascaux, Raymond Théodor, 1. Méthodes directes
2. *Linear Algebra and its applications*, Fourth Edition, Gilbert Strang
3. *Numerical Recipes. The art of scientific computing*, W. Press, S. Teukolsky, W. Wetterling and B. Flannery