GM3 -Analyse Numérique I

Timothée Schmoderer

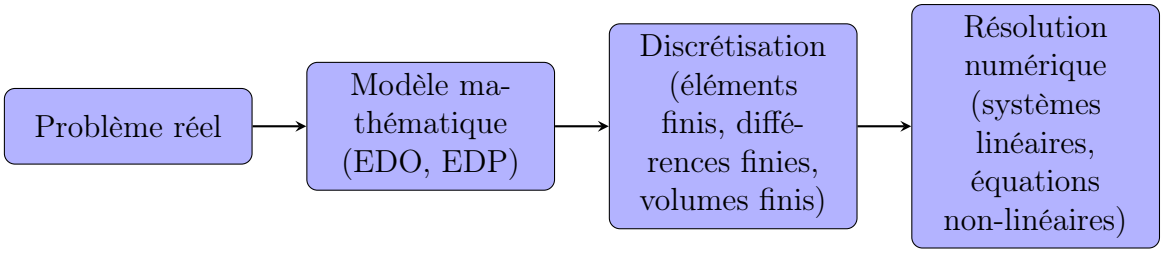
GM3 - Analyse Numérique I

Timothée Schmoderer *

Année 2021 - 2022

Introduction

Ce cours fait partie du module MMSN (Modélisation Mathématique et Simulation Numérique). La *Modélisation Mathématique* consiste à formuler un problème concret de la physique, de la biologie, de la mécanique etc. sous forme mathématique : équations différentielles ordinaires, équations aux dérivées partielles, système linéaire etc.).



Exemple (Équation des Ondes)**.** L'équation des ondes décrit la propagation des ondes ou les phénomènes de vibrations. Par exemple, en une dimension d'espace, elle est un modèle pour étudier les vibration d'une corde tendue. Au repos, la corde occupe un segment $I = [0, 1]$ de $\mathbb{R}$; sous l'action d'une force d'intensité f, elle se déforme et son déplacement normal est noté u (voir la figure). On suppose que la

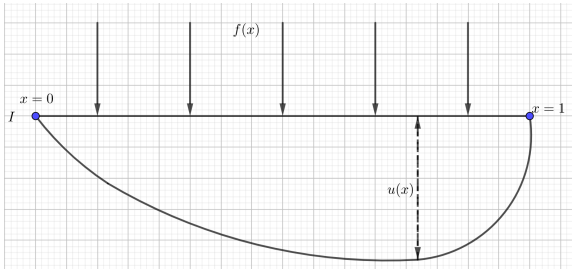


FIGURE 1 – Déplacement d'une corde élastique à t fixé.

corde est fixée sur ses bords (conditions aux limites de Dirichlet). L'équation des

*Ces notes sont essentiellement inspirées du cours de Maria Kazakova et d'Antoine Tonnoir. Elles sont encore en construction, pour toutes remarques timothee.schmoderer@insa-rouen.fr

ondes dont u est solution est donnée par

$$\begin{cases} \frac{\partial^2 u}{\partial t^2} - \frac{\partial^2 u}{\partial x^2} = f(t,x) & \text{dans} \quad \mathbb{R}_*^+ \times I \\ u(t, x=0) = u(t, x=1) = 0 & \text{pour tout } t \\ u(t=0,\cdot) = u_0(\cdot) & \text{dans} \quad I \\ \frac{\partial u}{\partial t}(t=0,\cdot) = u_1(\cdot) & \text{dans} \quad I \end{cases} \tag{1}$$

Remarquons qu'il s'agit d'une équation du deuxième ordre en temps et qu'il faut donc deux conditions initiales pour u. ▼

Quant à la *Simulation Numérique*, elle permet d'obtenir une solution à un problème mathématique à l'aide d'un ordinateur. Cette approche nous fournit une compréhension d'un phénomène et permet de prédire le comportement des objets en question, par exemple, une vague côtière, une onde électromagnétique ou élément de construction d'un bâtiment ou d'un avion lors de sa conception.

(a) Équation d'ondes 2D

(b) Simulation vague

(c) Simulation Éolienne

(d) Mode de vibration 0

(e) Mode de vibration 2

(f) Mode de vibration 3

FIGURE 2 – Exemple de simulations numériques de modèles mathématiques simples

L'**analyse numérique** considère à la fois la construction de différents algorithmes de résolution ainsi que leurs propriétés permettant de conclure si l'algorithme "marche bien" : précision, stabilité et "coût" de calcul (mémoire, temps de calcul). Les approches pour passer d'un modèle mathématique (continu) à un modèle de simulation opérationnel ("discrétisation")' sont considérées dans les cours d'Éléments finis, Différence finies ou méthodes spectrales (GM4, GM5). Une fois la bonne discrétisation adoptée, nous sommes amenés souvent à résoudre des systèmes linéaires (parfois plusieurs milliards d'équations à plusieurs milliards d'inconnues).

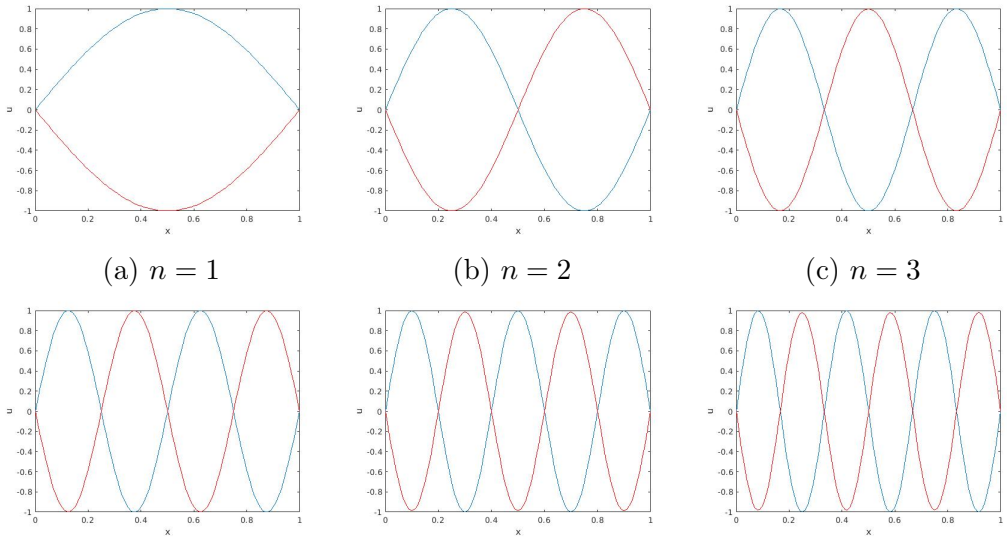
Exemple (Équation des Ondes (suite))**.** Le schéma de discrétisation implicite donne une relation entre l'approximation de u au pas de temps suivant, u^{n+1}, en fonction des deux étapes précédentes, u^n et u^{n-1}, (voir GM4 pour les détails)

$$-u_{i+1}^{n+1} + 2\left(1 + \frac{1}{r^2}\right) u_i^{n+1} - u_{i-1}^{n+1} = \frac{4}{r^2} u_i^n + u_{i+1}^{n-1} - 2\left(1 + \frac{1}{r^2}\right) u_i^{n-1} + u_{i-1}^{n-1}.$$

Ce schéma se traduit par la résolution d'un système linéaire à chaque pas de temps :

$$\begin{pmatrix} 2+\frac{2}{r^2} & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2+\frac{2}{r^2} \end{pmatrix} u^{n+1} = \frac{4}{r^2} \begin{pmatrix} 1 & 0 & & \\ 0 & \ddots & \ddots & \\ & \ddots & \ddots & 0 \\ & & 0 & 1 \end{pmatrix} u^n - \begin{pmatrix} 2+\frac{2}{r^2} & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2+\frac{2}{r^2} \end{pmatrix} u^{n-1}$$

Ce schéma est très facile à implémenter et on obtient alors les résultats numériques de la figure 3 illustrant les différents mode de vibration d'une corde. ▼



(a) $n = 1$ (b) $n = 2$ (c) $n = 3$

(d) $n = 4$ (e) $n = 5$ (f) $n = 6$

FIGURE 3 – Ondes stationnaires dans une corde. La fréquence fondamentale et 5 harmoniques sont illustrées.

L'objectif du cours est de donner une réponse numérique au problème suivant :

> Pour une matrice donnée à coefficients constants A et pour un vecteur colonne donné $\mathbf{b}$, trouver un vecteur $\mathbf{x}$ tel que $A\mathbf{x} = \mathbf{b}$.

Ressources à utiliser :

- Quarteroni, A., Sacco, R. and Saleri, F., 2010. *Numerical mathematics* (Vol. 37). Springer Science & Business Media.
- Burden, R.L., Faires, J.D. and Burden, A.M., 2010. *Numerical analysis : Cengage Learning*. Brooks/Cole.

Programme du cours :

- ☞ Rappel d'Algèbre Linéaire (1,5 séances)
- ☞ Méthodes directes (3.5 séances)
- ☞ Méthodes itératives (2 séances)
- ☞ IS
- ☞ Méthodes itératives suite (3 séances)
- ☞ Résolution des équations non linéaires (4 séances)
- ☞ DS

Table des matières

1 Résolution des systèmes linéaires **5**

1.1 Rappels d'algèbre linéaire . . . 5
1.1.1 Espaces vectoriels . . . 5
1.1.2 Matrices $\mathbb{M}_{n\times p}(\mathbb{K})$ . . . 6
1.1.3 Matrices et applications linéaires . . . 8
1.1.4 Déterminant et ses propriétés . . . 11
1.1.5 Produit scalaire . . . 12
1.1.6 Matrice symétrique définie positive . . . 13
1.1.7 Normes et conditionement . . . 13
1.2 Méthodes directes : Élimination de Gauss et ses variantes . . . 17
1.2.1 Algorithmes de descente et remontée . . . 17
1.2.2 Méthode de Gauss : Réduction . . . 20
1.2.3 Méthode de Gauss : factorisation LU . . . 24
1.2.4 Méthode de Gauss : conditions sur A . . . 25
1.2.5 Méthode Choleski . . . 28
1.2.6 Factorisation QR : orthogonalisation de Gram-Schmidt . . . 30
1.2.7 Méthode de Householder . . . 33
1.3 Rappels d'algèbre linéaire (suite) . . . 36
1.3.1 Éléments de théorie spectrale . . . 36
1.3.2 Série de matrices . . . 39
1.4 Méthodes itératives . . . 42
1.4.1 Méthode de Jacobi et de Gauss-Seidel . . . 45
1.4.2 Méthode de descente . . . 47
1.4.3 Les méthodes de Krylov . . . 50
1.4.4 Méthode de gradient conjugué . . . 52
1.4.5 Generalized Minimal Residual (GMRes) . . . 55
1.4.6 Principe du préconditionnement . . . 58
1.5 Méthode directe ou itérative ? . . . 58

2 Résolution des équations non linéaires **59**

2.1 Méthode de point fixe . . . 60
2.2 Vitesse de convergence . . . 63
2.3 Méthode de Newton . . . 64
2.4 Méthode de la fausse position . . . 66
2.5 Conclusion . . . 67

1 Résolution des systèmes linéaires

Considérons un systèmes d'équations linéaires de la forme $A\mathbf{x} = \mathbf{b}$ avec A une matrice inversible connue de dimension $n \times n$, $\mathbf{b}$ un vecteur connu et $\mathbf{x}$ le vecteurs des inconnues :

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}.$$

Il existe deux grandes familles de méthodes de résolution :

- Les **méthodes directes** qui permettent de résoudre le système exactement soit par triangularisation, soit par factorisation de la matrice A. Les principales méthodes sont :
  - Le pivot de Gauss,
  - La factorisation LU,
  - La factorisation de Cholesky,
  - Les factorisations de Householder et QR.
- Les **méthodes itératives** qui introduisent une notion de convergence vers la solution. Les principales méthodes sont :
  - Méthode de Jacobi,
  - Méthode de Gauss-Seidel,
  - Méthode du gradient.

Nous détaillerons les avantages et inconvénients de chacune de ces méthodes, mais avant toutes choses voici quelques rappels d'algèbre linéaire (M4).

1.1 Rappels d'algèbre linéaire

Dans cette partie nous rappelons les éléments de base d'algèbre linéaire que nous utiliserons dans le reste du cours.

1.1.1 Espaces vectoriels

Commençons par définir l'espace dans lequel nous travaillerons toujours.

Définition 1.1 (Espace vectoriel)**.** On dit que V est un espace vectoriel sur un corps $\mathbb{K}$ ($\mathbb{R}$ ou $\mathbb{C}$) si c'est un ensemble non-vide muni de deux opérations : l'addition ($+ : V^2 \to V$) et la multiplication de $\mathbb{K}$ ($\cdot : \mathbb{K} \times V \to V$), qui vérifient les propriétés suivantes :

(i) l'**addition** est *commutative* et *associative* :
$\forall\, v_1, v_2, v_3 \in V,\ v_1 + v_2 = v_2 + v_1 \quad$ et $\quad (v_1 + v_2) + v_3 = v_1 + (v_2 + v_3)$;

(ii) il existe un *élément neutre* (vecteur nul) :
$\exists\, 0_V \in V : \forall v \in V,\ v + 0_V = v$;

(iii) il existe un *élément opposé* :
$\forall\, v \in V\ \exists (-v) \in V :\ v + (-v) = 0_V$;

(iv) la **multiplication** est *associative* :
$\forall\, \lambda, \mu \in \mathbb{K} \quad \forall v \in V\ (\lambda\mu) \cdot v = \lambda(\mu \cdot v)$;

(v) on a $\forall\, v \in V, \quad 1 \cdot v = v, \quad$ et $\quad 0 \cdot v = 0_V$, où 0, 1 sont les éléments neutre additif et neutre multiplicatif de corps $\mathbb{K}$

(vi) les lois de distributivité suivantes sont vérifiées :
$\forall\, \lambda \in \mathbb{K} \quad \forall\, v_1, v_2 \in V\ \lambda(v_1 + v_2) = \lambda v_1 + \lambda v_2$
$\forall\, \lambda, \mu \in \mathbb{K} \quad \forall\, v \in V\ (\lambda + \mu) \cdot v = \lambda \cdot v + \mu \cdot v$

Exemple.

▼

Définition 1.2 (Sous espace vectoriel)**.** On dit qu'une partie non-vide W du $\mathbb{K}$-ev V est un sous-espace vectoriel si et seulement si W est un espace vectoriel sur $\mathbb{K}$.

Définition 1.3. La famille $(v_k)_{1 \le k \le n}$ de vecteurs de V est une *base* de V si

(i) $(e_k)_{1 \le k \le n}$ est une *famille libre* : $\sum_{k=1}^{n} \lambda_k e_k = 0 \iff \lambda_k = 0\ \forall\, k \in [1, \dots, n]$,

(ii) $(e_k)_{1 \le k \le n}$ est une *famille génératrice* : $\forall\, v \in V\ \exists\, (\lambda_k)_{1 \le k \le n} \in \mathbb{K} : v = \sum_{k=1}^{n} \lambda_k e_k$.

Propriété 1.4. Soit V est un e.v. qui admet une base de n vecteurs, alors toute famille libre de vecteurs de V compte au plus n éléments. Ainsi la dimension de V, noté $\dim(V)$, qui est définie comme le nombre de vecteurs de sa base, est égale à n. Si, au contraire, $\forall\, n \in \mathbb{N}$ il existe une famille libre de V, V est un espace de dimension infinie.

1.1.2 Matrices $\mathbb{M}_{n \times p}(\mathbb{K})$

Définition 1.5. On appelle une matrice un tableau des éléments de $\mathbb{K}$:

$$A = \begin{pmatrix} a_{11} & \dots & \dots & a_{1p} \\ & \ddots & \ddots & \\ a_{n1} & \dots & \dots & a_{np} \end{pmatrix} \quad \text{avec } a_{ij} \in \mathbb{K}.$$

Si $n = p$, alors A est une matrice carrée.

Dans ce cours, on s'intéressera à l'espace des matrices de taille $n \times p$ (n est le nombre de lignes, p est le nombre de colonnes) à coefficients dans $\mathbb{K}$, noté $\mathbb{M}_{n \times p}(\mathbb{K})$. On munit l'espace $\mathbb{M}_{n \times p}(\mathbb{K})$ de deux opérations de base :
Somme :
$\forall\, A \in \mathbb{M}_{n \times p}(\mathbb{K}), \forall\, B \in \mathbb{M}_{n \times p}(\mathbb{K})$, $A + B \in \mathbb{M}_{n \times p}(\mathbb{K})$ est une matrice avec des éléments :

Multiplication par un scalaire :
$\forall\, \lambda \in \mathbb{K}, \forall\, A \in \mathbb{M}_{n \times p}(\mathbb{K})$, $\lambda A \in \mathbb{M}_{n \times p}(\mathbb{K})$ est une matrice avec des éléments :

Exercice. Montrez que $\mathbb{M}_{n\times p}(\mathbb{K})$ avec les opérations définies ci-dessus est un espace vectoriel sur $\mathbb{K}$, donnez sa dimension et une base.

La structure de l'ensemble des matrices est plus riche que celle d'un simple espace vectoriel, car on peut définir le produit de deux matrices.
Produit matriciel : On peut définir le produit de $A = (a_{ij}) \in \mathbb{M}_{n\times p}(\mathbb{K})$ et $B = (b_{ij}) \in \mathbb{M}_{p\times m}(\mathbb{K})$ par :

avec $AB \in \mathbb{M}_{n\times m}(\mathbb{K})$

Remarque. En général, le produit matriciel n'est pas commutatif $AB \neq BA$. De plus, même si AB a un sens, rien ne garantit que BA en ait un.

Matrice transposée : On définit $\forall\, A \in \mathbb{M}_{n\times p}(\mathbb{R})$ sa matrice transposée, notée A^t, telle que $A^t \in \mathbb{M}_{p\times n}(\mathbb{R})$: $a^t_{ij} := a_{ji}$:

$$A = \begin{pmatrix} a_{11} & \dots & a_{1p} \\ & \ddots & \\ a_{n1} & \dots & a_{np} \end{pmatrix} \qquad A^t = \begin{pmatrix} a_{11} & \dots & a_{n1} \\ & \ddots & \\ a_{1p} & \dots & a_{np} \end{pmatrix}$$

Généralisation à $\mathbb{C}$: On définit une matrice adjointe A^* (ou A^H) définie par

$$a^H_{ij} := \overline{a_{ji}} := \Re(a_{ji}) - i\Im(a_{ji})$$

Proposition 1.6. $\forall\, A \in \mathbb{M}_{n\times p}(\mathbb{R})$ *les propriétés suivantes sont vérifiées :*

Définition 1.7 (Inverse d'une matrice)**.** Soit $A \in \mathbb{M}_{n\times n}(\mathbb{R})$ matrice $B \in \mathbb{M}_{n\times n}(\mathbb{R})$ est une matrice inverse de A si $AB = BA = I$ (ou I est la matrice identité). Si B existe, A est appelée inversible, et on note $B = A^{-1}$.

Proposition 1.8. *Soit $A \in \mathbb{M}_{n\times n}(\mathbb{R})$ et $B \in \mathbb{M}_{n\times n}(\mathbb{R})$ deux matrices inversibles. Les propriétés suivantes sont vérifiées :*

Démonstration.

□

Proposition 1.9. *Soit $A \in \mathbb{M}_{n\times n}(\mathbb{R})$ est inversible, alors $(A^{-1})^t = (A^t)^{-1} =: A^{-t}$*

Définition 1.10. On appelle matrice A *symétrique* si $A = A^t$, et *antisymétrique* si $A = -A^t$. Enfin, matrice A est appelée *orthogonale* si $A^tA = AA^t = I$ (donc si $A^{-1} = A^t$).

Exercice. Montrez que $A = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$ est inversible, donnez son inverse, et montrez qu'elle est orthogonale.

1.1.3 Matrices et applications linéaires

Dans cette sous-section on établit qu'une matrice correspond à une application linéaire. Cela nous permet (entre autre) de définir le noyau, l'image et le rang d'une matrice.

Définition 1.11. $\mathcal{A}$ est une application linéaire $\mathbb{R}^n \to \mathbb{R}^n$ si :

$$\forall(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^n \times \mathbb{R}^n \quad \forall(\alpha, \beta) \in \mathbb{R}^2 \quad \mathcal{A}(\alpha\mathbf{x} + \beta\mathbf{y}) = \alpha\mathcal{A}(\mathbf{x}) + \beta\mathcal{A}(\mathbf{y})$$

Proposition 1.12. *Soit $\mathcal{A}$ est une application linéaire : $\mathbb{R}^n \to \mathbb{R}^n$. Alors, $\exists! A \in \mathbb{M}_{n\times n}(\mathbb{R})$ tel que*

$$\forall\, \mathbf{x} \in \mathbb{R}^n : \quad \mathcal{A}(\mathbf{x}) = A\mathbf{x}.$$

Démonstration.

□

Remarque.

1. La matrice A est définie uniquement pour une base fixée $(e_k)_{1\leq k\leq n}$. Si A est la matrice associé à $\mathcal{A}$ dans la base $(e_k)_{1\leq k\leq n}$ et $P : e \to \tilde{e}$, alors $\tilde{A} = PAP^{-1}$ est la matrice dans la base $(\tilde{e}_k)_{1\leq k\leq n}$.
2. On peut vérifier que $\mathcal{A} \to A$ est une isomorphisme de $\mathcal{L}(\mathbb{R}^n, \mathbb{R}^n)$ vers $\mathbb{M}_{n\times n}(\mathbb{R})$.
3. On peut généraliser cette proposition aux matrices rectangulaire de $\mathbb{M}_{n\times p}(\mathbb{R})$ Dans ce cas $A \in \mathbb{M}_{n\times p}(\mathbb{R})$ est associée à une application $\mathbb{R}^p \to \mathbb{R}^n$.

Exemple. L'espace des polynôme de degré au plus 3 sur $\mathbb{R}$, noté $\mathbb{R}_3[X]$, est un espace vectoriel. On définit l'application linéaire $\mathcal{A} : \mathbb{R}_3[X] \to \mathbb{R}_3[X]$ par

$$\mathcal{A}(P) = X^2P'' + P(1).$$

Dans la base canonique $\{1, X, X^2, X^3\}$ de $\mathbb{R}_3[X]$ l'application $\mathcal{A}$ est représentée par :

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 6 \end{pmatrix}$$

▼

Définition 1.13 (Image et Noyau)**.** Soit $\mathcal{A}$ est une application linéaire $\mathbb{R}^n \to \mathbb{R}^n$ et A est sa matrice associée :

(i) L'*image* de $\mathcal{A}$ est l'ensemble :

$$\mathrm{Im}(\mathcal{A}) = \{\mathbf{y} \in \mathbb{R}^n \mid \exists x \in \mathbb{R}^n \ \mathcal{A}(\mathbf{x}) = \mathbf{y}\} = \{\mathbf{y} \in \mathbb{R}^n \mid \exists x \in \mathbb{R}^n \ A\mathbf{x} = \mathbf{y}\},$$

(ii) Le *noyau* de $\mathcal{A}$ est l'ensemble :

$$\mathrm{Ker}(\mathcal{A}) = \{\mathbf{x} \in \mathbb{R}^n \mid \mathcal{A}(\mathbf{x}) = \mathbf{0}\} = \{\mathbf{x} \in \mathbb{R}^n \mid A\mathbf{x} = \mathbf{0}\}.$$

Exercice. Déterminer $\mathrm{Im}(A)$ et $\mathrm{Ker}(A)$ pour les applications définies par les matrices :

$$A_1 = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}, \quad \text{et} \quad A_2 = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}.$$

Définition 1.14. Une application linéaire $\mathcal{A} : \mathbb{R}^n \to \mathbb{R}^n$ est dite
injective si $\mathrm{Ker}(A) = \{\mathbf{0}\}$,
surjective si $\mathrm{Im}(A) = \mathbb{R}^n$,
bijective si elle est injective et surjective.

Définition 1.15. $A \in \mathbb{M}_{n\times n}(\mathbb{R})$ est une matrice inversible si l'application associée est bijective (cette définition est équivalente à : $\exists A^{-1} \in \mathbb{M}_{n\times n}(\mathbb{R}) : A^{-1}A = AA^{-1} = I$).

Theorem 1.16 (du rang)**.** *Pour toute application linéaire* $\mathcal{A} : \mathbb{R}^n \to \mathbb{R}^n$ *on a* $\dim(Ker(A)) + \dim(Im(A)) = n$.

Démonstration.

□

Corollaire 1.17. *Si une application linéaire A est injective, donc* $\dim(Ker(A)) = 0$ *alors* $\dim(Im(A)) = n$*, et donc A est surjective, et ainsi A est bijective. Ou encore, si A est surjective, alors elle est bijective.*

Définition 1.18 (Rang)**.** La dimension de $\mathrm{Im}(A)$ correspond au rang de la matrice $\dim(\mathrm{Im}(A)) = \mathrm{rg}(A)$.

Propriété 1.19. Soit $A \in \mathbb{M}_{n\times n}(\mathbb{R})$, les propriétés suivantes sont équivalentes :

(i) A est inversible

(ii) $\mathrm{Ker}(A) = \{\mathbf{0}\}$

(iii) $\mathrm{rg}(A) = n$

(iv) les lignes (et les colonnes) de A sont linéairement indépendants.

(v) $\det(A) \neq 0$

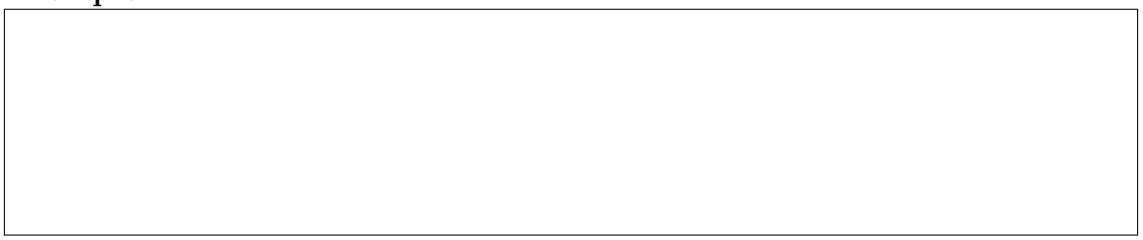
1.1.4 Déterminant et ses propriétés

Définition 1.20. Pour toute $A = (a_{ij}) \in \mathbb{M}_{n\times n}(\mathbb{R})$ on définie son déterminant (noté $\det(A)$)

$$\det(A) = \sum_{j=1}^{n} (-1)^{i+j} a_{ij} \det(M_{i,j}) = \sum_{i=1}^{n} (-1)^{i+j} a_{ij} \det(M_{i,j})$$

où $M_{i,j}$ est la matrice A à laquelle on a enlevé la ligne i et la colonne j, son déterminant $\det(M_{i,j})$ est appelé mineur.

Exemple.



Propriété 1.21. Soit $A \in \mathbb{M}_{n\times n}(\mathbb{R})$

$$\det(A) = \det(A^t), \quad \det(AB) = \det(A)\det(B), \quad \det(A^{-1}) = \frac{1}{\det(A)},$$
$$\det(A^*) = \overline{\det(A)}, \quad \det(\alpha A) = \alpha^n \det(A), \ \forall\, \alpha \in \mathbb{K}$$

Propriété 1.22. Soit $A \in \mathbb{M}_{n\times n}(\mathbb{R})$ alors on a les propriétés suivantes :

(i) Si A est telle que deux colonnes ou deux lignes coïncident, alors $\det(A) = 0$.

(ii) Si l'on permute deux lignes (ou deux colonnes) de A, alors le signe du déterminant change.

(iii) Si $A = \mathrm{diag}(d_1, \ldots, d_n)$, alors le déterminant est $\det(A) = \Pi_{i=1}^{n} d_i$.

(iv) A est inversible si et seulement si $\det(A) \neq 0$.

(v) Si A est inversible alors $A^{-1} = \frac{1}{\det(A)} C$, où $c_{ij} = (-1)^{i+j} \det(M_{j,i})$.

Theorem 1.23 (Formule de Cramer)**.** *Soit $A \in \mathbb{M}_{n\times n}(\mathbb{R})$ est inversible, et $\mathbf{b} \in \mathbb{R}^n$, alors la solution $\mathbf{x} \in \mathbb{R}^n$ d'un système linéaire $A\mathbf{x} = \mathbf{b}$ est définie par la formule de Cramer :*

où $A_i(b)$ est la matrice carrée formée en remplaçant la k-ème colonne de A par le vecteur colonne b.

Démonstration.

□

1.1.5 Produit scalaire

Définition 1.24. Produit scalaire sur un $\mathbb{K}$-e.v. V est défini comme une application $(\cdot,\cdot) : V \times V \to \mathbb{K}$ qui est

(i) *linéaire* par rapport aux vecteurs de V : $\forall\, v_1, v_2, v_3 \in V,\ \forall \gamma, \lambda \in \mathbb{K}\ (\gamma v_1 + \lambda v_2, v_3) = \gamma(v_1, v_3) + \lambda(v_2, v_3)$;

(ii) *hermitienne* $\forall\, v_1, v_2 \in V,\ (v_1, v_2) = \overline{(v_2, v_1)}$;

(iii) *définie positive* $\forall\, v \in V,\ (v, v) \geq 0$ et $(v, v) = 0 \iff v = 0_V$.

Exemple.

▼

Proposition 1.25. *Pour toute matrice* $A \in \mathbb{M}_{n\times p}(\mathbb{K})$ *et* $\forall\, \mathbf{x}, \mathbf{y} \in \mathbb{K}^n$ *:*

$$(A\mathbf{x}, \mathbf{y}) = (\mathbf{x}, A^*\mathbf{y}).$$

En particulier, la proposition 1.25 pour toute matrice unitaire $Q \in \mathbb{M}_{n\times n}(\mathbb{K})$ (c'est à dire $Q : Q^{-1} = Q^*$) donne $(Q\mathbf{x}, Q\mathbf{y}) = (\mathbf{x}, Q^*Q\mathbf{y})$ et on obtient

Propriété 1.26. Une matrice unitaire préserve le produit scalaire euclidien : $\forall\, Q \in \mathbf{U}(n)$, on a $(Q\mathbf{x}, Q\mathbf{y}) = (\mathbf{x}, \mathbf{y})$.

1.1.6 Matrice symétrique définie positive

Définition 1.27. $A \in \mathbb{M}_{n\times n}(\mathbb{K})$ est définie positive si $\forall\, x \in \mathbb{R}^n \backslash \{\mathbf{0}\} \quad (Ax, x) > 0$. Si l'inégalité est large, on appelle la matrice A semi-définie positive.

Exemple.

▼

Mais les matrices symétriques définies positives ont une propriété très utile :

Theorem 1.28. *Une matrice symétrique définie positive est inversible.*

Démonstration.

□

1.1.7 Normes et conditionement

Définition 1.29 (Norme)**.** Une norme d'une matrice est une application $\|\cdot\| : \mathbb{M}_{n\times n}(\mathbb{R}) \to \mathbb{R}^+$ tel que

(i) $\|A\| \geq 0$ et $\|A\| = 0 \iff$ A est la matrice nulle ;

(ii) $\|\alpha A\| = |\alpha| \, \|A\| \; \forall \alpha \in \mathbb{R}, \forall A \in \mathbb{M}_{n\times n}(\mathbb{R})$;

(iii) $\|A + B\| \leq \|A\| + \|B\| \; \forall A \in \mathbb{M}_{n\times n}(\mathbb{R}), \forall B \in \mathbb{M}_{n\times n}(\mathbb{R})$;

Exercice. Le déterminant est-il une norme sur l'ensemble des matrices inversibles ?

Les normes d'intérêt pratique pour ce cours ont deux propriétés en plus :

(i) Une norme matricielle est *compatible* à une norme vectorielle $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}^+$ si $\forall\, x \in \mathbb{R}^n \quad \|Ax\| \leq \|A\| \, \|x\|$ (Remarque. Ici on utilise la même notation $\|\cdot\|$ pour deux objets différents)

(ii) Une norme matricielle est *sous-multiplicative* si

Remarque. Il existe des normes matricielles qui ne sont pas sous-multiplicative, par exemple la norme infinie sur $\mathbb{M}_{2\times 2}(\mathbb{R})$ (i.e. l'application qui à une matrice associe le max de la valeur absolue de ses coefficients). **Exercice.** Montrez que c'est une norme. Trouvez 2 matrices pour lesquelles l'inégalité de sous-multiplicativité n'est pas vérifiée.

Si une norme matricielle est sous-multiplicative, il existe toujours une norme vectorielle compatible. Rappelons les normes vectorielles classiques sur $\mathbb{R}^n$ $\|\cdot\|_q$, $q = (1, 2, \infty)$:

$$\text{Soit } \mathbf{x} \in \mathbb{R}^n : \quad \|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i|, \quad \|\mathbf{x}\|_2 = \left(\sum_{i=1}^{n} |x_i|^2 \right)^{1/2}, \quad \|\mathbf{x}\|_\infty = \max_{i\in\{1,n\}} |x_i|.$$

On définit la norme q d'une matrice A à la façon suivante :

Définition 1.30 (Norme matricielle subordonnée à une norme vectorielle $\|\cdot\|_q$)**.**

$$\|A\|_q = \sup_{\|\mathbf{x}\|\neq 0} \frac{\|A\mathbf{x}\|_q}{\|\mathbf{x}\|_q}$$

On peut démontrer que cette application définie une norme compatible avec $\|\cdot\|_q$ et sous-multiplicative :

$$\|A\mathbf{x}\|_q \leq \|A\|_q \, \|\mathbf{x}\|_q, \quad \|AB\|_q \leq \|A\|_q \, \|B\|_q .$$

Enfin une norme subordonnée vérifie toujours $\|I\|_q = 1$.

Proposition 1.31. *(cf TD) Soit $A \in \mathbb{M}_{n\times n}(\mathbb{R})$, on peut définir les normes subordonnées :*

(i) $\|A\|_q = \sup_{\|\mathbf{x}\|_q=1} \|A\mathbf{x}\|_q$

(ii) $\|A\|_1 = \max_{1\leq j\leq n} \sum_{i=1}^{n} |a_{ij}| \qquad \|A\|_\infty = \max_{1\leq i\leq n} \sum_{j=1}^{n} |a_{ij}|$

Définition 1.32. On appelle *conditionnement* d'une matrice carré inversible le nombre

Le conditionnement d'une matrice mesure la "difficulté" à inverser une matrice.

Remarque. Le conditionnement nous permet d'étudier la *sensibilité* du problème, c'est à dire l'influence de petites perturbations des données sur la solution du problème. Considérons un système linéaire :

$$A\mathbf{x} = \mathbf{b}, \quad \mathbf{x}, \mathbf{b} \in \mathbb{R}^n, \quad A \in \mathbb{M}_{n\times n}(\mathbb{R})$$

si on admet une erreur $\delta\mathbf{b}$ dans le second membre $\mathbf{b}$, c'est à dire qu'on résout un système $A(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b} + \delta\mathbf{b}$, alors on peut estimer l'erreur dans solution obtenue par (cf TD) :

$$\frac{\|\delta\mathbf{x}\|_q}{\|\mathbf{x}\|_q} \leq \mathrm{cond}_q(A)\frac{\|\delta b\|_q}{\|b\|_q}.$$

Si au contraire on modifie la matrice A par δA et on cherche $\mathbf{x} + \delta\mathbf{x}$: $(A + \delta A)(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b}$, alors

$$\frac{\|\delta\mathbf{x}\|_q}{\|\mathbf{x} + \delta\mathbf{x}\|_q} \leq \mathrm{cond}_q(A)\frac{\|\delta A\|_q}{\|A\|_q}.$$

En conclusion si $\mathrm{cond}(A) \gg 1$, même des petites perturbations des données (A ou $\mathbf{b}$) peut conduire à une énorme perturbation de la solution, dans ce cas on dit que le problème est *mal conditionné.*

Exemple. Considérons un exemple de résolution d'un système 2×2 qui permet de définir l'intersection entre deux fonctions affines en $2D$.

On remarque sur la figure 4 une explosion de $\mathrm{cond}_q(A)$ avec diminution de l'angle entre les droites ! ▼

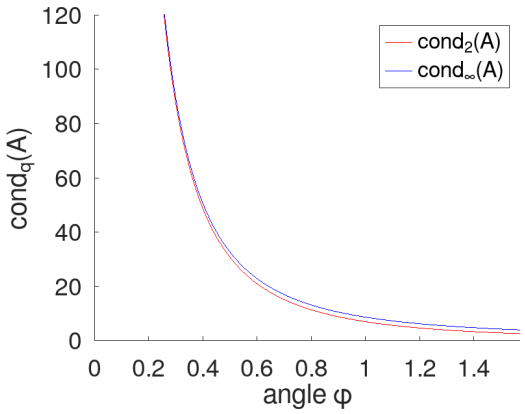


FIGURE 4 – Evolution du conditionement de la matrice A

Exemple. Un exemple connu d'une matrice mal conditionnée est la matrice de Hilbert définie par :

$$(H_n)_{i,j} = \frac{1}{i+j-1}, 1 \leq, i, j \leq n;$$

$$H_4 = \begin{pmatrix} 1 & 1/2 & 1/3 & 1/4 \\ 1/2 & 1/3 & 1/4 & 1/5 \\ 1/3 & 1/4 & 1/5 & 1/6 \\ 1/4 & 1/5 & 1/6 & 1/7 \end{pmatrix},$$

avec $\text{cond}_2(H_4) \approx 1.6 \times 10^4$, et même $\text{cond}_2(H_8) \approx 1.5 \times 10^{10}$. ▼

1.2 Méthodes directes : Élimination de Gauss et ses variantes

La formule de Cramer nous apporte une façon systématique de résoudre un système linéaire :

$$\text{Trouver un vecteur } \mathbf{x} \in \mathbb{R}^n, \text{ tel que } A\mathbf{x} = \mathbf{b} \iff \forall i \in [1, \dots, n] \quad x_i = \frac{\det(A_i(b))}{\det(A)}.$$

Toutefois il n'est pas toujours possible appliquer cette approche en pratique. Effectivement, les calculs des déterminants se font avec une relation récursive, et l'effort de calcul est de l'ordre $(n+1)!$ flops. Or la dimension typique d'un problème dans les applications est de l'ordre de $n = 10^2$ à $n = 10^7$.

	No. de flops de l'ordinateur				
n	10^9 (Giga)	10^{10}	10^{11}	10^{12} (Tera)	10^{15} (Peta)
10	10^{-1} sec	10^{-2} sec	10^{-3} sec	10^{-4} sec	négligeable
15	17 heures	1.74 heures	10.46 min	1 min	0.6 10^{-1} sec
20	4860 ans	486 ans	48.6 ans	4.86 ans	1.7 jour
25	h.l.	h.l.	h.l.	h.l.	38365 ans

FIGURE 5 – cf. Livre A. Quarteroni, R. Sacco, F. Saleri "Numerical Mathematics"

Objectif : Construire des algorithmes de resolution avec un coût de calculs raisonnable.

1.2.1 Algorithmes de descente et remontée

Définition 1.33 (Matrices triangulaires)**.**

(i) On dit qu'une matrice U est triangulaire supérieure si $\forall\, i > j \quad U_{i,j} = 0$,

(ii) On dit qu'une matrice L est triangulaire inférieur si $\forall\, i < j \quad L_{i,j} = 0$,

C'est à dire :

$$U = \begin{pmatrix} u_{11} & u_{21} & \dots & u_{1n} \\ 0 & u_{22} & \dots & u_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & u_{nn} \end{pmatrix}, \qquad L = \begin{pmatrix} l_{11} & 0 & \dots & 0 \\ l_{21} & l_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & \dots & l_{nn} \end{pmatrix}.$$

Proposition 1.34. *Soit L_1, L_2 (resp. U_1, U_2) deux matrices triangulaires inférieures (resp. supérieures), le produit L_1L_2 (resp. U_1U_2) est une matrice triangulaire inférieure (resp. supérieure).*

Démonstration.

□

Proposition 1.35. *Soit L (resp. U) une matrice triangulaire inférieure (resp. supérieure) inversible, son inverse L^{-1} (resp. U^{-1}) est une matrice triangulaire inférieure (resp. supérieure).*

Démonstration.

□

Dans le cas ou le système est de la forme :

$$L\mathbf{y} = \mathbf{b} \quad \text{ou} \quad U\mathbf{x} = \mathbf{y}$$

on a des algorithme de résolution très efficaces.

Exemple. Démontrons l'algorithme sur un exemple simple : Soit $L \in \mathbb{M}_{3\times 3}(\mathbb{R})$ une matrice inversible triangulaire inférieure.

▼

A présent, soient $L \in \mathbb{M}_{n\times n}(\mathbb{R})$ une matrice inversible triangulaire inférieure et

$\mathbf{b} \in \mathbb{R}^n$ un vecteur colonne. La résolution de $L\mathbf{y} = \mathbf{b}$ sous la forme

$$\begin{pmatrix} l_{11} & 0 & \dots & 0 \\ l_{21} & l_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & \dots & l_{nn} \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \Bigg\downarrow$$

s'écrit sous la forme de l'algorithme de descente :

Remarque. Pour que cet algorithme puisse être appliqué, on doit avoir $l_{ii} \neq 0$ ce qui revient à dire que la matrice L est inversible.

Algorithme 1 : [de descente] Résoudre $L\mathbf{y} = \mathbf{b}$.

```
pour i = 1 à n faire
    y_i ← b_i;
    pour j = 1 à i − 1 faire
        y_i ← y_i − l_ij y_j;
    fin
    y_i ← y_i / l_ii
fin
```

Pour calculer la *complexité* de l'algorithme on doit estimer le nombre d'opérations algébriques à chaque étape : à chaque itération i, on effectue $i-1$ sommes, $i-1$ produits et une division, soit au total

$$\sum_{i=1}^{n}(2(i-1)+1) = 2\left(\sum_{i=1}^{n} i\right) - n = n^2 \text{ opérations.}$$

De même, on construit l'*algorithme de remontée* pour résolution d'un système triangulaire supérieure donné par :

$$\begin{pmatrix} u_{11} & u_{21} & \dots & u_{1n} \\ 0 & u_{22} & \dots & u_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & u_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \Bigg\uparrow$$

Soit $U \in \mathbb{M}_{n\times n}(\mathbb{R})$ une matrice inversible triangulaire supérieure, alors $\mathbf{x} \in \mathbb{R}^n$, tel que $U\mathbf{x} = \mathbf{y}$, vérifie :

Algorithme 2 : [de remontée] Résoudre $U\mathbf{x} = \mathbf{y}$

pour $i = n$ *à* 1 **faire**
  $x_i \leftarrow y_i$;
  pour $j = i + 1$ *à* n **faire**
    $x_i \leftarrow x_i - u_{ij}x_j$;
  fin
  $x_i \leftarrow x_i/u_{ii}$
fin

1.2.2 Méthode de Gauss : Réduction

L'idée de la méthode d'élimination de Gauss est de réduire le système linéaire $A\mathbf{x} = \mathbf{b}$ à un système équivalent (c'est à dire à un système qui admet la même solution que le système initial) de la forme suivante :

$$U\mathbf{x} = \widetilde{\mathbf{b}},$$

où U est une matrice triangulaire supérieure et $\widetilde{\mathbf{b}}$ est le vecteur de second membre modifié. Ensuite ce nouveau système se résout par l'algorithme de remontée.

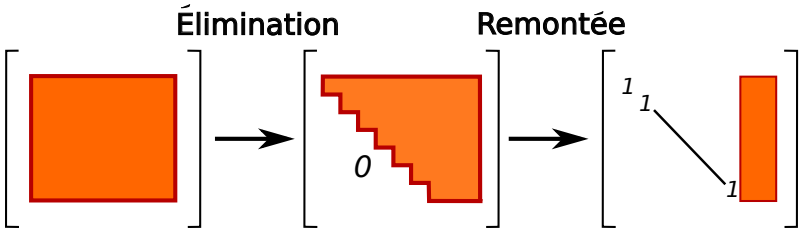


FIGURE 6 – Idée de l'algorithme de Gauss

Notons le système initial par $A^{(1)}\mathbf{x} = \mathbf{b}^{(1)}$. On suppose maintenant $a_{11} \neq 0$ et on introduit les multiplicateurs

$$m_{i1} = \frac{a_{i1}}{a_{11}}, \quad i = 2, \ldots, n.$$

La première étape de l'algorithme est alors :

$$\begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \ldots & a_{1n}^{(1)} \\ a_{21}^{(1)} & a_{22}^{(1)} & \ldots & a_{2n}^{(1)} \\ \vdots & \vdots & \vdots & \\ a_{n1}^{(1)} & a_{n2}^{(1)} & \ldots & a_{nn}^{(1)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1^{(1)} \\ b_2^{(1)} \\ \vdots \\ b_n^{(1)} \end{pmatrix} \iff \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \ldots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \ldots & a_{2n}^{(2)} \\ \vdots & \vdots & \vdots & \\ 0 & a_{n2}^{(2)} & \ldots & a_{nn}^{(2)} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_n^{(2)} \end{pmatrix}$$

où,

On note ce dernier système par $A^{(2)}\mathbf{x} = \mathbf{b}^{(2)}$. En procédant ainsi de suite, on

construit un système $A^{(k)}\mathbf{x} = \mathbf{b}^{(k)}$ où

$$A^{(k)} = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \dots & \dots & \dots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & & & & a_{2n}^{(2)} \\ \vdots & & \ddots & & & \vdots \\ 0 & \dots & 0 & a_{kk}^{(k)} & \dots & a_{kn}^{(k)} \\ \vdots & & \vdots & \vdots & & \vdots \\ 0 & \dots & 0 & a_{nk}^{(k)} & \dots & a_{nn}^{(k)} \end{pmatrix}$$

Le coefficient $a_{kk}^{(k)}$ est appelé le *pivot*. Si on suppose $\forall\, k = 1, \dots, n-1$ $a_{kk}^{(k)} \neq 0$, pour $k = n$ on obtient $A^{(n)}\mathbf{x} = \mathbf{b}^{(n)}$ avec

$$A^{(n)} = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \dots & \dots & \dots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & & & & a_{2n}^{(2)} \\ \vdots & & \ddots & & & \vdots \\ 0 & & & a_{kk}^{(k)} & \dots & a_{kn}^{(k)} \\ \vdots & & & & \ddots & \vdots \\ 0 & & & & & a_{nn}^{(n)} \end{pmatrix}$$ est une matrice triangulaire supérieure.

Résumé de passage d'une étape k à une étape $k+1$: Soit $a_{kk}^{(k)} \neq 0$, alors en définissant les multiplicateurs :

$$m_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}, \quad i = k+1, \dots, n \tag{2}$$

on définit

$$\begin{aligned} a_{ij}^{(k+1)} &= a_{ij}^{(k)} - m_{ik} a_{kj}^{(k)}, \quad i, j = k+1, \dots, n \\ b_i^{(k+1)} &= b_i^{(k)} - m_{ik} b_k^{(k)}, \quad i = k+1, \dots, n. \end{aligned} \tag{3}$$

FIGURE 7 – Voir pour un exemple détaillé d'application de la méthode de Gauss.

Exercice. Calculer la solution $H_3\mathbf{x} = \mathbf{b}$ où H_3 est une matrice de Hilbert d'ordre 3 définie dans exemple 1.1.7 et $\mathbf{b} = [11/6, 13/12, 47/60]^t$, en utilisant (2) et (3).

En résumé :

Algorithme 3 : Méthode de Gauss

Données : A une matrice inversible de taille n, $\mathbf{b}$ un vecteur colonne de taille n

Résultat : $\mathbf{x}$ un vecteur de taille n tel que $A\mathbf{x} = \mathbf{b}$

// Triangularisation de A et transformation de $\mathbf{b}$

pour $k = 1$ *à* $n - 1$ **faire**
  pour $i = k + 1$ *à* n **faire**
    $m_{ik} = a_{ik}/a_{kk}$
    pour $j = k$ *à* n **faire**
      $a_{ij} = a_{ij} - m_{ik}a_{kj}$
    fin
    $b_i = b_i - m_{ik}b_k$
  fin
fin

// Résolution du système transformé par l'algorithme de remontée

pour $i = n$ *à* 1 **faire**
  $x_i = b_i$
  pour $j = i + 1$ *à* n **faire**
    $x_i = x_i - a_{ij}x_j$
  fin
  $x_i = x_i/a_{ii}$
fin

Remarque (Choix du pivot (voir TD))**.** Considérons $A = \left(\begin{smallmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 7 & 8 & 9 \end{smallmatrix}\right)$ une matrice inversible. L'application de procédure d'élimination de Gauss au première étape donne : $A = \left(\begin{smallmatrix} 1 & 2 & 3 \\ 0 & 0 & -1 \\ 0 & -6 & -12 \end{smallmatrix}\right)$ (et l'exécution s'arrête si on pense à rajouter la vérification dans le code). Si $a_{kk}^{(k)} = 0$ on cherche un autre pivot :

1. Soit en permutant les lignes k et p où $|a_{pk}^{(k)}| = \max_{j=k+1,\dots,n} |a_{jk}^{(k)}|$.
2. Soit en permutant les colonnes k et p où $|a_{kp}^{(k)}| = \max_{j=k+1,\dots,n} |a_{kj}^{(k)}|$.
3. Soit en permutant les deux $|a_{pq}^{(k)}| = \max_{i=k+1,\dots,n} \max_{j=k+1,\dots,n} |a_{ij}^{(k)}|$.

Un autre avantage du pivotage est le suivant. Considérons le système $\begin{cases} 10^{-3}x + y &= b_1 \\ x + 10y &= b_2 \end{cases}$, il est équivalent aux deux systèmes suivant

$$\begin{cases} x + 10y &= b_2 \\ 10^{-3}x + y &= b_1 \end{cases} \quad \text{et} \quad \begin{cases} 10x + y &= b_2 \\ x + 10^{-3}y & 0b_1 \end{cases}$$

où on a permuté les deux équations et les deux inconnues dans le second cas. On note A_i la matrice associée au système i et U_i la matrice triangulaire supérieur résultante de la procédure de Gauss. On a

$$A_1 = \begin{pmatrix} 10^{-3} & 1 \\ 1 & 10 \end{pmatrix}, \quad U_1 = \begin{pmatrix} 10^{-3} & 1 \\ 0 & -990 \end{pmatrix},$$
$$A_2 = \begin{pmatrix} 1 & 10 \\ 10^{-3} & 1 \end{pmatrix}, \quad U_2 = \begin{pmatrix} 1 & 10 \\ 0 & 0.99 \end{pmatrix},$$
$$A_1 = \begin{pmatrix} 10 & 1 \\ 1 & 10^{-3} \end{pmatrix}, \quad U_3 = \begin{pmatrix} 10 & 1 \\ 0 & -0.099 \end{pmatrix}.$$

Bien que tous ces systèmes soient équivalents, ils ont des comportements différents vis à vis du conditionnement. en effet :

$$\begin{aligned} \mathrm{cond}_2(A_1) = \mathrm{cond}_2(A_2) = \mathrm{cond}_2(A_3) \approx 103, \\ \mathrm{cond}_2(U_1) \approx 990001, \quad \mathrm{cond}_2(U_2) \approx 103, \quad \mathrm{cond}_2(U_3) \approx 103. \end{aligned}$$

Lors du passage de A_1 à U_1 on constante que le conditionnement a étédétruit ; alors qu'il est restauré avec U_2 et U_3.

Considérons maintenant le coût de calculs de la méthode de Gauss. Pour l'étape d'élimination on a besoin de

$$2(n-1)n(n+1)/3 + n(n-1)$$

opérations. Ensuite n^2 pour l'algorithme de remontée. Ainsi au total $2n^3/3 + 2n^2$ opérations. Si on néglige les termes d'ordre inférieure on obtient le coût $\mathcal{O}(2n^3)$.

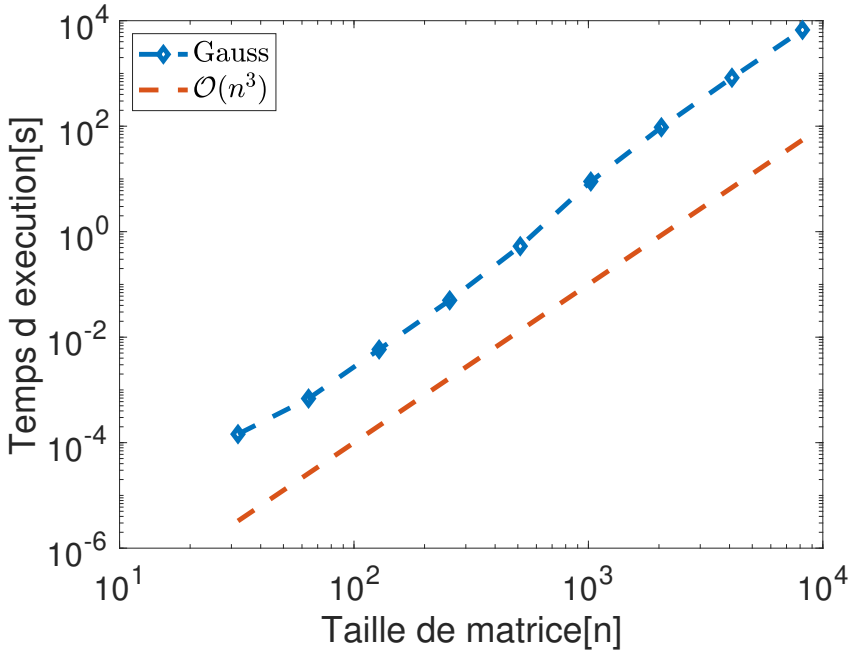


FIGURE 8 – Vérification numérique

	No. de flops de l'ordinateur		
n	10^9 (Giga)	10^{12} (Tera)	10^{15} (Peta)
10^2	$7 \cdot 10^{-4}$ sec	négligeable	négligeable
10^4	11 min	0.7 sec	$7 \cdot 10^{-4}$sec
10^6	21 ans	7.7 mois	11 min
10^8	h.l.	h.l.	21 ans

FIGURE 9 – Coût Numérique

1.2.3 Méthode de Gauss : factorisation LU

On peut voir l'algorithme décrit dans la section précédente comme une factorisation. Effectivement, en posant

$$M_k = \begin{pmatrix} 1 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & & 1 & 0 & & 0 \\ 0 & & -m_{k+1,k} & 1 & & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & -m_{n,k} & 0 & \dots & 1 \end{pmatrix} = \mathrm{I}_n - \mathbf{m}_k \mathbf{e}_k^t$$

où $\mathbf{m}_k = (0, ..., 0, m_{k+1,k}, ..., m_{n,k})^t \in \mathbb{R}^n$, on trouve

$$A^{(k+1)} = M_k A^{(k)}, \text{ et donc } M_{n-1} M_{n-2} \dots M_1 A = U.$$

La matrice M_k à diagonale unité admet une matrice inverse définie par

$$M_k^{-1} = 2\mathrm{I}_n - M_k = \mathrm{I}_n + \mathbf{m}_k \mathbf{e}_k^T.$$

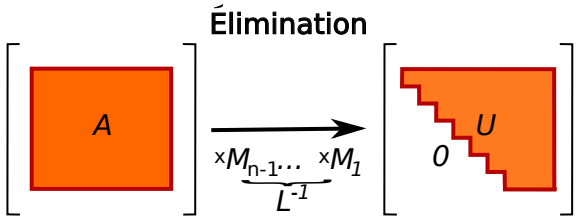
On remarque que $(\mathbf{m}_j \mathbf{e}_j^T)(\mathbf{m}_k \mathbf{e}_k^T)$ est une matrice nulle si $j \neq k$. Par conséquent,

$$M_1^{-1} M_2^{-1} \dots M_{n-1}^{-1} = \mathrm{I}_n + \sum_{k=1}^{n-1} \mathbf{m}_k \mathbf{e}_k = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ m_{21} & 1 & & & \vdots \\ \vdots & m_{32} & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & 0 \\ m_{n1} & m_{n2} & \dots & m_{n,n-1} & 1 \end{pmatrix}.$$

On définit alors

$$L = (M_{n-1} M_{n-2} \dots M_1)^{-1} = M_1^{-1} M_2^{-1} \dots M_{n-1}^{-1}$$

Et ainsi, $A = LU$.



Exercice. Calculer la décomposition LU pour la matrice $A = H_3$.

Algorithme 4 : Factorisation LU

Données : A une matrice inversible de taille n, $\mathbf{b}$ un vecteur colonne de taille n

Résultat : $\mathbf{x}$ un vecteur de taille n tel que $A\mathbf{x} = \mathbf{b}$

// Calcul de la décomposition LU. On initialise $L = I_n$. U est stockée dans A.

pour $k = 1$ *à* $n - 1$ **faire**
  pour $i = k + 1$ *à* n **faire**
    $\ell_{ik} = a_{ik}/a_{kk}$
    pour $j = k$ *à* n **faire**
      $a_{ij} = a_{ij} - \ell_{ik} a_{kj}$
    fin
  fin
fin

// Résolution de $L\mathbf{y} = \mathbf{b}$ par descente

pour $i = 1$ *à* n **faire**
  $y_i = b_i - \sum_{k=1}^{i-1} \ell_{ik} y_k$
fin

// Résolution de $U\mathbf{x} = \mathbf{y}$ par remontée

pour $i = n$ *à* 1 **faire**
  $x_i = 1/a_{ii} \left(y_i - \sum_{j=i+1}^{n} a_{ij} x_j \right)$
fin

Un autre avantage de la factorisation LU (par rapport à la réduction de Gauss) est qu'elle ne dépend pas du vecteur $\mathbf{b}$. On peut donc la calculer une fois pour A et ensuite seulement appliquer les algorithmes de descente et remontée pour chaque $\mathbf{b}$.

1.2.4 Méthode de Gauss : conditions sur A

Pour s'assurer que la méthode de Gauss atteint l'étape $(n-1)$, il ne suffit pas que A soit inversible, car cela ne garanti pas la condition nécessaire $\forall\, k = 1, \ldots, n-1$ $a_{kk}^{(k)} \neq 0$.

Theorem 1.36. *Soit A une matrice (pas forcément inversible). Elle admet une décomposition LU unique où L est à diagonale unité, i.e. $\forall\, i = 1, \ldots, n$ $L_{ii} = 1$ et U est une matrice triangulaire supérieure, si et seulement si toutes les sous matrices A_k, pour $k = 1, \ldots, n-1$, sont inversibles, où*

$$A_k \in \mathbb{M}_{k\times k}(\mathbb{R}) \quad \textit{et} \quad (A_k)_{ij} = A_{ij}, \forall (i,j) \in [1, \ldots, k]^2.$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1k} & \cdots & a_{1n} \\ a_{21} & a_{22} & & \cdots & & a_{2n} \\ \vdots & & \ddots & & & \vdots \\ a_{k1} & & \cdots & a_{kk}^{(k)} & \cdots & a_{kn}^{(k)} \\ \vdots & & & & \ddots & \vdots \\ a_{n1} & & \cdots & & & a_{nn}^{(n)} \end{bmatrix}$$

Démonstration.

□

Exercice. Dire si les matrices suivantes admettent une unique décomposition LU :

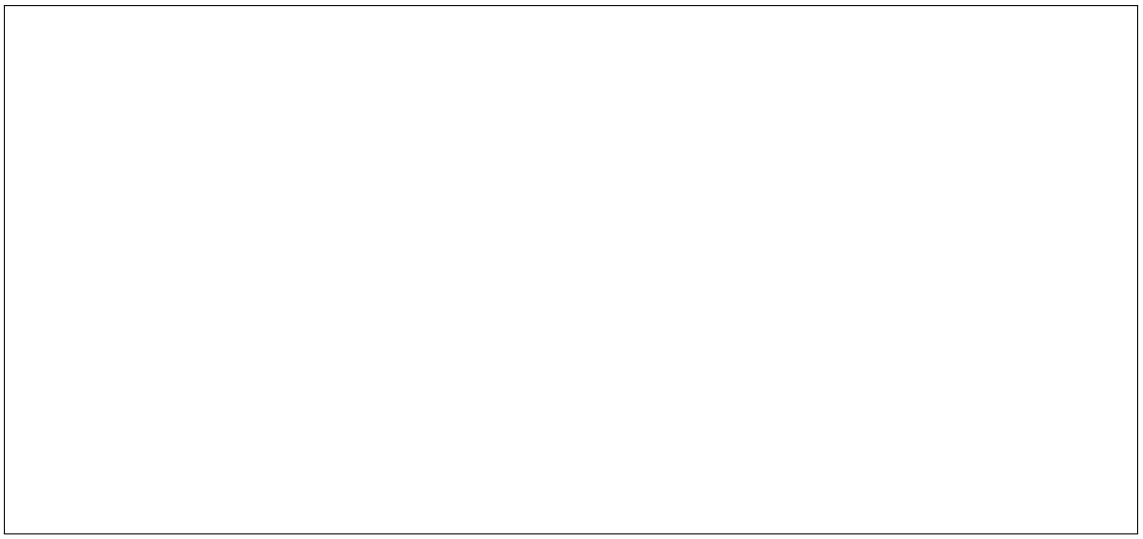
$$B = \begin{pmatrix} 1 & 2 \\ 1 & 2 \end{pmatrix}, \quad C = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad D = \begin{pmatrix} 0 & 1 \\ 0 & 2 \end{pmatrix}.$$

En fait, C n'admet aucune décomposition LU alors que D admet une infinité de de décomposition de la forme :

$$D = L_\alpha U_\alpha, \quad \text{avec} \quad L_\alpha = \begin{pmatrix} 1 & 0 \\ \alpha & 1 \end{pmatrix}, \quad U_\alpha = \begin{pmatrix} 0 & 1 \\ 0 & 2-\alpha \end{pmatrix}.$$

Proposition 1.37. *Si A est une matrice symétrique définie positive, alors ses sous matrices sont inversibles, et donc elle admet une décomposition LU.*

Démonstration.



□

Remarque (Stabilité de la méthode LU)**.** Pour une analyse profond d'erreurs d'arrondi (erreurs machine) vous pouvez consulter *A. Quarteroni, R. Sacco, F. Saleri.* Ici on mentionne juste que la stabilité de l'algorithme d'élimination de Gauss dépend de magnitudes des éléments de $A^{(k)}$, plus précisément de $\rho = \frac{\max\limits_{i,j,k} |A_{ij}^{(k)}|}{\max\limits_{i,j} |A_{ij}|}$. Mauvaise nouvelle : croissance exponentielle $\rho \sim 2^n$ est possible.

Exemple (**Wilkinson**)**.** Considérons une matrice définie comme suit

$$a_{ij} = \begin{cases} 1 & , \text{ si } i = j \vee j = n \\ -1 & , \text{ si } i > j \\ 0 & , \text{ sinon.} \end{cases}, \quad A = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & -1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 1 \\ -1 & -1 & -1 & -1 & -1 & 1 & 0 & 0 & 0 & 1 \\ -1 & -1 & -1 & -1 & -1 & -1 & 1 & 0 & 0 & 1 \\ -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 0 & 1 \\ -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 \end{pmatrix}.$$

Alors on a :

$$A = LU, \quad l_{ij} = \begin{cases} 1 & , \text{ si } i = j, \\ -1 & , \text{ si } i > j, \\ 0 & , \text{ sinon}\,; \end{cases} \quad u_{ij} = \begin{cases} 1 & , \text{ si } i = j \\ 2^{\text{i-1}} & , \text{ si } j = n \\ 0 & , \text{ sinon.} \end{cases}$$

▼

Cependant, bonne nouvelle : en pratique, $\rho \sim \sqrt{n}$.

1.2.5 Méthode Choleski

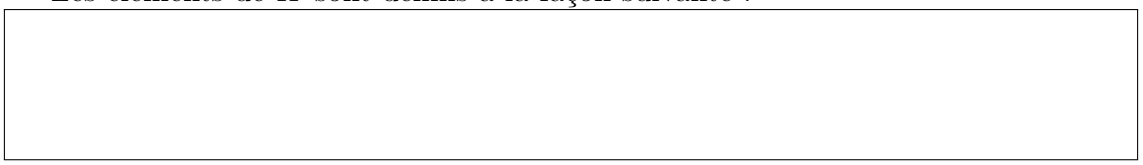
On va maintenant étudier la méthode de Choleski, qui est une méthode directe adaptée au cas où A est symétrique définie positive. On rappelle (Théorème 1.28) que si A est symétrique définie positive elle est en particulier inversible.

Proposition 1.38. *Si A est une matrice symétrique définie positive, alors elle admet une décomposition unique de la forme $A = HH^t$ où H est une matrice triangulaire inférieure à diagonale positive.*

Démonstration.

□

Les éléments de H sont définis à la façon suivante :



Exercice. Démontrez les formules ci-dessus.

Algorithme 5 : Méthode de Choleski

pour $k = 1$ *à* n **faire**
  $h_{kk} = a_{kk}$
  pour $i = 1$ *à* $k-1$ **faire**
    $h_{kk} = h_{kk} - h_{ik}^2$
  fin
  $h_{kk} = \sqrt{h_{kk}}$
  pour $i = k+1$ *à* n **faire**
    $h_{ik} = a_{ik}$
    pour $j = 1$ *à* $k-1$ **faire**
      $h_{ik} = h_{ik} - h_{ij}h_{kj}$
    fin
    $h_{ik} = h_{ik}/h_{kk}$
  fin
fin

Remarque (Pivot partiel et Choleski.)**.** Considérons une matrice A symétrique définie positive. On n'a pas besoin de permutation pour obtenir sa décomposition HH^t de Choleski. Par contre, numériquement, on utilise malgré tout la technique de pivot partiel pour minimiser les erreurs d'arrondi.

Comparaison Gauss/Choleski : Il est important de souligner que l'algorithme de Choleski permet à garder la nature symétrique de la matrice A ce qui permet d'économiser de la mémoire. Le coût de calcul de la résolution d'un système linéaire par la méthode de Choleski est $n^3/3 + \mathcal{O}(n^2)$, tandis que la méthode de Gauss demande $2n^3/3 + \mathcal{O}(n^2)$ opérations (cf TD). Dans le cas d'une matrice symétrique définie positive, la méthode de Choleski est donc environ deux fois moins chère. Mais attention dans le contexte actuel des ordinateurs modernes ça ne veut pas dire qu'elle est deux fois plus rapide (la gestion de la mémoire joue un rôle très important).

1.2.6 Factorisation QR : orthogonalisation de Gram-Schmidt

Idée : Factoriser la matrice A sous la forme $A = QR$ où R est une matrice triangulaire supérieure et Q est une matrice orthogonale ($Q^{-1} = Q^t$). La résolution du problème linéaire se fait alors en appliquant l'algorithme de remontée au vecteur $Q^t\mathbf{b}$.

Remarque. Dans un cas d'une matrice à coefficients complexes, Q est une matrice unitaire ($Q^{-1} = Q^*$).

Cette factorisation se construit en utilisant soit l'algorithme d'orthogonalisation de Gram-Schmidt soit les matrices de transformation : Householder ou Givens (cf TD).

Theorem 1.39 (Gram-Schmidt)**.** *Soit un ensemble de k vecteurs $(\mathbf{v}_1, \ldots, \mathbf{v}_k)$ linéairement indépendant. On peut construire un ensemble de vecteurs $(\mathbf{q}_1, \ldots, \mathbf{q}_k)$ vérifiant :*

(i) $\mathit{Vect}(\mathbf{v}_1, \ldots, \mathbf{v}_k) = \mathit{Vect}(\mathbf{q}_1, \ldots, \mathbf{q}_k)$

(ii) $(\mathbf{q}_i, \mathbf{q}_j) = \delta_{ij}$

Démonstration.

□

Ainsi les nouveaux vecteurs $\mathbf{q}_i$ sont construit à la façon suivante :

$$\widetilde{\mathbf{q}}_i = \mathbf{v}_i - \sum_{j=1}^{i-1} (\mathbf{v}_i, \mathbf{q}_j)\, \mathbf{q}_j, \qquad \mathbf{q}_i = \widetilde{\mathbf{q}}_i / \|\widetilde{\mathbf{q}}_i\|$$

Algorithme 6 : Algorithme de Gram-Schmidt

```
pour k = 1 à n faire
    q_i = v_i
    pour k = 1 à i − 1 faire
        q_i = q_i − (v_i, q_j)q_j
    fin
    q_i = q_i/‖q_i‖
fin
```

Remarque. Attention, le calcul de la complexité pour cet algorithme ne peut pas être déterminée directement. En effet, les produits scalaires ne sont pas des opérations élémentaires

Theorem 1.40. *Soit A une matrice inversible. Elle admet une décomposition QR où Q est une matrice orthogonale et R est une matrice triangulaire supérieure.*

Démonstration.

□

Remarque. Numériquement, on évite l'utilisation de cette approche car l'algorithme d'orthogonalisation de G.S. est instable et conduit à une matrice Q non parfaitement orthogonale.

Exemple. Les ordinateurs ne font pas les opérations sur $\mathbb{R}$ "correctement". Calculons décomposition QR de matrice de Hilbert $H_{10} \in \mathbb{M}_{10\times 10}$ en utilisant l'algorithme 6, et faisons une simple vérification :

$$Q^tQ = \begin{pmatrix} 1.0000 & 0.0000 & -0.0000 & 0.0000 & -0.0000 & 0.0000 & -0.0000 & -0.0000 & -0.0000 & -0.0000 \\ 0.0000 & 1.0000 & -0.0000 & 0.0000 & -0.0000 & 0.0000 & -0.0000 & -0.0000 & -0.0000 & -0.0000 \\ -0.0000 & -0.0000 & 1.0000 & 0.0000 & -0.0000 & 0.0000 & -0.0000 & -0.0000 & -0.0000 & -0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 1.0000 & -0.0000 & 0.0000 & -0.0000 & -0.0000 & -0.0000 & -0.0000 \\ -0.0000 & -0.0000 & -0.0000 & -0.0000 & 1.0000 & 0.0000 & -0.0008 & -0.0007 & -0.0007 & -0.0006 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 1.0000 & -0.0540 & -0.0430 & -0.0360 & -0.0289 \\ -0.0000 & -0.0000 & -0.0000 & -0.0000 & -0.0008 & -0.0540 & 1.0000 & 0.9999 & 0.9998 & 0.9996 \\ -0.0000 & -0.0000 & -0.0000 & -0.0000 & -0.0007 & -0.0430 & 0.9999 & 1.0000 & 1.0000 & 0.9999 \\ -0.0000 & -0.0000 & -0.0000 & -0.0000 & -0.0007 & -0.0360 & 0.9998 & 1.0000 & 1.0000 & 1.0000 \\ -0.0000 & -0.0000 & -0.0000 & -0.0000 & -0.0006 & -0.0289 & 0.9996 & 0.9999 & 1.0000 & 1.0000 \end{pmatrix}$$

▼

Il faut donc proposer un moyen stable de calculer la décomposition QR.

1.2.7 Méthode de Householder

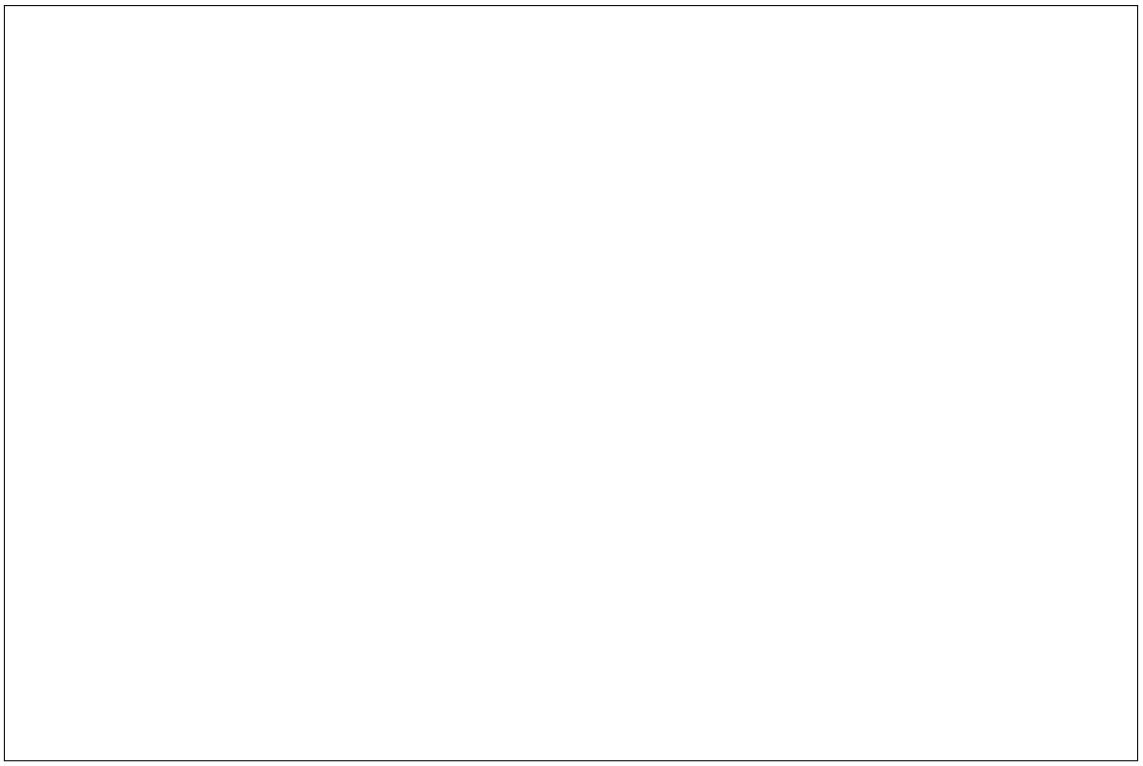
Soit $\mathbf{v} \in \mathbb{R}^n$. On définit la matrice élémentaire de Householder $H(\mathbf{v})$ comme suit

$$H(\mathbf{v}) = I - 2\frac{\mathbf{v}\,\mathbf{v}^t}{\|\mathbf{v}\|^2}$$

Interprétation géométrique :

Proposition 1.41. *La matrice $H(\mathbf{v})$ est symétrique et orthogonale et $H(\mathbf{v}\pm\|\mathbf{v}\|\mathbf{e}_k)\mathbf{v} = \mp\|\mathbf{v}\|\mathbf{e}_k$*

Démonstration.



□

Exercice. De la même manière que dans la preuve précédente, montrez que si $\mathbf{v} = \mathbf{b} - \mathbf{a} \neq 0$ avec $\|b\| = \|a\|$, alors $H(\mathbf{v})\mathbf{a} = \mathbf{b}$.

Remarque. On déduit de la proposition précédente qu'on peut transformer un vecteur $\mathbf{x}$ quelconque en un vecteur n'ayant qu'une coordonnée non nulle en multipliant par la matrice de Householder qui convient, via le choix du bon vecteur unitaire de la base canonique

Exemple. Considérons $\mathbf{x} = [1, 1, 1, 1]^t$.

▼

On remarque également que les matrices $H(\mathbf{u})$ (pour des vecteurs $\mathbf{u}$ bien choisis) nous permettent de réduire la matrice A à une matrice triangulaire inférieure. Cette idée forme une base pour la méthode de Householder :

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{2n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \times H^{(1)} \left(\mathbf{a}_1 + \|\mathbf{a}_1\| \, \mathbf{e}_1 \right) \quad \text{où} \quad \mathbf{a}_1 = (a_{11}, \dots, a_{n1})^t$$

$$\to A^{(2)} = \begin{pmatrix} a_{11}^{(2)} & a_{12}^{(2)} & \dots & a_{1n}^{(2)} \\ 0 & a_{22}^{(2)} & \dots & a_{2n}^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2}^{(2)} & \dots & a_{nn}^{(2)} \end{pmatrix} \times H^{(2)} \left(\tilde{\mathbf{a}}_2 + \|\tilde{\mathbf{a}}_2\| \, \mathbf{e}_2 \right) \quad \text{où} \quad \tilde{\mathbf{a}}_2 = \left(0, a_{22}^{(2)}, \dots, a_{n2}^{(2)} \right)^t$$

$$\to A^{(3)} = \begin{pmatrix} a_{11}^{(3)} & a_{12}^{(3)} & a_{13}^{(3)} & \dots & a_{1n}^{(3)} \\ 0 & a_{22}^{(3)} & a_{23}^{(3)} & \dots & a_{2n}^{(3)} \\ 0 & 0 & a_{33}^{(3)} & \dots & a_{3n}^{(3)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & \dots & a_{nn}^{(3)} \end{pmatrix} \quad \dots$$

A l'étape k on construit une matrice $A^{(k)}$ comme

$$A^{(k)} = H^{(k-1)} H^{(k-2)} \dots H^{(1)} A$$

avec$\forall\, k$ $H^{(k)} = H\left(\tilde{\mathbf{a}}_k^{(k)} + \|\tilde{\mathbf{a}}_k^{(k)}\| \mathbf{e}_k \right), \tilde{\mathbf{a}}_k^{(k)} = (0 \dots, a_{kk}^{(k)}, \dots a_{nk}^{(k)})^t$ et

$$A^{(k)} = \begin{pmatrix} a_{11}^{(k)} & a_{12}^{(k)} & \dots & a_{1k}^{(k)} & \dots & a_{1n}^{(k)} \\ 0 & a_{22}^{(k)} & \dots & a_{2k}^{(k)} & \dots & a_{2n}^{(k)} \\ 0 & 0 & \ddots & \vdots & \vdots & \vdots \\ \vdots & \vdots & 0 & a_{kk}^{(k)} & \dots & a_{kn}^{(k)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & a_{nk}^{(k)} & \dots & a_{nn}^{(k)} \end{pmatrix}.$$

On déduit alors :

$$A^{(n)} = H^{(n-1)} H^{(n-2)} \dots H^{(1)} A = R(\text{triangulaire supérieure !})$$

et

$$Q = (H^{(n-1)} H^{(n-2)} \dots H^{(1)})^{-1} = \left(H^{(1)}\right)^t \dots \left(H^{(n-2)}\right)^t \left(H^{(n-1)}\right)^t = H^{(1)} \dots H^{(n-2)} H^{(n-1)}$$

Remarque. Attention ! ! La décomposition QR conduit a une matrice triangulaire supérieure différente de la matrice U de la décomposition LU, $R \neq U$.

La *complexité* de la méthode de Householder s'estime à $\mathcal{O}\left(\frac{4n^3}{3}\right)$ (donc plus élevée que pour l'algorithme de la décomposition LU). Mais une des avantages importants de la méthode de Householder est sa stabilité numérique.

Cette méthode s'applique également pour le cas des systèmes surdéterminés où $A \in \mathbb{M}_{n\times p}$, $n \geq p$ est une matrice rectangulaire, ainsi que pour des problèmes de minimisation écrits :

$$\text{Trouver } \min_{x\in\mathbb{R}^p} \|A\mathbf{x} - \mathbf{b}\|$$

où $A \in \mathbb{M}_{n\times p}$, $(n \neq p)$ (cf TD).

1.3 Rappels d'algèbre linéaire (suite)

1.3.1 Éléments de théorie spectrale

Définition 1.42. Soit A une matrice carré. On appelle polynôme caractéristique et on note $P_A(\lambda)$ le polynôme définie par

Ce polynôme de degré n admet n racines complexes appelées valeurs propres. La multiplicité d'une racine est appelée *multiplicité algébrique* de la valeur propre.

Définition 1.43. On appelle vecteur propre $\mathbf{x}$ associé à la valeur propre λ un vecteur non nul vérifiant $A\mathbf{x} = \lambda\mathbf{x}$.

Remarque. Si λ est valeur propre, alors il existe au moins un vecteur propre associé car $\det(A - \lambda I) = 0$

Définition 1.44. On appelle rayon spectrale et on note $\rho(A)$ le maximum des modules des valeurs propres.

Définition 1.45. On appelle sous espace propre associé à la valeur propre λ l'e.v. défini par $E(\lambda) = \ker(A - \lambda I)$.

Définition 1.46. On appelle *multiplicité géométrique* de la dimension du sous espace propre associé.

Proposition 1.47 (Admis)**.** *La multiplicité algébrique est supérieure ou égale à la multiplicité géométrique.*

Proposition 1.48. *Soit* $\mathbf{x} \in E(\lambda_1)$ *et* $\mathbf{y} \in E(\lambda_2)$ *où* $\lambda_1 \neq \lambda_2$ *et* $\mathbf{x}, \mathbf{y} \neq 0$*. Les vecteurs propres* $\mathbf{x}$ *et* $\mathbf{y}$ *sont linéairement indépendants.*

Démonstration.

□

Définition 1.49. On dit qu'une matrice A est diagonalisable si et seulement si il existe une matrice inversible P t.q. $A = PDP^{-1}$ où D est diagonale.

Remarque. Dire qu'une matrice est diagonalisable revient à dire qu'il existe une base dans laquelle A est diagonale. P représente alors la matrice de passage de cette base dans la base canonique. On dira que A et D sont des matrices semblables.

Proposition 1.50 (Admis)**.** *A est diagonalisable si et seulement si la multiplicité géométrique est égale à multiplicité algébrique pour toutes les valeurs propres. De plus D est formé des valeurs propres de A.*

Exercice. Les matrices suivantes sont elles diagonalisables ?

$$A = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad C = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

Theorem 1.51 (Factorisation de Schur, admis)**.**
Toute matrice A est triangularisable dans une base orthonormale, ç.à.d admet une décomposition

$$A = UTU^*$$

où U une matrice unitaire et T une matrice triangulaire supérieure.

Remarque.

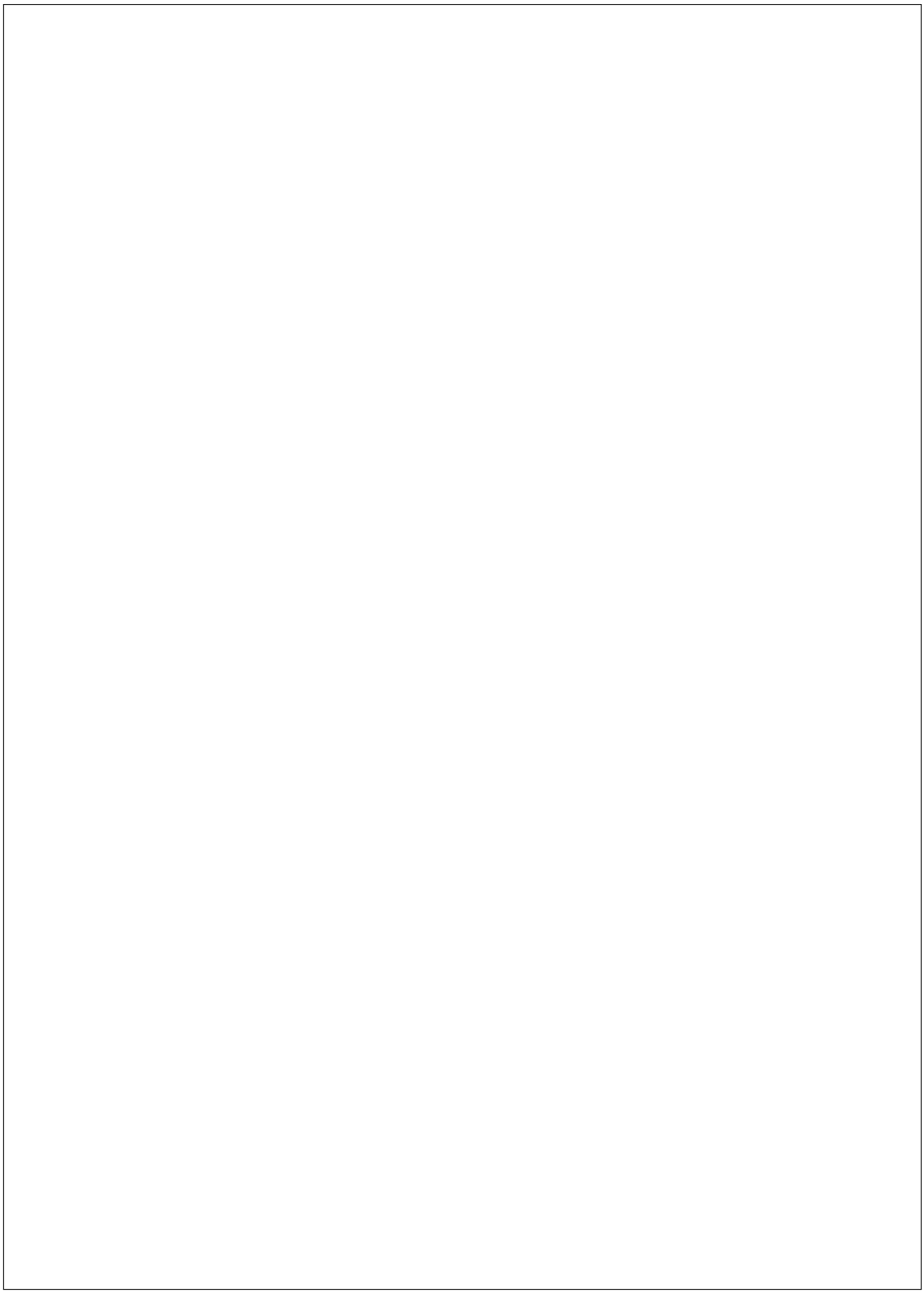
1. On rappelle qu'une matrice unitaire vérifie $U^*U = I$.
2. Les coefficients des matrices U et T sont à priori complexes, même si A est à coefficients réels.
3. Les coefficients diagonaux de T sont égaux aux valeurs propres de A. En effet :

Définition 1.52. On appelle A une matrice normale si elle commute avec son adjointe :

$$AA^* = A^*A.$$

Theorem 1.53 (Diagonalisation base orthonormale)**.**
Une matrice A est diagonalisable dans une base orthonormale si et seulement si elle est normale

Démonstration.



□

Proposition 1.54 (voir TD)**.** *Soit A une matrice telle que $A = A^*$. On a alors :*

où le spectre de A est $\lambda(A) = \{\lambda_1, \ldots, \lambda_n\}$ avec $|\lambda_1| \leq \ldots \leq |\lambda_n|$.

1.3.2 Série de matrices

Définition 1.55. Soit $P(x)$ un polynôme définie par $P(x) = \sum_{i=0}^{n} a_i x^i$. On note $P(A)$ une matrice définie par $P(A) = \sum_{i=0}^{n} a_i A^i$.

Proposition 1.56. *Si λ est valeur propre de A, alors $P(\lambda)$ est valeur propre de $P(A)$.*

Démonstration.

□

Theorem 1.57 (Cayley - Hamilton)**.** *En rappelant qu'on définit le polynôme caractéristique* $P_A(\lambda) = \det(A - \lambda I)$*, on a*

Démonstration.

□

Remarque.

1. On déduit de ce théorème que pour toute matrice A, on peut exprimer A^n en fonction de A^0, ..., A^{n-1}.
2. On appelle polynôme minimal le polynôme P de plus petit degré satisfaisant $P(A) = 0$
   On déduit du théorème ci-dessus que $\deg(P) \leq n$.

Définition 1.58. On dira qu'une suite de matrices $(A_i)_{i\geq 0}$ converge vers A si

où $\|\cdot\|$ est une norme matricielle.

Remarque. Dans la définition ci-dessus, le choix de la norme ne compte pas. En effet, l'espace des matrices est de dimension fini, et on sait que toutes les normes sont alors équivalentes.

Proposition 1.59 (voir TD)**.**
Pour toute matrice $A \in \mathbb{M}_{n\times n}(\mathbb{R})$ *et pour tout* $\varepsilon > 0$ *, il existe une norme matricielle*

subordonnée $\|\cdot\|$ *telle que*

$$\rho(A) \leq \|A\| \leq \rho(A) + \varepsilon.$$

On verra par la suite que la convergence des méthodes itératives nécessite une étude de la convergence suivante

$$\lim_{i \to +\infty} A^i = 0.$$

Proposition 1.60 (CNS)**.** *Soit* $A \in \mathbb{M}_{n\times n}(\mathbb{R})$*, alors les conditions suivante sont équivalentes*

(i) $\lim_{i \to +\infty} A^i = 0$,

(ii) $\forall\, \mathbf{x}$, $\lim_{i \to +\infty} A^i \mathbf{x} = 0$

(iii) $\rho(A) < 1$

(iv) Pour au moins une norme subordonnée $\|A\| < 1$

Démonstration.

□

Le rayon spectral est donc l'information déterminante pour étudier la convergence des suites de matrices.

Objectif : Construire des algorithmes itératifs de résolution d'un système linéaire et définir leurs propriétés et avantages.

1.4 Méthodes itératives

Dans la première partie de ce chapitre, nous nous sommes intéressés à des méthodes directes pour résoudre le problème linéaire. On rappelle les deux approches :

1. la factorisation LU et ses extensions (permutations, Cholesky, matrices bandes)
2. la factorisation QR

Les méthodes directes se caractérisent par le fait que la résolution s'effectue en un nombre fini d'étapes. Les méthodes directes sont très efficaces : elles donnent la solution exacte (aux erreurs d'arrondi près) du système linéaire considéré. Elles ont l'inconvénient de nécessiter une assez grande place mémoire car elles nécessitent le stockage de toute la matrice en mémoire vive.

Cependant, si le système a été obtenu à partir de la discrétisation d'équations aux dérivés partielles, il est en général "creux", c.à.d. qu'un grand nombre des coefficients de la matrice du système sont nuls ; de plus la matrice a souvent une structure "bande", i.e. les éléments non nuls de la matrice sont localisés sur certaines diagonales. On a vu que dans ce cas, la méthode de Cholesky "conserve le profil" (voir TD).

Lorsqu'on a affaire à de très gros systèmes issus de l'ingénierie (mécanique des fluides, calcul des structures, ...), où n peut être de l'ordre de plusieurs milliers, on cherche à utiliser des méthodes nécessitant le moins de mémoire possible.

Dans cette partie on s'intéresse à des méthodes itératives qui, à la différence des méthodes directes, convergent théoriquement en un nombre infini d'étapes. "L'espoir" est qu'en peu d'itérations, la solution approchée soit très proche de la solution exacte et que le coût des itérations soit faible.

Soit $A \in \mathbb{M}_{n\times n}(\mathbb{R})$ une matrice inversible et $\mathbf{b} \in \mathbb{R}^n$, on cherche toujours ici à résoudre le système linéaire $A\mathbf{x} = \mathbf{b}$, mais de façon itérative, c.à.d. par la construction d'une suite.

Définition 1.61. On appelle méthode itérative de résolution du système linéaire $A\mathbf{x} = \mathbf{b}$ une méthode qui construit une suite $(\mathbf{x}^k)_{k\in\mathbb{N}}$ (où l'itéré $\mathbf{x}^k$ est calculé à partir des itérés précédent $\mathbf{x}^0, \ldots, \mathbf{x}^{k-1}$) convergente vers $\mathbf{x}$ solution du système.

Définition 1.62. On dit qu'une méthode itérative est convergente si pour tout choix initial $\mathbf{x}^0 \in \mathbb{R}^n$, on a :

Pour résoudre le système linéaire $A \in \mathbb{M}_{n\times n}(\mathbb{R})$, $\mathbf{b} \in \mathbb{R}^n$:

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

une idée naturelle est de travailler avec une matrice M inversible qui soit "proche" de A, mais plus facile que A à inverser. On appelle cette matrice M matrice de préconditionnement. On écrit alors $A = M - N$ et on réécrit le système linéaire comme suit $M\mathbf{x} = (M - A)\mathbf{x} + \mathbf{b} = N\mathbf{x} + \mathbf{b}$.

Cette forme suggère la construction de la suite $\mathbf{x}^{k+1}$

(4)

Remarque. Si l'algorithme converge, i.e. $\mathbf{x}^k \to \mathbf{x}^*$, alors il converge vers la solution du problème initial.

L'objectif sera de choisir convenablement M pour que le calcul de $M^{-1}\mathbf{y}$ soit peu coûteux, pour que l'algorithme converge et qu'il converge vite. On rappelle (définition 1.62) que la méthode itérative (4) converge si $\forall\, \mathbf{x}_0 \in \mathbb{R}^n \lim_{k\to\infty} \mathbf{x}^k = \mathbf{x}$

Proposition 1.63. *La méthode converge si et seulement si* $\rho(M^{-1}N) < 1$.

Démonstration.

□

Critère d'arrêt : En théorie, il faudrait effectuer un nombre infini d'itérations pour obtenir la solution exacte d'un système linéaire avec une méthode itérative. En pratique, ce n'est ni nécessaire, ni raisonnable. L'objectif étant de faire peu d'itérations, il important d'avoir un bon critère d'arrêt de la méthode :

- Un premier estimateur est donné par le résidu $\mathbf{r}^k = \mathbf{b} - A\mathbf{x}^k$ comme suit :

$$\|\mathbf{r}^k\| \leq \varepsilon \|\mathbf{b}\|,$$

avec ce critère on a

Ce test est donc pertinent si le conditionnement de A n'est pas trop grand.

- Un autre estimateur est donné par l'incrément

$$\delta^k = \mathbf{x}^{k+1} - \mathbf{x}^k.$$

On peut montrer que le contrôle par l'incrément n'est pertinent que quand $\rho(M^{-1}N) \ll 1$.

En résumé, on a deux critères d'arrêt possible pour les méthodes itératives : l'un basé sur le résidu, l'autre sur l'incrément. Le premier est pertinent quand le système est bien conditionné, le second quand le rayon spectral de la matrice d'itération n'est pas trop proche de 1.

Estimation de la vitesse de convergence : Soit $\mathbf{x}^0 \in \mathbb{R}^n$ donné et soit $(\mathbf{x}^k)_{k\geq 0}$ la suite définie par (4). Notons la matrice d'itération $B = M^{-1}N$. On a vu que, si $\rho(B) < 1$, alors $\mathbf{x}^k \to \mathbf{x}$ quand $k \to \infty$, où $\mathbf{x}$ est la solution du système $A\mathbf{x} = \mathbf{b}$. On peut montrer (sauf cas particuliers qu'on ne précise pas ici)

Proposition 1.64 (Admis)**.**

$$\frac{\|\mathbf{x}^{k+1} - \mathbf{x}\|}{\|\mathbf{x}^k - \mathbf{x}\|} \to \rho(B) \text{ } \textit{lorsque } k \to +\infty.$$

indépendamment de la norme choisie sur $\mathbb{R}^n$*.*

Remarque.

1. Le rayon spectral $\rho(B)$ de la matrice B est donc une bonne estimation de la vitesse de convergence.
2. Pour estimer cette vitesse de convergence lorsqu'on ne connaît pas x, on peut utiliser le fait qu'on a aussi

$$\frac{\|\mathbf{x}^{k+1} - \mathbf{x}\|}{\|\mathbf{x}^k - \mathbf{x}\|} \to \rho(B) \text{ lorsque } k \to +\infty.$$

En résumé, l'étude des méthodes itératives repose donc sur deux questions suivantes :

- étant donné une méthode itérative de matrice $B = M^{-1}N$, déterminer si la méthode est convergente, c'est-à-dire si $\rho(M^{-1}N) < 1$, ou de façon équivalente, s'il existe une norme matricielle telle que $\|M^{-1}N\| < 1$.
- étant donné deux méthodes itératives convergentes, les comparer : la méthode itérative la plus "rapide" est celle dont la matrice a le plus petit rayon spectral.

Enfin, l'algorithme général s'écrit de la façon suivante :

Algorithme 7 : Algorithme itératif ($A = M - N$)

$\mathbf{x} = \mathbf{x}^0$
$\mathbf{r}^0 = A\mathbf{x} - \mathbf{b}$
tant que $\|\mathbf{r}\| \geq \varepsilon\|\mathbf{r}^0\|$ **faire**
  $\mathbf{y} = M^{-1}\mathbf{r}$
  $\mathbf{x} = \mathbf{x} - \mathbf{y}$
  $\mathbf{r} = A\mathbf{x} - \mathbf{b}$
fin

On a bien $\mathbf{x}^{k+1} = \mathbf{x}^k - \mathbf{y}^k = \mathbf{x}^k - M^{-1}(M - N)\mathbf{x}^k + M^{-1}\mathbf{b} = M^{-1}N\mathbf{x}^k + M^{-1}\mathbf{b}$

Remarque. Pour la norme 2, on effectue le test d'arrêt directement sur le carré des normes pour éviter le calcul de la racine.

1.4.1 Méthode de Jacobi et de Gauss-Seidel

Soit $A = D - E - F$ la décomposition de A définie par :

$$D = (d_{ij})_{i,j=1,\dots,n} \quad \text{avec} \quad \begin{cases} d_{ii} = a_{ii} & i = 1, \dots, n \\ d_{ij} = 0 & i \neq j \end{cases}$$

$$E = (e_{ij})_{i,j=1,\dots,n} \quad \text{avec} \quad \begin{cases} e_{ij} = -a_{ij} & i > j \\ e_{ij} = 0 & i \leqslant j \end{cases}$$

$$F = (f_{ij})_{i,j=1,\dots,n} \quad \text{avec} \quad \begin{cases} f_{ij} = -a_{ij} & i < j \\ f_{ij} = 0 & i \geqslant j \end{cases}$$

Méthode de Jacobi **Méthode de Gauss-Seidel**

On pose :

$$\begin{cases} M = D \\ N = F + E \end{cases} \qquad \begin{cases} M = D - E \\ N = F \end{cases}$$

La matrice d'itérations s'écrit :

$$B = D^{-1}(E + F) \qquad B = (D - E)^{-1}F$$

Les composantes du vecteur $\mathbf{x}^{k+1} = (x_i^{k+1})_{i=1}^n$ sont alors

Remarque.

1. Les deux méthodes nécessite $A_{ii} \neq 0$.
2. Coût / itérations de la méthode de Jacobi est $\mathcal{O}(2n^2)$ et de la méthode de Gauss-Seidel est $\mathcal{O}(3n^2)$.

Comparaison :

Méthode de Jacobi :

Méthode de Gauss-Seidel :

Pour calculer une composante quelconque x_i^{k+1} par la méthode de Jacobi on utilise donc $(n-1)$ des composantes du vecteur $\mathbf{x}^k$, qu'il faut donc garder en mémoire

pendant tout le calcul du vecteur $\mathbf{x}_{k+1}$. Autrement dit, une itération de la méthode immobilise $2n$ mémoires de l'ordinateur. La méthode de Gauss-Seidel permet utiliser "mieux" les quantités déjà calculées en utilisant la "nouvelle" valeur x_i^{k+1} plutôt que l'"ancienne" valeur x_i^k. Par conséquent, on peut espérer que la méthode de G.S. converge plus vite que celle de Jacobi. En revanche, la méthode de Jacobi présente l'avantage d'être facilement parallélisable.

Définition 1.65. On dit qu'une matrice est à diagonale strictement dominante si et seulement si

Proposition 1.66. *Si la matrice A est à diagonale strictement dominante, alors les méthodes de Jacobi et Gauss-Seidel convergent.*

Démonstration.

□

1.4.2 Méthode de descente

Considérons le cas d'une matrice A symétrique définie positive.

Proposition 1.67. *Si $A = A^t > 0$ le produit $(A\mathbf{x}, \mathbf{y}) = \mathbf{y}^t A\mathbf{x}$ définit un produit scalaire et on lui associe la norme : $\|\mathbf{x}\|_A = \sqrt{(A\mathbf{x}, \mathbf{x})}$.*

Theorem 1.68. *Soit $\mathbf{x}^*$ la solution de $A\mathbf{x} = \mathbf{b}$. Alors, $\mathbf{x}^*$ réalise le*

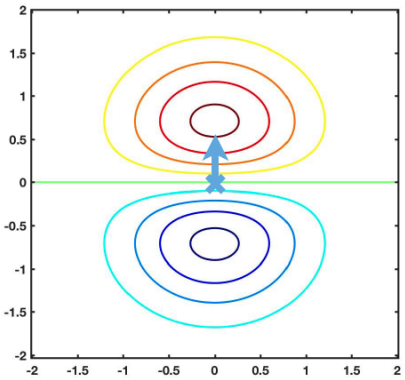
Démonstration.

□

Donc pour résoudre le problème linéaire on doit définir $\mathbf{x}^*$ minimisant $\mathcal{J}(\mathbf{x})$: en partant du point initial $\mathbf{x}^0 \in \mathbb{R}^n$ on doit choisir une direction appropriée pour s'approcher de la solution $\mathbf{x}^*$. Le choix optimale de la direction est inconnu à priori, mais on sait que le gradient d'une fonction est dirigé vers la direction de plus fort accroissement, donc une idée naturelle de choisir cette direction. Or, $\nabla\mathcal{J}(\mathbf{x}^k) = A\mathbf{x}^k - \mathbf{b}$, et donc la direction du gradient coïncide avec celle du résidu et on peut la calculer à partir de l'itérée $\mathbf{x}^k$. D'où l'algorithme de descente :

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha \underbrace{\left(A\mathbf{x}^k - \mathbf{b}\right)}_{=\nabla\mathcal{J}(\mathbf{x}^k)} \tag{5}$$

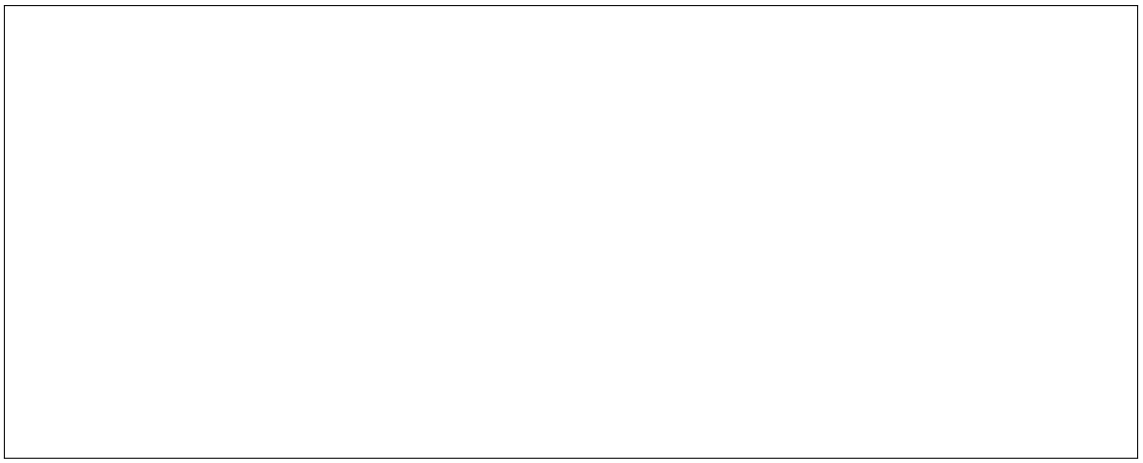
le choix de α (appelé le pas de descente) est très important. On peut réécrire l'itération sous la forme : $\mathbf{x}^{k+1} = \underbrace{(I - \alpha A)}_{M^{-1}N}\mathbf{x}^k + \alpha\mathbf{b}$.



Theorem 1.69. *L'algorithme* (5) *est convergent si et seulement si le pas vérifie* $0 < \alpha < \frac{2}{\lambda_n}$.

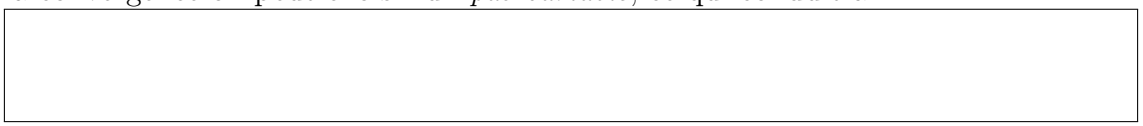
Démonstration.

□



□

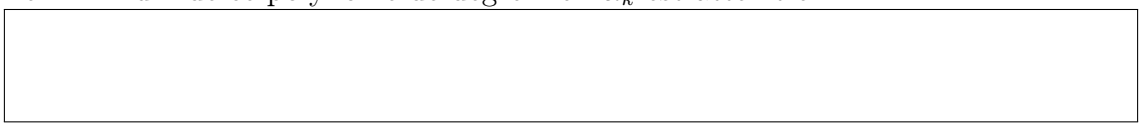
On constate que le choix de α a une influence sur la convergence de la méthode de descente. Le résultat du théorème précédent est bien théoriquement, mais en pratique on ne pourra pas s'en servir. Effectivement, il faudrait connaître la valeur propre maximal λ_n, ce qui n'est pas possible en temps raisonnable. Pour améliorer la convergence on peut choisir un *pas variable*, ce qui conduit à



On va alors choisir le pas a chaque étape afin de minimiser :

$$\begin{aligned}\mathcal{J}\left(\mathbf{x}^{k+1}\right) &= \frac{1}{2}\left(A\mathbf{x}^k - \alpha_k A\mathbf{r}^k, \mathbf{x}^k - \alpha_k \mathbf{r}^k\right) - \left(b, \mathbf{x}^k - \alpha_k \mathbf{r}^k\right) \\ &= \frac{1}{2}\alpha_k^2\left(A\mathbf{r}^k, \mathbf{r}^k\right) - \alpha_k \left\|\mathbf{r}^k\right\|_2^2 + \text{const.}\end{aligned}$$

Le minimum de ce polynôme de degré 2 en α_k est atteint en



En résumé, on a les deux algorithmes suivants :

Algorithme 7 : Pas fixe

$\mathbf{x} = \mathbf{x}^0$
$\mathbf{r}^0 = A\mathbf{x} - \mathbf{b}$
tant que $\|\mathbf{r}\| \geq \varepsilon\|\mathbf{r}^0\|$ **faire**
  $\mathbf{x} = \mathbf{x} - \alpha\mathbf{r}$
  $\mathbf{r} = A\mathbf{x} - \mathbf{b}$
fin

Algorithme 8 : Pas variable

$\mathbf{x} = \mathbf{x}^0$
$\mathbf{r}^0 = A\mathbf{x} - \mathbf{b}$
tant que $\|\mathbf{r}\| \geq \varepsilon\|\mathbf{r}^0\|$ **faire**
  $\alpha = (\mathbf{r}, \mathbf{r})/(A\mathbf{r}, \mathbf{r})$
  $\mathbf{x} = \mathbf{x} - \alpha\mathbf{r}$
  $\mathbf{r} = A\mathbf{x} - \mathbf{b}$
fin

En conclusion :

- Résoudre un système linéaire avec une méthode itérative consiste à construire, en partant d'une donnée initiale $\mathbf{x}^0$, une suite de vecteurs $\mathbf{x}^k$ convergeant vers la solution exacte quand $k \to \infty$;
- une méthode itérative converge si pour toute donnée initiale $\mathbf{x}^0$ on a $\mathbf{x}^k \to \mathbf{x}^*$ quand $k \to \infty$;

- une méthode itérative converge si et seulement si le rayon spectral de la matrice d'itération ($B = M^{-1}N$) est strictement plus petit que 1 ;
- les méthodes itératives traditionnelles sont celles de Jacobi et de Gauss-Seidel. Une condition suffisante de convergence est que la matrice soit à diagonale strictement dominante par ligne ;
- dans la méthode de descente, la convergence est accélérée à l'aide d'un paramètre ;

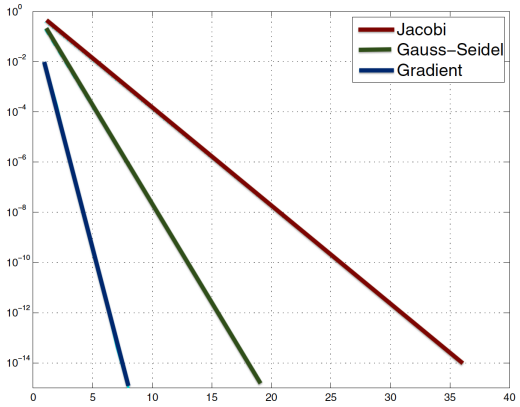


FIGURE 11 – Comparaison de convergence des méthodes (A. Quarteroni, R. Sacco, F. Saleri "Numerical Mathematics")

1.4.3 Les méthodes de Krylov

La méthode du gradient reprend souvent la même direction, on essaie donc de l'améliorer en forçant une direction différente à chaque itération.

Définition 1.70. Soit $\mathbf{x}_0$, on note $\mathbf{r}_0 = A\mathbf{x}_0 - b$. L'espace de Krylov d'ordre k associé est défini par

Si on considère la méthode de descente avec pas variable, on peut voir que $\mathbf{r}^k = \prod_{j=0}^{k-1}(I - \alpha_j A)\,\mathbf{r}^0$ et donc $\mathbf{r}^k = p_k(A)\mathbf{r}^0$ ou $p_k(A)$ est un polynôme de degré k, et on remarque immédiatement que $\mathbf{r}^k \in K_{k+1}(A, \mathbf{r}_0)$.

De façon similaire, l'itérée de la méthode de descente avec pas variable étant donné par $\mathbf{x}^k = \mathbf{x}^0 + \sum_{j=0}^{k-1} \alpha_j \mathbf{r}^j$, on a donc

Donc de manière plus général on peut réécrire la méthode de descente comme suit

$$\mathbf{x}^k = \mathbf{x}^0 + q_{k-1}(A)\mathbf{r}^0.$$

ou $q_{k-1}(A)$ est un polynôme en A. Ainsi on cherche la solution non pas directement dans l'espace $\mathbb{R}^n$ mais dans le sous-espace W_k. Le résultat suivant nous permet de dire que cette approche semble être raisonnable :

Proposition 1.71. *Soit $A \in \mathbb{M}_{n\times n}(\mathbb{R})$ et $\mathbf{v} \in \mathbb{R}^n$. La dimension de $K_k(A;\mathbf{v})$ est égal à k si et seulement si le degré $\deg_A v \geq k$. Ou $\deg_A \mathbf{v}$ est défini par*

$$\deg_A \mathbf{v} := \min\{\deg(P(A)) \text{ où } P(A)\mathbf{v} = \mathbf{0}\}$$

Et donc par théorème de Cayley-Hamilton $\deg_A \mathbf{v} \leq n$, il est possible qu'on cherche la solution dans l'espace de Krylov de dimension petite devant la dimension n.

Deux stratégie sont possible :

- chercher $\mathbf{x}^k \in W_k$ tel que $\mathbf{r}^k$ est orthogonal à $K_k(A;\mathbf{r}^0)$:
$$\mathbf{v}^T\left(\mathbf{b} - A\mathbf{x}^{(k)}\right) = 0 \quad \forall \mathbf{v} \in K_k\left(A;\mathbf{r}^{(0)}\right)$$
- chercher $\mathbf{x}^k \in W_k$ minimisant la norme de résidu $\|\mathbf{r}^k\|_2$
$$\left\|\mathbf{b} - A\mathbf{x}^{(k)}\right\|_2 = \min_{\mathbf{v}\in W_k} \|\mathbf{b} - A\mathbf{v}\|_2$$

Proposition 1.72. *Il existe un entier $k_0 \leq n$ critique tel que*

$$K_0(A;\mathbf{r}_0) \subsetneq K_1(A;\mathbf{r}_0) \subsetneq \ldots \subsetneq K_{k_0}(A;\mathbf{r}_0) = K_{k_0+1}(A;\mathbf{r}_0) = \ldots = K_n(A;\mathbf{r}_0)$$

Démonstration.

□

De plus, on peut montrer que k_0 satisfait la propriété suivante (voir preuve de la prop. suivante) :

$$k_0 = \min\{k \mid A^{-1}\mathbf{r}_0 \in K_n(A, \mathbf{r}_0)\}.$$

La proposition suivante établit le résultat général pour la solution exacte de $A\mathbf{x} = \mathbf{b}$:

Proposition 1.73. *La solution* $\mathbf{x} = A^{-1}\mathbf{b}$ *appartient à l'espace* $W_{k_0} = x^0 + K_{k_0}(A; \mathbf{r}_0)$

Démonstration.

□

1.4.4 Méthode de gradient conjugué

La méthode du gradient conjugué s'applique dans le cas où la matrice A est **symétrique définie positive**.

Soit $A = A^t > 0$. On définit $(\mathbf{x}^k)$

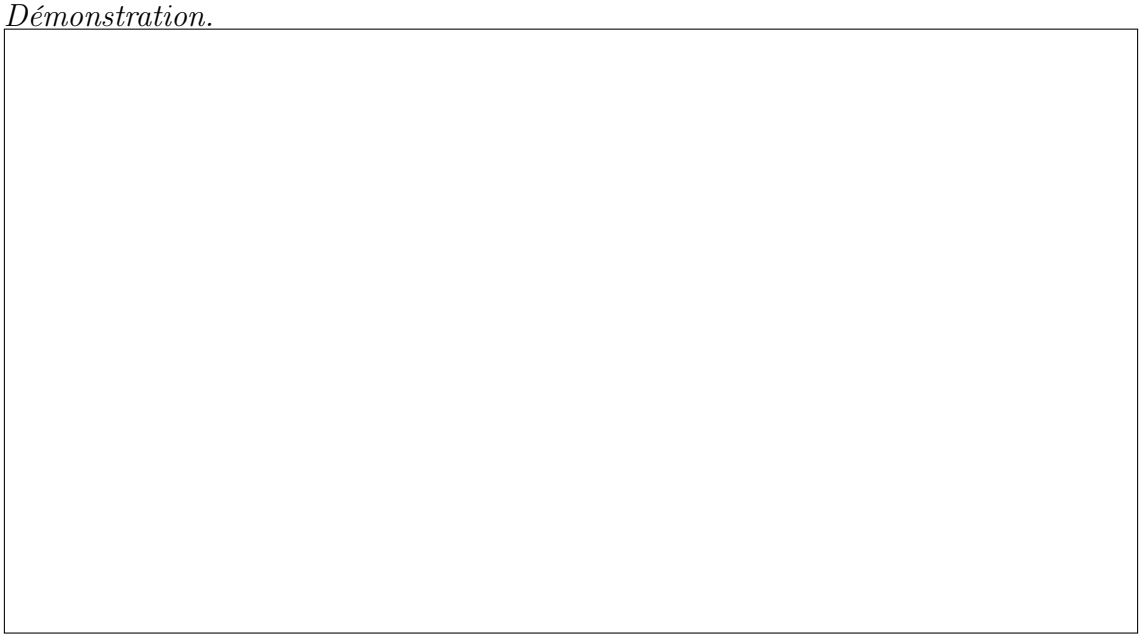
$$\begin{cases} \mathbf{x}^k \in W_k = \mathbf{x}^0 + K_k(A; \mathbf{r}_0) \\ \mathbf{v}^t \left(A\mathbf{x}^k - \mathbf{b}\right) = 0 \quad \forall \mathbf{v} \in K_k(A; \mathbf{r}_0) \end{cases} \tag{6}$$

Proposition 1.74. *Soit* k_0 *l'entier critique de la proposition 1.72 pour le quel* $K_{k_0}(A; \mathbf{r}_0) = K_{k_0+1}(A; \mathbf{r}_0)$. *Alors on a* $A\mathbf{x}^{k_0} = \mathbf{b}$ *et donc* $\mathbf{x}^{k_0}$ *est la solution exact.*

Version pratique du gradient conjugué : On considère une matrice symétrique définie positive $A \in \mathbb{M}_{n\times n}$ et $\mathbf{x}^0$, $\mathbf{b} \in \mathbb{R}^n$ fixés, et on pose $\mathbf{r}^0 := \mathbf{p}^0 := \mathbf{b} - A\mathbf{x}^0$. On définit alors les trois suites $(\mathbf{x}^k)_{k\geq 0}$, $(\mathbf{r}^k)_{k\geq 0}$, et $(\mathbf{p}^k)_{k\geq 0}$ par :

Alors la suite des $\mathbf{x}^k$ coïncide avec la suite des itérés de la méthode du gradient conjugué (6).

Démonstration.



Nous venons d'établir que la suite des $\mathbf{x}^k$ construite vérifie la première équation de (6). Pour conclure, il reste donc à démontrer qu'on a

$$\forall\, \mathbf{v} \in K_k, \quad \big(A\mathbf{x}^k - \mathbf{b}, \mathbf{v}\big) = 0 \quad \Leftrightarrow (\mathbf{r}^k, \mathbf{v}) = 0.$$

Pour ceci nous allons démontrer par récurrence sur $k \geq 0$ qu'on a

$$\forall\, j = 0 \ldots k-1, \quad \left\{ \begin{array}{rl} \big(\mathbf{r}^k, \mathbf{r}^j\big) & = 0 \\ \big(A\mathbf{p}^k, \mathbf{p}^j\big) & = 0 \end{array} \right. . \tag{7}$$

Il n'y a rien à démontrer pour $k = 0$. Supposons donc que ce soit vrai pour k, et démontrons que c'est encore vrai pour $k+1$. En utilisant la définition de $\mathbf{r}^k$ et de $\mathbf{p}^k$, on trouve

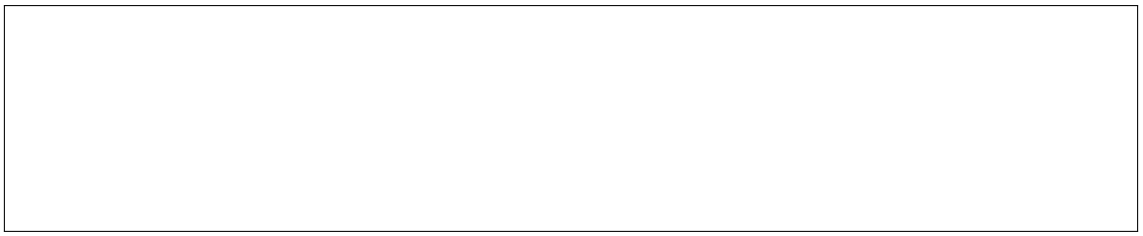
D'après l'hypothèse de récurrence on a :

$$\forall\, j = 0 \ldots k-1, \quad (\mathbf{r}^k, \mathbf{r}^j) = (A\mathbf{p}^k, \mathbf{p}^j) = (A\mathbf{p}^k, \mathbf{p}^{j-1}) = 0.$$

On vient donc de montrer que $(\mathbf{r}^{k+1}, \mathbf{r}^j) = 0$ pour $j < k$. Dans le cas $j = k$, on trouve (par le calcul ci-dessus) que $(\mathbf{r}^{k+1}, \mathbf{r}^k) = \|\mathbf{r}^k\|_2^2 - \alpha_k \|\mathbf{p}\|_A^2$, et alors l'expression de α_k nous permet de conclure que $(\mathbf{r}^{k+1}, \mathbf{r}^k) = 0$. En conclusion on a établi que

$$\forall\, j = 0 \ldots k, \quad (\mathbf{r}^{k+1}, \mathbf{r}^j) = 0. \tag{8}$$

Démontrons maintenant qu'on a également $(A\mathbf{p}^{k+1}, \mathbf{p}^j) = 0$ pour $j = 0 \ldots k$. Par definition de $\mathbf{p}^k$, et puisque $A\mathbf{p}^j = (\mathbf{r}^j - \mathbf{r}^{j+1})/\alpha_j$ (par definition de $\mathbf{r}^k$) on obtient



Dans la dernière égalité, le premier terme est nul puisque dans tous les cas on a $(\mathbf{r}^{k+1}, \mathbf{r}^j) = 0$. Par ailleurs si $j < k$, on a d'une part $(\mathbf{r}^{k+1}, \mathbf{r}^{j+1}) = 0$ d'après (8), et d'autre part $(A\mathbf{p}^k, \mathbf{p}^j) = 0$ d'après l'hypothèse de récurrence. On en déduit $(A\mathbf{p}^{k+1}, \mathbf{p}^j) = 0$ si $j < k$. Enfin, dans le cas où $j = k$, avec les expressions de α_k et β_k on déduit

Ceci clôt notre raisonnement par récurrence, et démontre que (7) est vrai pour tout $k \geq 0$.

La première propriété de (7) montre que $\mathbf{r}^j$, $j = 0 \dots k-1$ est une famille de k vecteurs linéairement indépendants appartenant à K_k, c'est donc une base de K_k. On voit donc que la propriété $(\mathbf{r}^k, \mathbf{r}^j) = 0 \quad \forall\, j = 0 \dots k-1$ revient à écrire $(\mathbf{r}^k, \mathbf{v}) = 0$ pour tout $\mathbf{v} \in K_k$. C'est précisément ce qui restait à démontrer pour conclure définitivement la preuve. □

La dimension $\dim(K_k(A; \mathbf{r}_0))$ des espaces de Krylov augmente avec k et on pourrait donc s'attendre à ce que le coût de chaque itération du gradient conjugué augmente avec k. Le résultat précédent démontre que chaque itération du gradient conjugué coûte $\mathcal{O}(n)$. C'est l'une des raisons du succès de la méthode du gradient conjugué.

Algorithme 9 : Gradient Conjugué

$\mathbf{x} = \mathbf{x}^0$
$\mathbf{r} = A\mathbf{x} - \mathbf{b}$
$\mathbf{p} = \mathbf{r}$
tant que $\|\mathbf{r}\| \geq \varepsilon \|\mathbf{r}^0\|$ **faire**
  $\alpha = \frac{(\mathbf{r},\mathbf{r})}{(A\mathbf{p},\mathbf{p})}$
  $\mathbf{x} = \mathbf{x} - \alpha\mathbf{p}$
  $\mathbf{r} = \mathbf{r} - \alpha A\mathbf{p}$
  $\beta = \frac{(\mathbf{r},\mathbf{r})}{\alpha(A\mathbf{p},\mathbf{p})}$
  $\mathbf{p} = \mathbf{r} + \beta\mathbf{p}$
fin

Le gradient conjugué ressemble à la descente de gradient, mais il est bien supérieur en efficacité ! On voit à quel point il est facile d'implémenter le gradient conjugué. La proposition suivante établit un résultat de convergence pour cette méthode itérative.

Theorem 1.75 (Admis). *Soit $A \in \mathbb{M}_{n\times n}$ une matrice symétrique définie positive. On note $\|\mathbf{x}\|_A^2 := (A\mathbf{x}, \mathbf{x})$. Étant donné $\mathbf{x}^0$, $\mathbf{b} \in \mathbb{R}^n$, si $\left(\mathbf{x}^k\right)_{k\geq 0}$ est la suite construite par la méthode du gradient conjugué*

$$\left\|\mathbf{x}^k - \mathbf{x}^*\right\|_A \leq 2\left(\frac{\sqrt{\text{cond}_2(A)}-1}{\sqrt{\text{cond}_2(A)}+1}\right)^k \left\|\mathbf{x}^0 - \mathbf{x}^*\right\|_{\text{A}} \quad \forall k \geq 0$$

1.4.5 Generalized Minimal Residual (GMRes)

La méthode du gradient conjugué (vu dans section précédente) est une méthode de projection orthogonale sur les espaces de Krylov où l'itérée $\mathbf{x}^k \in W_k$ est tel que $\mathbf{r}^k$ est orthogonal à $K_k(A; \mathbf{r}^0)$ $(A = A^t > 0)$:

$$\mathbf{v}^T\left(\mathbf{b} - A\mathbf{x}^{(k)}\right) = 0 \quad \forall\, \mathbf{v} \in K_k\left(A; \mathbf{r}^{(0)}\right)$$

Cette approche est efficace pour la résolution de systèmes symétriques définis positifs. Nous allons maintenant introduire une approche pour le cas général où $\mathbf{x}^k \in W_k$ minimise la norme de résidu $\|\mathbf{r}^k\|_2$:

$$\boxed{} \tag{9}$$

C'est le principe de la méthode Generalized Minimal Residual ou GMRes (cette construction correspond à une projection oblique sur les espaces de Krylov). Afin de préciser la construction de la méthode GMRes, on va chercher à construire une base orthonormale de $K_k(A, \mathbf{r_0}) := K_k$. On sait que $\{A^j\mathbf{r_0}\}_{j=0}^{k-1}$ est une base de K_k mais elle n'est pas orthonormale. On considère la base $\{v_j\}_{j=1}^k$ obtenue à partir de $\{A^j\mathbf{r_0}\}_{j=0}^{k-1}$ suivant un procédé d'orthonormalisation de Gram-Schmidt selon les formules

$$\begin{cases} \mathbf{v}_1 = \mathbf{r_0}/\|\mathbf{r_0}\|_2 \\ \mathbf{w}_{j+1} = A\mathbf{v}_j - \sum_{i=1}^j (A\mathbf{v}_j, \mathbf{v}_i)\mathbf{v}_i = A\mathbf{v}_j - \sum_{i=1}^j \mathbf{v}_i\mathbf{v}_i^t A\mathbf{v}_j \\ \mathbf{v}_{j+1} = \mathbf{w}_{j+1}/\|\mathbf{w}_{j+1}\|_2 \end{cases} \tag{10}$$

On rappelle que la procédure d'orthonormalisation de Gram-Schmidt s'avère instable si l'on ne fait pas attention à la manière dont est calculée concrètement la somme à la deuxième ligne. Le procédé d'Arnoldi, présenté ci-dessous, donne algébriquement le même résultat que la procédure de Gram-Schmidt mais est plus stable numériquement.

Algorithme 10 : Algorithme d'Arnoldi

$\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}^0$
$\mathbf{v}_1 = \mathbf{r_0}/\|\mathbf{r_0}\|_2$
pour $j = 1$ *à* k **faire**
  $\mathbf{w} = A\mathbf{v}_j$ // Dans G.S. classique on aurait pris $w = A^j\mathbf{r}_0$.
  pour $i = 1$ *à* j **faire**
    $\mathbf{w} = \mathbf{w} - (\mathbf{w}, \mathbf{v}_i)\mathbf{v}_i$
  fin
  $\mathbf{v}_{j+1} = \mathbf{w}/\|\mathbf{w}\|_2$
fin

Introduisons quelques notations. On notera par

$$V_k = [\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \ldots, \mathbf{v}_k]$$

la matrice orthogonale d'ordre $n \times k$ et $\bar{H}_k$ une matrice de Heissenberg de taille $(k+1) \times k$ (une ligne de plus que de colonnes) définie par

C'est à dire, de la forme

$$\bar{H}_k = \begin{pmatrix} h_{11} & h_{12} & h_{13} & h_{14} & \ldots & h_{1k} \\ h_{21} & h_{22} & h_{23} & h_{24} & \ldots & h_{2k} \\ 0 & h_{32} & h_{33} & h_{34} & \ldots & h_{3k} \\ \vdots & 0 & \ddots & \ddots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & h_{k,k-1} & h_{k,k} \\ 0 & 0 & 0 & \ldots & 0 & h_{k+1,k} \end{pmatrix}$$

La sous matrice formée des k premières lignes sera notée par H_k. L'intérêt de ces matrices provient du fait qu'elles permettent une écriture matricielle du procédé d'Arnoldi :

$$AV_k = V_{k+1}\bar{H}_k.$$

Par définition de la méthode GMRES on cherche $\mathbf{x}^k \in W_k = \mathbf{x}_0 + K_k$, en choisissant la base $\{\mathbf{v}_j\}_{j=1}^k$ qu'on vient de construire, on réécrit

$$\mathbf{x}^k = \mathbf{x}^0 + V_k\mathbf{y}, \text{ où } \mathbf{y} \in \mathbb{R}^k.$$

On peut donc réécrire le problème de minimisation (9) de manière équivalente :

En résumé, l'algorithme de GMRes à chaque itération k est de la forme :

Algorithme 11 : GMRes Étape k : idée générale

$\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}^0$

$\beta = \|\mathbf{r_0}\|_2$

$(V_{k+1}, \overline{H}_k) \leftarrow$ algorithme d'Arnoldi $(A, \mathbf{r_0})$

$\mathbf{y} \leftarrow$ résolution du problème de moindres carrés $(\bar{H}_k, \beta)$ $*$

$\mathbf{x}^k = \mathbf{x}^0 + V_k\mathbf{y}$

Il reste à discuter de la manière dont on peut résoudre le problème de minimisation. Comme $\ker(\bar{H}_k) = \{0\}$, la matrice $\bar{H}_k^t\bar{H}_k$ est symétrique définie positive, et on a une équivalence :

$$\min_{y \in K_k} \|\bar{H}_k\mathbf{y} - \beta\mathbf{e}_1\|_2 \iff \bar{H}_k^t\bar{H}_k\mathbf{y} = \beta\bar{H}_k^t\mathbf{e}_1$$

et on peut appliquer la méthode Cholesky ou la décomposition QR. Notons que compte tenu de la structure de $\bar{H}_k^t$, on a juste à "éliminer" à l'aide d'une matrice orthogonale les termes sous diagonaux pour obtenir R. On procède par la méthode de Givens en utilisant les matrices de rotation plane (voir TD).

Pour garantir une précision satisfaisante, on peut être amené à augmenter k substantiellement, et ceci est réellement problématique car le coût du calcul croît très vite avec k, typiquement en $\mathcal{O}(k^3)$. Cette stratégie telle quelle n'est donc pas raisonnable en général. La solution est d'adapter une variante de la méthode appelé "restarted GMRes".

Résumé : Une méthode de Krylov consiste à construire itérativement (en k) une solution $\mathbf{x}^k \in \mathbf{x}^0 + K_k(A, \mathbf{r}_0)$. Elle est définie par :

- le choix de la base de K_k qu'on utilise.
- le critère d'optimalité qu'on souhaite optimisé.
- La solution de notre problème $\mathbf{x} \in \mathbf{x}^0 + K_{k_0}(A, \mathbf{r}_0)$. Par conséquent, une méthode de Krylov converge en k_0 (critique) itérations. Ces méthodes corres-

pondent donc à des méthodes directes, mais elles sont utilisés comme des méthodes itératives, le but étant de faire moins d'itérations que k_0.

1.4.6 Principe du préconditionnement

L'idée est de résoudre un système équivalent

$$MA\mathbf{x} = M\mathbf{b}.$$

où on souhaite choisir M tel que

$$\mathrm{cond}_2(M^{-1}A) < \mathrm{cond}_2(A).$$

À priori, le meilleur choix de préconditionneur est $M = A^{-1}$ ce qui évidemment n'est pas applicable en pratique car trop coûteux. On construit en général une approximation M de A^{-1} de sorte que le calcul de $MA\mathbf{x}$ soit peu coûteux.

Exemple (Quelques préconditionneurs)**.**
Jacobi : On choisit dans ce cas $M = D^{-1}$ où D est la diagonale de A.
Gauss-Seidel : On choisit dans ce cas $M = (D - E)^{-1}$ (E est la partie triangulaire strictement inférieure de A). ▼

On pourrait penser qu'un tel procédé pose problème pour appliquer à la méthode du gradient conjugué car même si M et A sont symétrique définie positives, il n'y pas de raison que $M^{-1}A$ l'est. Afin de préserver cette propriété on utilise un préconditionnement symétrique :

$$M^tAM\widetilde{x} = M^tb \quad \text{où} \quad \widetilde{x} = M^{-1}x.$$

1.5 Méthode directe ou itérative ?

Tout dépend du problème. En général, les méthodes directes (surtout quand elles sont implémentées de manière sophistiquée) sont plus efficaces que les méthodes itératives quand ces dernières ne sont pas utilisées avec des préconditionneurs performants. Cependant, elles sont plus sensibles au conditionnement de la matrice et peuvent nécessiter une mémoire importante.

Il est également utile de souligner que les méthodes directes ont explicitement besoin des coefficients de la matrice, contrairement aux méthodes itératives. Pour ces dernières, il est seulement nécessaire de pouvoir calculer le produit matrice-vecteur pour des vecteurs arbitraires. Cette propriété est particulièrement intéressante dans les problèmes où la matrice n'est pas construite explicitement.

2 Résolution des équations non linéaires

Dans ce chapitre, on s'intéressera au problème non linéaire suivant :

$$\text{Trouver } \mathbf{x}^* \in \mathbb{R}^n, \quad \text{t.q.} \quad F(\mathbf{x}^*) = \mathbf{0} \tag{11}$$

où a priori la fonction F peut être vectorielle, i.e. à valeurs dans R^p.
Remarque. Soulignons que si $p > n$, à priori le problème n'admet pas de solution dans le cas général et on s'intéresse plus tôt au problème de minimisation :

$$\text{Trouver } \mathbf{x} \in \mathbb{R}^n, \quad \text{t.q.} \quad \min_{\mathbf{x}\in\mathbb{R}^n} F(\mathbf{x})$$

Dans le chapitre précédent, on a étudié des méthodes de résolution du système dans le cas particulier $F(\mathbf{x}) = A\mathbf{x} - \mathbf{b}$, $A \in \mathbb{M}_n(\mathbb{R})$, $\mathbf{b} \in \mathbb{R}^n$. On va maintenant étendre le champ d'étude au cas où F n'est pas forcément affine. On étudiera les méthodes pour la résolution de (11) suivantes :

- les méthodes de point fixe
- les méthodes de type Newton
- les méthode de la fausse position

Quelques problèmes types :
(Équation d'état d'un gaz). Nous voulons déterminer le volume V occupé par un gaz dont la température est T et dont la pression est p. L'équation d'état est donnée par

$$\left[p + a(N/V)^2\right](V - Nb) = kNT$$

où a et b sont deux coefficients qui dépendent du gaz considéré, N est le nombre de molécules contenues dans le volume V et k est la constante de Boltzmann. Nous devons donc résoudre une équation non linéaire dont la racine est V.

(Dynamique des populations). Pour étudier une population (par ex. une population de poissons rouges), on considère l'équation

$$\phi(\mathbf{x}) = \mathbf{x}R(\mathbf{x})$$

qui donne une relation entre le nombre d'individus à la génération $\mathbf{x}$ et le nombre d'individus à la génération suivante $\phi(\mathbf{x})$. La fonction $R(\mathbf{x})$ modélise la vitesse d'évolution de la population considérée et peut être choisie de différentes manières. Parmi les plus connues, on peut citer prédateurs-proies avec saturation, modèle de Malthus ou de croissance avec ressources limitées.

(Résolution d'EDO par des méthodes implicites). Les équations différentielles sont utilisées pour construire des modèles mathématiques de processus d'évolution physiques et biologiques, par exemple pour l'étude de la radioactivité, la mécanique céleste ou la dynamique des populations :

$$Y'(x) = f(x, Y), \quad x \in [a, b]$$

L'algorithme de résolution par une méthode implicite (*voir cours d'Analyse numérique (S6)*) :

$$Y_{n+1} = Y_n + \Delta x f(x_n, Y_{n+1}).$$

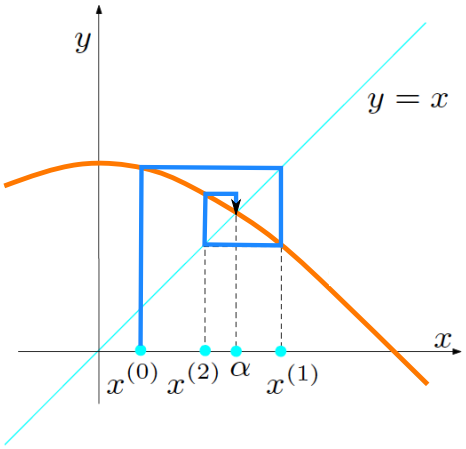
2.1 Méthode de point fixe

On cherche à résoudre le problème 11 dans le cas où F s'écrit $F(\mathbf{x}) = G(\mathbf{x}) - \mathbf{x}$, avec $G : \mathbb{R}^n \to \mathbb{R}^n$ une fonction. On a alors le problème suivant à résoudre

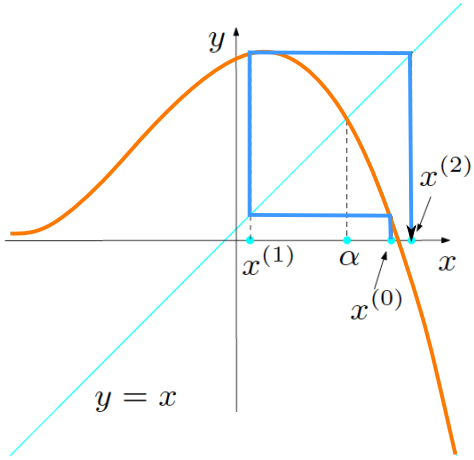
Si un tel $\mathbf{x}$ existe, on dit que c'est un point fixe de G et on peut essayer de le calculer à l'aide de l'algorithme suivant où $\mathbf{x}^0$ est une donnée initiale :

$$\mathbf{x}^{k+1} = G(\mathbf{x}^k), \quad k \geq 0.$$

Cet algorithme est appelé méthode de point fixe ou itérations de point fixe et on dit que G est la fonction d'itération.



Convergence



Divergence

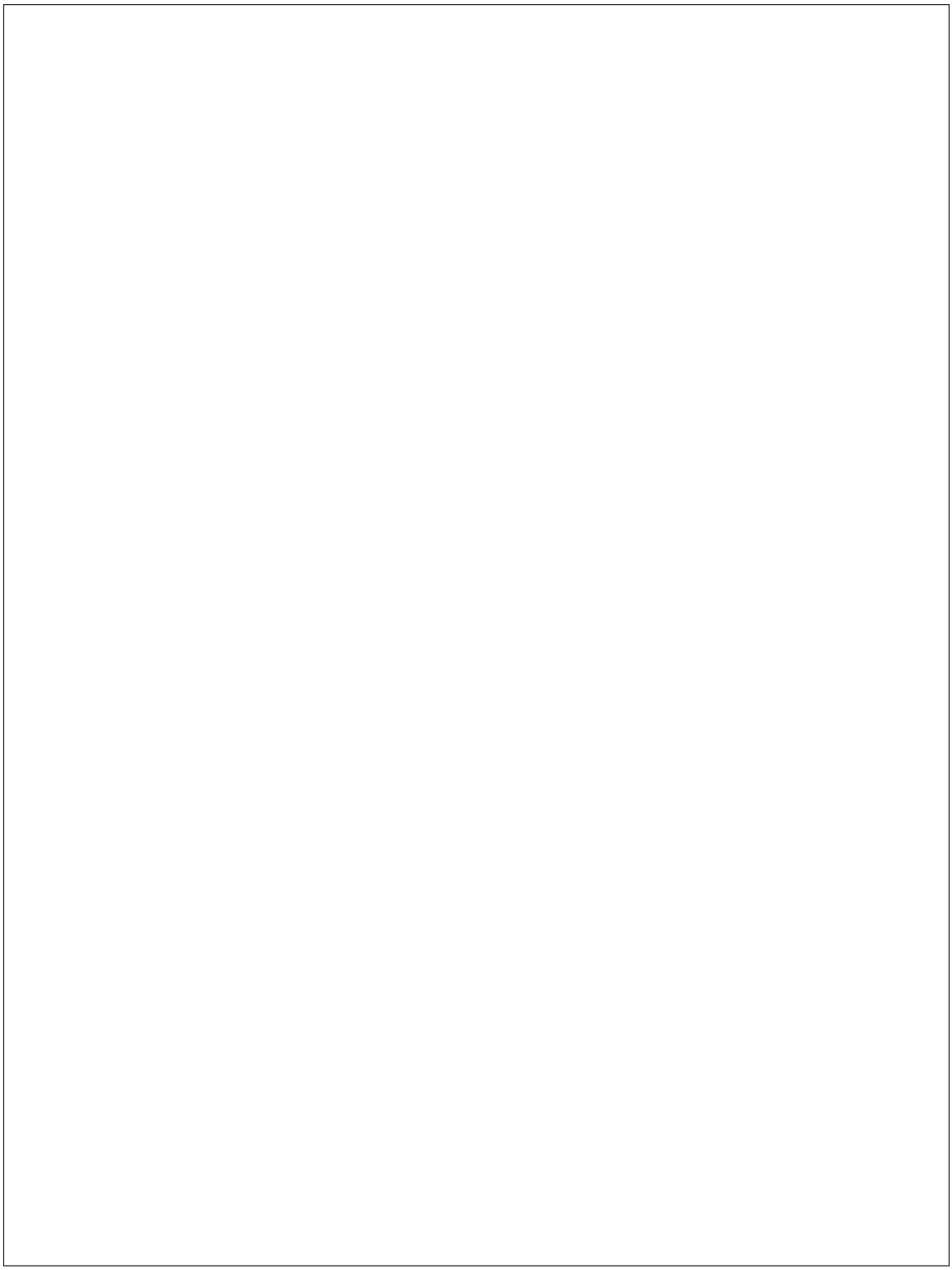
Les itérations de point fixe peuvent ne pas converger, comme le montre la figure ci-dessus. Donc il nous faut établir le résultat de convergence.

Theorem 2.1. *On suppose que l'application $G : \mathbb{R}^n \to \mathbb{R}^n$ vérifie la propriété*

$$\exists\, K \in]0,1[,\ \forall\, \mathbf{x}, \mathbf{y} \in \mathbb{R}^n \quad \|G(\mathbf{x}) - G(\mathbf{y})\| \leq K\|\mathbf{x} - \mathbf{y}\|.$$

Alors l'application G admet un unique point fixe et la suite $\mathbf{x}^{k+1} = G(\mathbf{x}^k)$ converge vers ce point fixe.

Démonstration.



□

Définition 2.2. On dit que l'application G est différentiable au point $\mathbf{x}_0$ s'il existe une application linéaire $d_{\mathbf{x}_0}G$ vérifiant :

Cette application est appelée la différentielle et elle généralise la notion de dérivée.

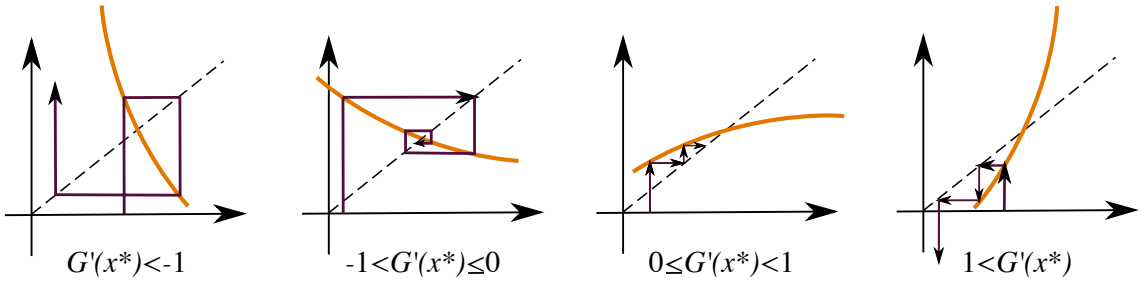
L'application linéaire $d_{\mathbf{x}_0}G$ peut être représentée par sa matrice Jacobienne :

$$\left[\frac{\partial G_i}{\partial x_j}(\mathbf{x})\right]_{i,j=1}^{n} = \begin{bmatrix} \frac{\partial G_1}{\partial x_1}(\mathbf{x}) & \frac{\partial G_1}{\partial x_2}(\mathbf{x}) & \cdots & \frac{\partial G_1}{\partial x_n}(\mathbf{x}) \\ \frac{\partial G_2}{\partial x_1}(\mathbf{x}) & \cdots & \cdot & \frac{\partial G_2}{\partial x_n}(\mathbf{x}) \\ \vdots & & & \vdots \\ \frac{\partial G_n}{\partial x_1}(\mathbf{x}) & \frac{\partial G_n}{\partial x_2}(\mathbf{x}) & \cdots & \frac{\partial G_n}{\partial x_n}(\mathbf{x}) \end{bmatrix}$$

Theorem 2.3. *Si l'application G admet un point fixe $\mathbf{x}^* \in \mathbb{R}^n$, et différentiable en ce point et vérifie $\|d_{\mathbf{x}^*}G\| < 1$, alors il existe un voisinage V tel que $\forall\, \mathbf{x}_0 \in V$, la suite $\mathbf{x}^{k+1} = G(\mathbf{x}^k)$ converge vers le point fixe.*

Démonstration.

□



Algorithme 12 : Algorithme de point fixe

$\mathbf{x}_1 = G(\mathbf{x}_0)$

tant que $\|\mathbf{x}_0 - \mathbf{x}_1\| \geq \varepsilon$ **faire**
  $\mathbf{x}_0 = \mathbf{x}_1$
  $\mathbf{x}_1 = G(\mathbf{x}_0)$
fin

Remarque. Lorsque le test de convergence est satisfait, on a alors

2.2 Vitesse de convergence

Définition 2.4. On dit que la suite $(\mathbf{x}^k)_{k\geq 0}$ est d'ordre r s'il existe une constante $C > 0$ telle que :

$$\lim_{k\to\infty} \frac{\|\mathbf{x}^{k+1} - \mathbf{x}^*\|}{\|\mathbf{x}^k - \mathbf{x}^*\|^r} = C$$

Remarque. Cela revient à dire $\|\mathbf{x}^{k+1} - \mathbf{x}^*\| \sim C\|\mathbf{x}^k - \mathbf{x}^*\|^r$. Cette notion permet de "mesurer" la vitesse de convergence d'une suite (quand elle converge).

Si les hypothèses du théorème 2.3 sont vérifiées, on a alors :

$$\mathbf{x}^{k+1} = G(\mathbf{x}^* + \mathbf{x}^k - \mathbf{x}^*) = \mathbf{x}^* + d_{\mathbf{x}^*}G(\mathbf{x}^k - \mathbf{x}^*) + \Phi(\mathbf{x}^*, \mathbf{x}^k)\mathbf{x}^*.$$

d'où $\|\mathbf{x}^{k+1} - \mathbf{x}^*\| = \|d_{\mathbf{x}^*}G(\mathbf{x}^k - \mathbf{x}^*) + \Phi(\mathbf{x}^*, \mathbf{x}^k)\|$ et donc on déduit que la suite est au moins d'ordre 1.

Theorem 2.5. *Si l'application G admet un point fixe, est r fois continûment différentiable et vérifie :*

$$d_{\mathbf{x}^*}G = \ldots = d^{r-1}_{\mathbf{x}^*}G = 0 \quad \text{et} \quad d^r_{\mathbf{x}^*}G \neq \mathbf{0}.$$

alors il existe un voisinage V t.q. $\forall\, \mathbf{x}_0 \in V$, la suite générée par $\mathbf{x}^{k+1} = G(\mathbf{x}^k)$ converge à l'ordre r vers le point fixe.

Démonstration (cas $F : \mathbb{R} \to \mathbb{R}$).

□

2.3 Méthode de Newton

On présente la méthode de Newton pour la résolution du problème 11 dans le cas général. Donnons l'idée de cette méthode dans le cas particulier $f : \mathbb{R} \to \mathbb{R}$. Soit $f \in C^3(\mathbb{R}, \mathbb{R})$ et $x^* \in \mathbb{R}$ tel que $f(x^*) = 0$. On cherche à construire une suite $\left(x^k\right)_{k\in\mathbb{N}} \in \mathbb{R}^n$ qui converge vers x^* de manière quadratique (i.e. vitesse de convergence d'ordre $r = 2$). On pose

et on a $g(x) = x \Leftrightarrow f(x) = 0$. Si par miracle on a $g'(x^*) = 0$, alors la méthode de point fixe sur g va donner une suite $\left(x^{(k)}\right)_{k\in\mathbb{N}}$ (pour $x^0 \in V$ donné par le théorème 2.5) telle que $x^{(k)} \to x^*$ de manière au moins quadratique. Or on a

$$g'(x) = 1 - h'(x)f(x) - f'(x)h(x)$$

et donc $g'(x^*) = 1 - f'(x^*)h(x^*)$. Il suffit donc de prendre h tel que $h(x^*) = \frac{1}{f'(x^*)}$, ce qui est possible si $f'(x^*) \neq 0$.

En résumé, si $f \in C^3(\mathbb{R}, \mathbb{R})$ est telle que $f'(x^*) \neq 0$ et $f(x^*) = 0$, on peut construire, pour x assez proche de x^*, la fonction $g \in C^2(\mathbb{R}, \mathbb{R})$ définie par

$$g(x) = x - \frac{f(x)}{f'(x)}$$

Grâce au théorème 2.5, il existe un voisinage V tel que si $x^0 \in V$ alors la suite définie par

$$x^{k+1} = g(x^k) = x^k - \frac{f(x^k)}{f'(x^k)},$$

converge vers x^* de manière au moins quadratique.

Pour retenir la construction de la méthode de Newton, remarquons que la suite de Newton peut s'obtenir naturellement en remplaçant l'équation $f(x^*) = 0$ par $f(x^{k+1}) = 0$, et $f(x^{k+1})$ par le développement limité en x^k :

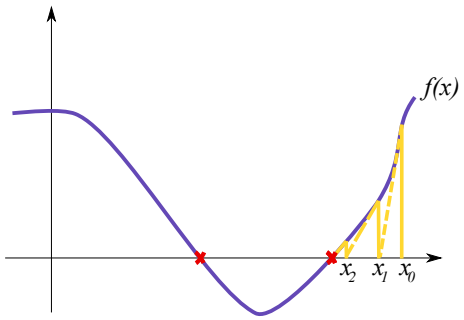
$$0 = f(x^{k+1}) \simeq f(x^k) + f'(x^k)(x^{k+1} - x^k).$$

Algorithme 13 : Méthode de Newton

$x_1 = x_0 - f(x_0)/f'(x_0)$

tant que $\|x_0 - x_1\| \geq \varepsilon$ **faire**
  $x_0 = x_1$
  $x_1 = x_0 - f(x_0)/f'(x_0)$
fin

Remarque. Le point x^{k+1} est définie comme le point où la tangente de f en x^k s'annule.



Remarque. La méthode se généralise au cas d'une fonction $F : \mathbb{R}^n \to \mathbb{R}^n$ de la façon suivante :

$$G(\mathbf{x}) = \mathbf{x} - H(\mathbf{x})F(\mathbf{x}) \Rightarrow d_{\mathbf{x}^*}G(\mathbf{x}^*) = \mathbf{0} = I - H(\mathbf{x}^*)d_{\mathbf{x}^*}F(x^*) \iff H(\mathbf{x}^*) = (d_{\mathbf{x}^*}F)^{-1}(\mathbf{x}^*)$$

et donc on a l'itération de Newton dans le cas général :

Revenons maintenant sur la vitesse de convergence de la méthode de Newton. On rappelle que, en générale, la suite $(\mathbf{x}^k)_{k\geq 0}$ est de l'ordre r (où la convergence est de l'ordre r) s'il existe $C \in \mathbb{R}^+$ telle que

$$\lim_{k\to\infty} \frac{\|\mathbf{x}^{k+1} - \mathbf{x}^*\|}{\|\mathbf{x}^k - \mathbf{x}^*\|^r} = C. \tag{12}$$

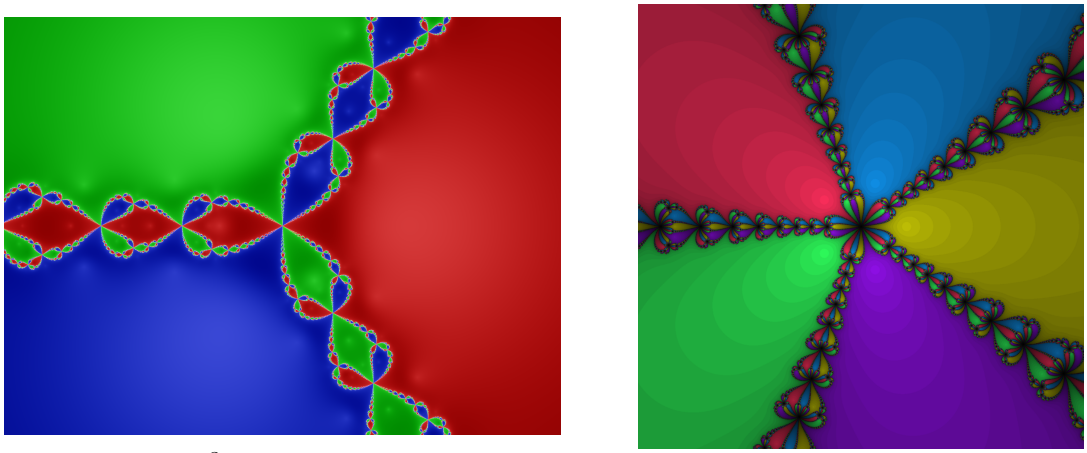
Remarquons d'abord que si une suite $(\mathbf{x}^k)_{k\geq 0}$ converge vers $\mathbf{x}^*$ lorsque k tend vers l'infini, alors on a forcément $C \leq 1$.

Dans le cas où $r = 1$, la convergence est dite

- *sous-linéaire*, si $C = 1$.
- *au moins linéaire*, si $C \in [0, 1[$.
- *linéaire*, si $C \in]0, 1[$.
- *super-linéaire*, si $C = 0$ (et on peut établir (12) pour $r > 1$)

La convergence linéaire (au sens donné ci-dessus), est déjà une convergence très rapide. Par exemple, les suites géométriques définies par $x^k = C^k$ avec $C \in]0, 1[$ sont des suites qui convergent linéairement (vers 0), car elles vérifient évidemment la définition. Si $r = 2$, la convergence est quadratique, elle est encore plus rapide que linéaire.

On a vu déjà que les méthodes de résolution d'équations non linéaires peuvent être divergente. Il faut penser à bien choisir x_0. C'est aussi le cas de la méthode de Newton, introduite ci-dessus. Lorsqu'elle converge, elle converge très vite (comme on a vu par construction la vitesse de convergence est quadratique). Mais lorsqu'elle diverge, elle diverge aussi très vite.



(a) $z^3 - 1 = 0,\ z \in \mathbb{C}$ (b) $z^5 - 1 = 0,\ z \in \mathbb{C}$

FIGURE 12 – Bassins d'attraction des racines du polynôme (en couleur).

2.4 Méthode de la fausse position

On rappelle que dans la méthode de Newton on génère la suite x^k ainsi :

$$x^{k+1} = x^k - \frac{f(x^k)}{f'(x^k)}.$$

Pour pouvoir utiliser la méthode, on calcule la dérivée f'. Dans certaines situations, ce calcul peut être coûteux voir impossible. Une idée est alors d'utiliser l'approximation de la dérivée :

$$f'(x^k) \simeq \frac{f(x^k) - f(x^{k-1})}{x^k - x^{k-1}}.$$

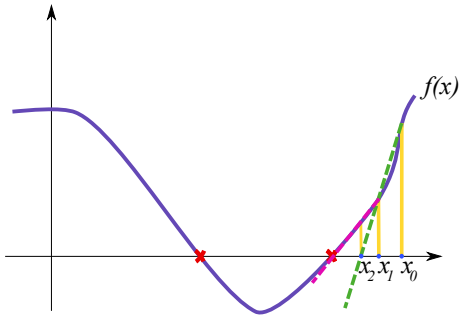
On en déduit alors *la méthode de la fausse position*

$$x^{k+1} = x^k - \frac{x^k - x^{k-1}}{f(x^k) - f(x^{k-1})} f(x^k)$$

Remarque.

1. À la différence de la méthode de Newton, il faut deux points pour initialiser la méthode de la fausse position.
2. L'itérée x^{k+1} correspond au point où s'annule la droite définie par :

$$y = \frac{f(x^k) - f(x^{k-1})}{x^k - x^{k-1}}(x - x^k) + f(x^k).$$



Theorem 2.6. *Si f est 2 fois continûment différentiable, admet un 0 en x^* et $f'(x^*) \neq 0$, alors il existe un voisinage V dans lequel la méthode de fausse position converge et est d'ordre $(1+\sqrt{5})/2$ au moins.*

Algorithme 14 : Méthode de la fausse position

$x_1 = x_0 - f(x_0)/f'(x_0)$

tant que $\|x_0 - x_1\| \geq \varepsilon$ **faire**

  $x_2 = x_1 - (x_1 - x_0)f(x_1)/(f(x_1) - f(x_0))$

  $x_0 = x_1$

  $x_1 = x_2$

fin

Remarque. Pour la mise en œuvre de la méthode, en particulier si le calcul de $f(x)$ est coûteux, il est intéressant de remarquer qu'à chaque étape une seule évaluation de f est nécessaire.

2.5 Conclusion

1. On est assuré de la convergence des méthodes de Newton et de la fausse position seulement lorsque l'initialisation est dans un voisinage proche de la solution. Il faut donc avoir à priori une bonne solution approchée.
2. Comparée à la méthode de Newton, où la convergence vers la racine simple est au moins quadratique, et plus précisément est de l'ordre p où p est le plus petit entier $p \geq 2$ tel que $f^{(p)}(x^*) \neq 0$, la méthode de la fausse position dans la même configuration convergera à l'ordre $\frac{1+\sqrt{1+4(p-1)}}{2}$.
3. Lorsque la racine est double les méthodes de Newton et de la fausse position convergent mais à un ordre plus petit.
4. On peut adapter la méthode de Newton pour résoudre des problèmes de minimisation.