

Mathématiques assistées par ordinateur

Chapitre 9 : Calcul matriciel et algèbre linéaire

Michael Eisermann

Mat249, DLST L2S4, Année 2008-2009

www-fourier.ujf-grenoble.fr/~eiserm/cours#mao

Document mis à jour le 6 juillet 2009



Sommaire

- 1 Résolution de systèmes d'équations linéaires
 - Systèmes d'équations linéaires, l'algorithme de Gauss
 - Calcul matriciel : addition, multiplication, inversion, déterminant
 - Stabilité numérique, conditionnement d'une matrice
- 2 Réduction des endomorphismes
 - Espaces vectoriels et applications linéaires
 - Vecteurs propres, polynôme caractéristique
 - Polynôme minimal, méthodes de calcul
- 3 Méthodes approchées itératives
 - La méthode de la puissance
 - La méthode des itérations inverses
 - Matrices hermitiennes et symétriques
- 4 Comment fonctionne Google ?
 - Comment mesurer l'importance d'une page web ?
 - Le modèle PageRank : marche aléatoire sur le web
 - Existence, unicité, et calcul de la solution

Systèmes d'équations linéaires

Dans la suite nous fixons un corps \mathbb{K} (par exemple \mathbb{Q} , \mathbb{R} , ou \mathbb{C}).

Nous souhaitons résoudre un **système d'équations linéaires** :

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & = & y_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n & = & y_2 \\ & \vdots & \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n & = & y_m \end{cases}$$

Ici les coefficients $a_{ij} \in \mathbb{K}$ et $y_i \in \mathbb{K}$ sont donnés.

L'objectif est de trouver toutes les solutions $x_i \in \mathbb{K}$.

On écrit ce système plus succinctement comme $Ax = y$ où

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}, \quad x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}, \quad y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}.$$

Cette structuration est bien plus qu'une notation commode :

⇒ Programmes = structure des données + algorithmes !

⇒ Calcul matriciel : addition, multiplication, inverse, déterminant, ...

Matrices triangulaires et échelonnées

On cherche à résoudre un système d'équations linéaires $Ax = y$.

Si A est **triangulaire**, la solution est immédiate : il suffit de remonter.

$$A = \begin{pmatrix} 1 & * & * & * \\ 0 & 1 & * & * \\ 0 & 0 & 1 & * \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{voire} \quad A = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Plus généralement la solution est facile si A est **échelonnée** :

$$A = \begin{pmatrix} 1 & * & * & * & * & * & * \\ 0 & 0 & 1 & * & * & * & * \\ 0 & 0 & 0 & 1 & * & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad \text{voire} \quad A = \begin{pmatrix} 1 & * & 0 & 0 & * & * & 0 \\ 0 & 0 & 1 & 0 & * & * & 0 \\ 0 & 0 & 0 & 1 & * & * & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Dans cet exemple l'équation $Ax = y$ n'a pas de solution si $y_5 \neq 0$.

Si $y_5 = 0$, les solutions x telles que $Ax = y$ forment un sous-espace affine de \mathbb{K}^7 de dimension 3 : on peut choisir x_2, x_5, x_6 arbitrairement, les valeurs x_7, x_4, x_3, x_1 s'en déduisent en remontant.

« *Solution générale = solution particulière + solutions homogènes.* »

Opérations élémentaires

Objectif : mettre sous forme échelonnée une matrice donnée

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} = \begin{pmatrix} L_1 \\ \vdots \\ L_m \end{pmatrix}.$$

Pour ceci on effectue des **opérations élémentaires** sur les lignes :

- $L_i \leftrightarrow L_j$ échanger la ligne i et la ligne j ,
- $L_i \leftarrow \lambda L_i$ multiplier la ligne i par un facteur inversible λ ,
- $L_i \leftarrow L_i + \lambda L_j$ ajouter un multiple de la ligne j à la ligne i .

À noter que chacune de ces opérations est **inversible** !

Ainsi on les appelle aussi **transformations d'équivalence**.

◆ Pour résoudre $Ax = y$ on effectue ces opérations sur $(A | y)$: c'est la matrice A agrandie en accolant le vecteur colonne y .

Le fait que nos opérations transformant $(A | y)$ en $(A' | y')$ soient inversibles garantit que les systèmes linéaires $Ax = y$ et $A'x = y'$ ont exactement les mêmes solutions.

L'algorithme de Gauss : idée

D'abord on choisit un pivot $a_{j1} \neq 0$ et on le place en tête :

$$\begin{pmatrix} a_{11} & a_{21} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix} \xrightarrow{L_1 \leftrightarrow L_j} \begin{pmatrix} a_{11} & a_{21} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

Ensuite on annule la première colonne :

$$\begin{array}{c} \xrightarrow[\substack{a_{11} \neq 0 \\ L_1 \leftarrow a_{11}^{-1} L_1}]{} \\ \xrightarrow[\substack{j=2, \dots, m \\ L_j \leftarrow L_j - a_{j1} L_1}]{} \end{array} \begin{pmatrix} 1 & a_{21} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$
$$\begin{pmatrix} 1 & a_{21} & \dots & a_{1n} \\ 0 & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ 0 & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

On itère cette méthode sur la sous-matrice $\{2, \dots, m\} \times \{2, \dots, n\}$.

L'algorithme de Gauss : formulation détaillée

L'algorithme de Gauss transforme une matrice A en une matrice échelonnée à l'aide des opérations élémentaires sur les lignes :

- 1 On initialise $c \leftarrow 1$ et $\ell \leftarrow 1$. Ici c est le numéro de la colonne, et ℓ est le numéro de la ligne à partir de laquelle on cherche un pivot.
- 2 Si $c > n$ ou $\ell > m$, alors on arrête.
- 3 Si la colonne c n'a que des coefficients nuls à partir de la ligne ℓ , on incrémente c et on passe à l'étape 2.
- 4 On choisit un pivot a_{kc} non nul, où $k \in \{\ell, \dots, m\}$.
 - En calcul exact on cherche un pivot a_{kc} le plus simple possible.
 - En calcul approché on cherche a_{kc} de plus grande valeur absolue.
- 5 Si $k > \ell$ on effectue $L_\ell \leftrightarrow L_k$ échangeant les lignes ℓ et k .
- 6 On divise, $L_\ell \leftarrow a_{\ell c}^{-1} L_\ell$, afin d'obtenir un pivot $a_{\ell c} = 1$.
- 7 On effectue pour toutes les lignes $k = \ell + 1, \dots, m$ l'opération $L_k \leftarrow L_k - a_{kc} L_\ell$ annulant les coefficients en dessous du pivot.
- 8 Optionnellement on annule les coefficients au dessus du pivot.
- 9 On incrémente c et ℓ et on passe à l'étape 2.

L'algorithme de Gauss : complexité du calcul

Proposition

Afin de mettre une matrice $A \in \mathbb{K}^{m \times n}$ sous forme échelonnée, l'algorithme de Gauss effectue au plus mn^2 opérations dans \mathbb{K} .

Démonstration. Comptons le nombre d'opérations à effectuer afin de traiter la c -ième colonne pour $c = 1, \dots, n$. (On supposera $n \geq m$.)

- D'abord, pour mettre le pivot en place :
 $n - c$ échanges puis $n - c$ divisions dans \mathbb{K} .
- Ensuite, pour éliminer les coefficients dans la colonne du pivot :
 $(m - 1)(n - c)$ multiplications et soustractions dans \mathbb{K} .

Au total cela fait $2m(n - c)$ opérations dans \mathbb{K} pour la c -ième colonne. Il faut ensuite sommer sur $c = 1, \dots, n$, ce qui donne le résultat. \square

 On a appliqué ici quelques optimisations : les coefficients des colonnes précédentes sont déjà annulés et ne jouent plus aucun rôle. Ceci économise un facteur $\frac{1}{2}$. (Sinon on effectue $2mn^2$ opérations.)

 On considère ici les opérations dans \mathbb{K} à coût constant, ce qui n'est justifié que si \mathbb{K} est fini, ou si les coefficients restent petits. Sur un corps comme \mathbb{Q} ou $\mathbb{Q}(X)$, même pour une matrice modeste, les calculs provoquent souvent une « explosion des coefficients ».

Avertissement numérique : le bon pivot

Soit à résoudre le système linéaire suivant avec un paramètre $\varepsilon \approx 0$:

$$\begin{cases} \varepsilon x + 1.0y = 1.0 \\ x + 2.0y = 3.0 \end{cases}$$

La solution exacte est $x = \frac{1}{1-2\varepsilon} \approx 1$ et $y = \frac{1-3\varepsilon}{1-2\varepsilon} \approx 1$. (Exercice !)

Supposons d'abord que l'on utilise le pivot 1, comme il se doit :

$$\begin{array}{l} \text{On obtient} \\ \text{puis} \end{array} \begin{cases} x + 2.0y = 3.0 \\ \varepsilon x + 1.0y = 1.0 \end{cases}$$
$$\begin{cases} x + 2.0y = 3.0 \\ (1.0 - 2.0\varepsilon)y = (1.0 - 3.0\varepsilon) \end{cases}$$

 Sur ordinateur on effectue souvent des calculs arrondis !
Par exemple, pour une précision de 52 bits et $|\varepsilon| \leq 2^{-54}$, le calcul arrondi évalue $1.0 - 2.0\varepsilon$ à 1.0 et de même $1.0 - 3.0\varepsilon$ à 1.0.

En mode approché, on calcule $y = 1.0$ puis en remontant $x = 1.0$.
Ce n'est plus exact, mais reste assez proche de la vraie solution.

Avertissement numérique : le mauvais pivot

On considère toujours le même système linéaire où $\varepsilon \approx 0$.

$$\begin{cases} \varepsilon x + 1.0y = 1.0 \\ x + 2.0y = 3.0 \end{cases}$$

Supposons que l'on utilise (maladroitement) comme pivot ε :

On obtient
$$\begin{cases} x + \frac{1.0}{\varepsilon}y = \frac{1.0}{\varepsilon} \\ x + 2.0y = 3.0 \end{cases}$$

puis
$$\begin{cases} x + \frac{1.0}{\varepsilon}y = \frac{1.0}{\varepsilon} \\ (2.0 - \frac{1.0}{\varepsilon})y = (3.0 - \frac{1.0}{\varepsilon}) \end{cases}$$

 Ici le calcul arrondi mène inévitablement à la catastrophe ! Pour une précision de 52 bits et $0 < \varepsilon \leq 2^{-54}$, le calcul arrondi évalue $(2.0 - \frac{1.0}{\varepsilon})$ à $(-\frac{1.0}{\varepsilon})$ et de même $(3.0 - \frac{1.0}{\varepsilon})$ à $(-\frac{1.0}{\varepsilon})$.

On calcule ainsi $y = 1.0$ et en remontant $x = 0.0$. C'est très faux !
Pire encore, l'utilisateur non averti ne s'en rendra pas compte.

Contrairement à un calcul réfléchi, des logiciels numériques ignorent souvent des anomalies. Des vérifications sont indispensables !

Calcul matriciel : notation

Fixons $I = \{1, \dots, m\}$ et $J = \{1, \dots, n\}$ avec $m, n \in \mathbb{N}$.

Une **matrice** de taille $m \times n$ à coefficient dans \mathbb{K} est une famille $A = (a_{ij})$ d'éléments $a_{ij} \in \mathbb{K}$ indexés par $(i, j) \in I \times J$.

Ce n'est rien autre qu'une application $a: I \times J \rightarrow \mathbb{K}$ notée $(i, j) \mapsto a_{ij}$. L'ensemble de ces matrices sera noté $\mathbb{K}^{m \times n}$.

◆ Dans la pratique A s'écrit comme un schéma rectangulaire, avec $i \in I$ indexant les lignes et $j \in J$ indexant les colonnes.

Les matrices $m \times 1$ sont les **vecteurs colonnes**, alors que les matrices $1 \times n$ sont les **vecteurs lignes**.

À chaque matrice $A = (a_{ij})_{ij}$ de taille $m \times n$ on peut associer la matrice transposée $A^t = (a_{ij})_{ji}$ de taille $n \times m$.

◆ Sur ordinateur, on stocke une matrice $m \times n$ comme un **tableau**. Cela marche assez bien pour les matrices de taille modeste. (Pour les grandes matrices creuses il faudra économiser de la mémoire. . .)

Jusqu'ici $\mathbb{K}^{m \times n}$ n'est qu'un ensemble sans structure spécifique. La théorie devient intéressante quand \mathbb{K} est un corps (ou anneau) : dans ce cas on peut définir une addition et une multiplication. . .

Calcul matriciel : opérations

Soit $(\mathbb{K}, +, \cdot)$ un corps. On définit l'addition des matrices $m \times n$ par

$$+ : \mathbb{K}^{m \times n} \times \mathbb{K}^{m \times n} \rightarrow \mathbb{K}^{m \times n}, \quad (A, B) \mapsto C = A + B, \quad c_{ij} = a_{ij} + b_{ij}.$$

La multiplication n'est définie que pour des matrices composables,

$$* : \mathbb{K}^{m \times n} \times \mathbb{K}^{n \times r} \rightarrow \mathbb{K}^{m \times r}, \quad (A, B) \mapsto C = AB, \quad c_{ik} = \sum_{j=1}^n a_{ij} b_{jk}.$$

Ainsi les matrices agissent (à gauche) sur les vecteurs (colonnes) :

$$* : \mathbb{K}^{m \times n} \times \mathbb{K}^n \rightarrow \mathbb{K}^m, \quad (A, x) \mapsto y = Ax, \quad y_i = \sum_{j=1}^n a_{ij} x_j.$$

En plus on a la multiplication scalaire (à gauche)

$$\cdot : \mathbb{K} \times \mathbb{K}^{m \times n} \rightarrow \mathbb{K}^{m \times n}, \quad (\lambda, A) \mapsto B = \lambda A, \quad b_{ij} = \lambda a_{ij}.$$

Proposition

L'ensemble $\mathbb{K}^{n \times n}$ des matrices carrées de taille $n \times n$ sur \mathbb{K} muni de l'addition et de la multiplication définies ci-dessus forme un anneau (non commutatif pour $n \geq 2$, car en général $AB \neq BA$.)

Matrices inversibles

L'élément neutre de $(\mathbb{K}^{n \times n}, *)$ est $\mathbf{1} = 1_{n \times n} = \begin{pmatrix} 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & & 0 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & & 1 & 0 \\ 0 & 0 & \dots & 0 & 1 \end{pmatrix}$.

Définition

On dit qu'une matrice $A \in \mathbb{K}^{n \times n}$ est **inversible** dans $\mathbb{K}^{n \times n}$ s'il existe une matrice $A' \in \mathbb{K}^{n \times n}$ telle que $AA' = A'A = \mathbf{1}$.

Dans ce cas A' est unique. On l'appelle **l'inverse** de A , noté A^{-1} .

Définition

L'ensemble des matrices inversibles dans $\mathbb{K}^{n \times n}$ est appelé le **groupe linéaire** et noté $GL_n(\mathbb{K})$.

Si A et B sont inversibles, alors leur produit AB l'est aussi. Effectivement, $(AB)(B^{-1}A^{-1}) = AB B^{-1} A^{-1} = AA^{-1} = \mathbf{1}$. Ainsi on obtient un produit $*$: $GL_n(\mathbb{K}) \times GL_n(\mathbb{K}) \rightarrow GL_n(\mathbb{K})$.



En général la somme de matrices inversibles n'est pas inversible.

Comment calculer l'inverse d'une matrice ?

L'algorithme de Gauss résout des systèmes d'équations linéaires.

Il calcule aussi efficacement l'inverse d'une matrice $A \in \mathbb{K}^{n \times n}$:

- 1 On accole la matrice identité $1_{n \times n}$ à droite, noté $(A \mid 1_{n \times 1})$.
- 2 On effectue la réduction de A sous forme échelonnée.
(Multiplication par des matrices inversibles élémentaires.)
- 3 Si A est inversible, alors on obtient à droite la matrice inverse.
(Sinon le calcul s'arrête et signale que A n'était pas inversible.)

Pourquoi cet algorithme est-il correct ?

- On commence par $(A_0 \mid B_0) = (A \mid 1_{n \times n})$.
Cette initialisation assure l'égalité $A_0 = B_0 A$.
- Chaque opération élémentaire sur les lignes transforme $(A_k \mid B_k)$ en $(A_{k+1} \mid B_{k+1})$ où $A_{k+1} = T_k A_k$ et $B_{k+1} = T_k B_k$.
On préserve ainsi l'égalité $A_k = B_k A$.
- À la fin on obtient $(A_m \mid B_m)$ où $A_m = \mathbf{1}$.
Ainsi l'égalité $\mathbf{1} = B_m A$ assure que $B_m = A^{-1}$.

Le déterminant

Soit \mathbb{K} un corps. Pour tout $n \in \mathbb{N}$ il existe une et une seule application $\det: \mathbb{K}^{n \times n} \rightarrow \mathbb{K}$, appelée **déterminant**, qui soit alternée, multilinéaire par rapport aux colonnes et unitaire au sens que $\det(1_{n \times n}) = 1_{\mathbb{K}}$.

Elle jouit des propriétés suivantes :

- 1 Le déterminant se développe en la formule polynomiale

$$\det A = \sum_{\sigma} \text{sign}(\sigma) \cdot a_{1,\sigma(1)} \cdot a_{2,\sigma(2)} \cdots a_{n,\sigma(n)}$$

où $\sigma: \{1, \dots, n\} \rightarrow \{1, \dots, n\}$ parcourt toutes les permutations.

- 2 Le déterminant est invariant par transposition : $\det(A^t) = \det(A)$.
Il est donc aussi alterné et multilinéaire par rapport aux lignes.
- 3 Le déterminant est multiplicatif : $\det(AB) = \det(A) \det(B)$.
Par restriction on obtient donc $\det: \text{GL}_n(\mathbb{K}) \rightarrow \mathbb{K}^\times$.
- 4 Une matrice A est inversible dans $\mathbb{K}^{n \times n}$ si et seulement si $\det(A)$ est inversible dans \mathbb{K} . (Ici « \Rightarrow » découle de la multiplicativité, alors que « \Leftarrow » se déduit de la formule de Cramer, qui exprime A^{-1} en fonction de A et de $\det A$.)

Comment calculer le déterminant d'une matrice ?

 Aussi élégante qu'elle soit, la belle formule

$$\det A = \sum_{\sigma} \text{sign}(\sigma) \cdot a_{1,\sigma(1)} \cdot a_{2,\sigma(2)} \cdots a_{n,\sigma(n)}$$

n'est utilisable que pour n très petit. (Calculez $10!$ puis $20!$ puis $50!$)

 De même, le développement par lignes et/ou colonnes n'est profitable que si la matrice présente beaucoup de zéros. Sinon on retombe sur la formule ci-dessus, trop lourde pour n grand.

 À nouveau l'algorithme de Gauss vient à notre secours :
On réduit la matrice $A \in \mathbb{K}^{n \times n}$ à la matrice identité, où $\det \mathbf{1} = 1$.
Lors de l'algorithme on note le changement du déterminant :

- Chaque inversion $L_i \leftrightarrow L_j$ renverse le signe de $\det A$.
- Chaque multiplication $L_i \leftarrow \lambda L_i$ multiplie $\det A$ par λ .
- Chaque addition $L_i \leftarrow L_i + \lambda L_j$ laisse $\det A$ invariant.

Avertissement numérique

Pour toute méthode numérique il est important de comprendre la propagation d'erreurs afin d'éviter un usage inapproprié.

Exemple. On considère l'équation $Ax = y$ avec

$$A = \begin{pmatrix} 0.780 & 0.563 \\ 0.913 & 0.659 \end{pmatrix} \quad \text{et} \quad y = \begin{pmatrix} 0.217 \\ 0.254 \end{pmatrix}.$$

La matrice A est inversible, car on trouve $\det A = 10^{-6}$.

Lequel des deux résultats approchés suivants est meilleur :

$$\tilde{x} = \begin{pmatrix} 0.999 \\ -1.001 \end{pmatrix} \quad \text{ou} \quad \hat{x} = \begin{pmatrix} 0.341 \\ -0.087 \end{pmatrix}?$$

On pourrait naïvement calculer les erreurs $|A\tilde{x} - y|$ et $|A\hat{x} - y|$, puis choisir la solution qui minimise cette erreur. Ici c'est \hat{x} , car

$$A\tilde{x} - y = \begin{pmatrix} -0.001343 \\ -0.001572 \end{pmatrix} \quad \text{et} \quad A\hat{x} - y = \begin{pmatrix} -0.0000001 \\ 0.0000000 \end{pmatrix}.$$

Pourtant la solution exacte est $x = \begin{pmatrix} +1 \\ -1 \end{pmatrix}$... Le vérifier !

À notre grande surprise, c'est donc \tilde{x} qui est le plus proche. Comment expliquer puis quantifier ce phénomène étrange ?

La norme d'une matrice

Dans la plupart de nos exemples nous travaillons dans \mathbb{R}^n ou \mathbb{C}^n .
On munit cet espace de la norme euclidienne $|x| = \sqrt{\sum_{k=1}^n |x_k|^2}$.
On déduit de cette norme de vecteurs une norme de matrices :

Définition

Pour $A \in \mathbb{C}^{n \times n}$ on définit la **norme** par $\|A\| := \sup\{|Ax|; |x| \leq 1\}$.

La norme $\|A\|$ mesure la « distorsion » : on a $|Ax| \leq \|A\| \cdot |x|$ pour tout $x \in \mathbb{C}^n$, et $\|A\|$ est la plus petite valeur assurant cette inégalité.

Exemple : Si $A \sim \text{diag}(\lambda_1, \dots, \lambda_n)$, alors $\|A\| = \max\{|\lambda_1|, \dots, |\lambda_n|\}$.

Proposition

L'application $A \mapsto \|A\|$ définit une norme sur $\mathbb{C}^{n \times n}$:

homogénéité : $\|\lambda A\| = |\lambda| \cdot \|A\|,$

inégalité triangulaire : $\|A + B\| \leq \|A\| + \|B\|,$

positivité : $\|A\| \geq 0,$ **et** $\|A\| = 0 \Leftrightarrow A = 0.$

Le conditionnement d'une matrice

Considérons une équation $Ax = y$ où la matrice A est inversible.

Question de **stabilité numérique** ou de **sensibilité aux erreurs** :

Comment varie la solution x si l'on perturbe le vecteur y ?

Plus explicitement, soit $\tilde{y} = y + \delta y$ et soit \tilde{x} solution de $A\tilde{x} = \tilde{y}$.

Puisque tout est linéaire, on a $\tilde{x} = x + \delta x$ où $A\delta x = \delta y$.

Comment majorer l'erreur relative $\frac{|\delta x|}{|x|}$ en fonction de $\frac{|\delta y|}{|y|}$?

D'une part on a $|\delta y| \leq \|A\| \cdot |\delta x|$ et $|\delta x| \leq \|A^{-1}\| \cdot |\delta y|$, donc

$$\frac{1}{\|A\|} \frac{|\delta y|}{|x|} \leq \frac{|\delta x|}{|x|} \leq \|A^{-1}\| \frac{|\delta y|}{|x|}.$$

D'autre part on a $|y| \leq \|A\| \cdot |x|$ et $|x| \leq \|A^{-1}\| \cdot |y|$, donc

$$\frac{1}{\|A\| \cdot \|A^{-1}\|} \cdot \frac{|\delta y|}{|y|} \leq \frac{|\delta x|}{|x|} \leq \|A\| \cdot \|A^{-1}\| \cdot \frac{|\delta y|}{|y|}.$$

Définition

On appelle $\text{cond}(A) := \|A\| \cdot \|A^{-1}\|$ le **conditionnement** de A .

Le conditionnement mesure la sensibilité aux erreurs

On considère une matrice A inversible et une solution $Ax = y$.

Si $A\tilde{x} = \tilde{y}$, où $\tilde{y} = y + \delta y$ entraîne $\tilde{x} = x + \delta x$, alors

$$\frac{1}{\text{cond}(A)} \frac{|\delta y|}{|y|} \leq \frac{|\delta x|}{|x|} \leq \text{cond}(A) \frac{|\delta y|}{|y|}.$$

Ainsi le conditionnement décrit comment une perturbation de y s'amplifie en une perturbation de x :

- On a toujours $\text{cond}(A) = \|A\| \cdot \|A^{-1}\| \geq \|AA^{-1}\| = 1$.
- Si $\text{cond}(A)$ est proche de 1, alors une petite perturbation de y entraîne une petite perturbation de x .
- Si $\text{cond}(A)$ est grand, une petite perturbation de y peut entraîner une grande perturbation de x .

 Le conditionnement est un problème inhérent à la matrice A . Il est indépendant des erreurs d'arrondi : le phénomène persiste même si les calculs intermédiaires sont effectués de manière exacte.

 Ce problème devient inévitable si l'on calcule en mode approché ou si les données initiales ne sont connues approximativement.

Conditionnement d'une matrice : exemple

Considérons la matrice $A = \begin{pmatrix} 7 & 10 \\ 5 & 7 \end{pmatrix}$ qui a pour inverse $A^{-1} = \begin{pmatrix} -7 & 10 \\ 5 & -7 \end{pmatrix}$.

Pour les valeurs propres de A on trouve $\lambda_1 = 7 - 5\sqrt{2} \approx -0.071$ et $\lambda_2 = 7 + 5\sqrt{2} \approx 14.071$, donc la norme de A est $\|A\| \approx 14.071$.

Les valeurs propres de A^{-1} sont $-7 \pm 5\sqrt{2}$, donc $\|A^{-1}\| \approx 14.071$.

Ainsi le conditionnement de A est $\text{cond}(A) = \|A\| \cdot \|A^{-1}\| \approx 198$.

Ceci indique que $Ax = y$ peut être sensible aux perturbations de y .

Exemple. $Ax = \begin{pmatrix} 1.00 \\ 0.70 \end{pmatrix}$ donne $x = \begin{pmatrix} 0.00 \\ 0.10 \end{pmatrix}$,

alors que $A\tilde{x} = \begin{pmatrix} 1.01 \\ 0.69 \end{pmatrix}$ donne $\tilde{x} = \begin{pmatrix} -0.17 \\ 0.22 \end{pmatrix}$.

On constate que le changement relatif en x est plus grand que le changement relatif en y , mais tout au plus par un facteur $\text{cond}(A)$.

Conclusion

Un conditionnement $\text{cond}(A) \approx 10^c$ indique une possible perte de précision, typiquement de c décimales : En général, la donnée de y avec une précision de ℓ décimales significatives mène à une solution x avec une précision de $\ell - c$ décimales significatives seulement.

Espaces vectoriels et bases

Soit E un espace vectoriel sur un corps \mathbb{K} .

Définition

Une **base** de E est une famille $(u_i)_{i \in I}$ dans E de sorte que tout $v \in E$ s'écrit de manière unique comme $v = \sum_{i \in I} \lambda_i u_i$ où $\lambda_i \in \mathbb{K}$.

Ici la somme $\sum \lambda_i u_i$ est toujours finie. Si jamais I est infini, on exige que seulement un nombre fini des coefficients λ_i soient non nuls.

Théorème

*Tout espace vectoriel E admet des bases. Deux bases $(u_i)_{i \in I}$ et $(v_j)_{j \in J}$ de E ont toujours le même cardinal, c'est-à-dire $I \cong J$. Ceci permet de définir la **dimension** de E comme $\dim E := \text{card}(I)$.*

Exemples :

\mathbb{R}^n avec la base canonique $e_1 = (1, 0, \dots, 0), \dots, e_n = (0, \dots, 0, 1)$.

$\mathbb{R}[X]_{\leq n}$ avec la base formée des monômes $1, X, X^2, X^3, \dots, X^n$.

$\mathbb{R}[X]$ avec la base formée des monômes $1, X, X^2, X^3, \dots$.

$\mathbb{R}^{\mathbb{N}}$, l'espace des suites infinies (a_0, a_1, a_2, \dots) . Base ?

$C([a, b])$, l'espace des fonctions continues $f: [a, b] \rightarrow \mathbb{R}$. Base ?

Le rôle particulier de \mathbb{K}^n

\mathbb{K}^n est le modèle universel d'un \mathbb{K} -espace vectoriel de dimension n .

Soit E un \mathbb{K} -espace vectoriel de dimension finie, tel que $\dim E = n$.
Le choix d'une base revient à fixer un isomorphisme $\phi: \mathbb{K}^n \xrightarrow{\sim} E$.

Plus explicitement, toute base (u_1, \dots, u_n) de E définit un isomorphisme $\phi: \mathbb{K}^n \xrightarrow{\sim} E$ par $\phi(\lambda_1, \dots, \lambda_n) = \lambda_1 u_1 + \dots + \lambda_n u_n$.
L'inverse $\phi^{-1}(v)$ est l'écriture de v dans la base (u_1, \dots, u_n) .

◆ Moyennant un tel isomorphisme $\phi: \mathbb{K}^n \xrightarrow{\sim} E$ et son inverse $\phi^{-1}: E \rightarrow \mathbb{K}^n$ on peut ramener tous les calculs dans des espaces vectoriels de dimension finie à des calculs dans \mathbb{K}^n .

⚠ À noter que ceci ne justifie pas d'écrire $E = \mathbb{K}^n$.
On a $E \cong \mathbb{K}^n$ mais cet isomorphisme n'est pas unique.
En général, l'espace E n'est pas muni d'une base canonique.

Le choix d'une base adaptée au problème

Regardons l'espace $\mathbb{K}[X]_{\leq n}$ des polynômes de degré $\leq n$ sur \mathbb{K} .
Différentes bases seront utiles, suivant le problème en question :

- Pour le développement de Taylor en 0 on travaille dans la base formée des monômes $1, X, X^2, \dots, X^n$.
- Pour le développement de Taylor en x_0 , par contre, on travaille plutôt dans la base $1, (X - x_0), (X - x_0)^2, \dots, (X - x_0)^n$.
- L'interpolation de Lagrange en x_0, x_1, \dots, x_n s'exprime le plus facilement dans la base $P_k = \prod_{j \neq k} \frac{X - x_j}{x_k - x_j}$.
- L'algorithme des différences divisées travaille avec la base $1, (X - x_0), (X - x_0)(X - x_1), \dots, (X - x_0)(X - x_1) \cdots (X - x_{n-1})$.
- Si $\mathbb{K}[X]_{\leq n}$ est muni d'un produit scalaire, on préfère travailler avec une base orthonormée : algorithme de Gram-Schmidt, polynômes de Legendre, de Tchebychev, de Hermite, etc.
- ... la liste peut être prolongée à volonté.

Souvent le problème se résout par le choix judicieux d'une base adaptée. Les algorithmes sous-jacents méritent donc de l'attention.

Applications linéaires et matrices

Soit E un espace vectoriel, et soit (u_1, \dots, u_n) une base de E .

Soit F un espace vectoriel, et soit (v_1, \dots, v_m) une base de F .

Soit G un espace vectoriel, et soit (w_1, \dots, w_r) une base de G .

Correspondance. Toute application linéaire $f: E \rightarrow F$ définit une matrice $A = (a_{ij})_{\substack{i=1, \dots, m \\ j=1, \dots, n}}$ telle que $f(u_j) = \sum_{i=1}^m a_{ij} v_i$ pour tout j . Réciproquement toute matrice définit ainsi une application linéaire.

Cette correspondance est résumée dans le diagramme suivant :

$$\begin{array}{ccc} E & \xrightarrow{f} & F \\ \phi \uparrow \cong & & \psi \uparrow \cong \\ \mathbb{K}^n & \xrightarrow{A} & \mathbb{K}^m \end{array}$$

 La correspondance entre applications linéaires $f: E \rightarrow F$ et matrices $A \in \mathbb{K}^{m \times n}$ nécessite le choix de deux bases, pour E et F ! Tout autre choix marchera, mais la correspondance sera différente.

Le rôle particulier des matrices

Les matrices $A \in \mathbb{K}^{m \times n}$ sont un modèle universel pour des applications linéaires entre \mathbb{K} -espace vectoriels de dimension finie.

Addition. Si $f, g: E \rightarrow F$ sont représentées par $A, B \in \mathbb{K}^{m \times n}$, alors l'application somme $f + g$ est représentée par $A + B$.
De même, λf est représentée par λA .

Multiplication. Si $f: E \rightarrow F$ est représentée par $A \in \mathbb{K}^{m \times n}$ et si $g: F \rightarrow G$ est représentée par $B \in \mathbb{K}^{r \times m}$, alors l'application composée $g \circ f: E \rightarrow G$ est représentée par $BA \in \mathbb{K}^{r \times n}$.

$$\begin{array}{ccccc} E & \xrightarrow{f} & F & \xrightarrow{g} & G \\ \phi \uparrow \cong & & \psi \uparrow \cong & & v \uparrow \cong \\ \mathbb{K}^n & \xrightarrow{A} & \mathbb{K}^m & \xrightarrow{B} & \mathbb{K}^r \end{array}$$

Exercice/révision : Démontrer ces affirmations.

◆ Ces observations nous permettent de ramener tous les calculs sur des applications linéaires au calcul matriciel, déjà discuté. Ceci justifie de ne regarder que des matrices dans la suite.

Vecteurs et valeurs propres, polynôme caractéristique

Définition

Soit $A \in \mathbb{K}^{n \times n}$ une matrice carrée. Un vecteur non nul $v \in \mathbb{K}^n \setminus \{0\}$ est un **vecteur propre** de A si $Av = \lambda v$ où $\lambda \in \mathbb{K}$. Dans ce cas λ est une **valeur propre** de A , et que v est un vecteur propre associé à λ .

Puisque $Av = \lambda v$ équivaut à $(A - \lambda I)v = 0$, on cherche des valeurs λ pour lesquelles la matrice $A - \lambda I$ ait un noyau non trivial.

Définition

On appelle $\chi_A := \det(A - XI)$ le **polynôme caractéristique** de A .

C'est effectivement un polynôme dans $\mathbb{K}[X]$ de degré n . On a $\chi_A(\lambda) = 0$ si et seulement si λ est une valeur propre de A .

◆ Pour trouver les valeurs propres de A on est ainsi ramené à résoudre une équation polynomiale, comme déjà discuté.
Vive la résolution d'équations polynomiales !

◆ Pour trouver l'espace des vecteurs propres associés à λ on est ramené à trouver le noyau de la matrice $A - \lambda I$, comme déjà discuté.
Vive l'algorithme de Gauss !

Comment calculer le polynôme caractéristique ?

Pour une matrice A de petite taille on peut calculer son polynôme caractéristique $\chi_A = \det(A - XI)$ en développant ce déterminant par une méthode quelconque. Ce calcul est effectué sur $\mathbb{K}[X]$.

◆ Pour les matrices plus grandes il est beaucoup plus efficace de calculer le polynôme caractéristique par interpolation !

Pour $A \in \mathbb{K}^{n \times n}$ on sait que $\chi_A \in \mathbb{K}[X]$ est de degré n .

Il suffit donc de connaître sa valeur en $n + 1$ points distincts.

On calcule donc $n + 1$ déterminants $\det(A - \lambda I)$ en remplaçant λ par $n + 1$ valeurs choisies arbitrairement, par exemple $\lambda = 0, 1, 2, \dots, n$.

Puis on reconstruit le polynôme caractéristique χ_A par interpolation de Lagrange. (Revoir l'algorithme des différences divisées.)

Avantage : Bien qu'il y ait plus de déterminants à calculer, ce sont des déterminants sur \mathbb{K} : la variable X n'y figure plus. Sur un corps \mathbb{K} le déterminant se calcule efficacement par l'algorithme de Gauss, comme déjà discuté.

Le polynôme minimal d'une matrice carrée

Soit $A \in \mathbb{K}^{n \times n}$ une matrice carrée sur un corps \mathbb{K} .

Pour tout polynôme $P = a_0 + a_1X + \cdots + a_kX^k$ dans $\mathbb{K}[X]$ on définit la matrice $P(A) := a_0I + a_1A + \cdots + a_kA^k$ dans $\mathbb{K}^{n \times n}$.

Puisque $\dim \mathbb{K}^{n \times n} = n^2$, les matrices $1, A, A^2, \dots, A^{n^2}$ sont linéairement liées. Il existe donc un polynôme P tel que $P(A) = 0$. On dit que le polynôme P **annule** la matrice A si $P(A) = 0$.

Définition

Le **polynôme minimal** d'une matrice carrée $A \in \mathbb{K}^{n \times n}$ est l'unique polynôme unitaire $\mu \in \mathbb{K}[X]$ de degré minimal tel que $\mu(A) = 0$.

Remarque

Le polynôme minimal divise tout polynôme annulateur P de A :
Par division euclidienne on a $P = \mu \cdot Q + R$ où $\deg R < \deg \mu$.
Puisque P annule la matrice A , on obtient ainsi $0 = P(A) = R(A)$.
Si $\deg R \geq 1$ alors μ ne serait pas minimal. On conclut que $R = 0$.

Polynôme minimal : un premier exemple

Nous cherchons le polynôme minimal de la matrice $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$.

D'abord on pourra contempler quelques puissances de A :

$$I = A^0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad A^1 = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, \quad A^2 = \begin{pmatrix} 7 & 10 \\ 15 & 22 \end{pmatrix}, \quad \dots$$

Degré 1 : existe-t-il un polynôme annulateur $P = a_0 + X$?

Ceci voudrait dire que $P(A) = a_0A^0 + A^1 = 0$. C'est impossible !

Degré 2 : existe-t-il un polynôme annulateur $P = a_0 + a_1X + X^2$?

Ceci voudrait dire que $P(A) = a_0A^0 + a_1A^1 + A^2 = 0$. Calculons !

$$P(A) = \begin{pmatrix} a_0 + a_1 + 7 & 2a_1 + 10 \\ 3a_1 + 15 & a_0 + 4a_1 + 22 \end{pmatrix}$$

On voit que le polynôme $P = X^2 - 5X - 2$ annule la matrice A .

C'est donc le polynôme minimal de A , car degré < 2 est impossible.

Remarquons finalement qu'ici le polynôme minimal μ_A coïncide avec le polynôme caractéristique $\chi_A = \det(A - XI) = X^2 - 5X - 2$.

Polynômes minimal : le théorème de Cayley-Hamilton

Théorème (de Cayley-Hamilton, admis)

*Le polynôme caractéristique χ_A annule la matrice A .
Autrement dit, le polynôme minimal μ_A divise χ_A .*



En particulier on en déduit que $\deg \mu_A \leq \deg \chi_A = n$.

Exemple : Pour $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$ on trouve le polynôme caractéristique

$$\chi_A = -X^3 + 15X^2 + 18X.$$

Les puissances de A sont

$$A^1 = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}, \quad A^2 = \begin{pmatrix} 30 & 36 & 42 \\ 66 & 81 & 96 \\ 102 & 126 & 150 \end{pmatrix}, \quad A^3 = \begin{pmatrix} 468 & 576 & 684 \\ 1062 & 1305 & 1548 \\ 1656 & 2034 & 2412 \end{pmatrix}.$$

On constate (après calcul) qu'effectivement

$$A^3 - 15A^2 - 18A = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Polynômes minimal et caractéristique : exemples

Exemple où $A \in \mathbb{C}^{3 \times 3}$ a trois valeurs propres distinctes :

$$A = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \quad \Rightarrow \quad \chi_A = \mu_A = (X - a)(X - b)(X - c)$$

Exemples où $A \in \mathbb{C}^{3 \times 3}$ n'a que deux valeurs propres distinctes :

$$A = \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & b \end{pmatrix} \quad \Rightarrow \quad \chi_A = (X - a)^2(X - b)$$

$$\mu_A = (X - a)(X - b)$$

$$A = \begin{pmatrix} a & 1 & 0 \\ 0 & a & 0 \\ 0 & 0 & b \end{pmatrix} \quad \Rightarrow \quad \chi_A = \mu_A = (X - a)^2(X - b)$$

Exemples où $A \in \mathbb{C}^{3 \times 3}$ n'a qu'une seule valeur propre :

$$A = \begin{pmatrix} a & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \quad \Rightarrow \quad \chi_A = (X - a)^3$$

$$\mu_A = (X - a)$$

$$A = \begin{pmatrix} a & 1 & 0 \\ 0 & a & 0 \\ 0 & 0 & a \end{pmatrix} \quad \Rightarrow \quad \chi_A = (X - a)^3$$

$$\mu_A = (X - a)^2$$

$$A = \begin{pmatrix} a & 1 & 0 \\ 0 & a & 1 \\ 0 & 0 & a \end{pmatrix} \quad \Rightarrow \quad \chi_A = \mu_A = (X - a)^3$$

Par le théorème de Jordan ces exemples épuisent déjà toutes les possibilités à équivalence dans $\mathbb{C}^{3 \times 3}$ près.

Polynômes minimal et caractéristique : racines

Corollaire

*Le polynôme minimal μ_A divise le polynôme caractéristique χ_A .
Il a les mêmes racines, éventuellement de multiplicité réduite.*

Démonstration. On a $\mu_A \mid \chi_A$, c'est-à-dire $\chi_A = \mu_A \cdot Q$.
Ainsi toute racine de μ_A est aussi une racine de χ_A .

Réciproquement, supposons que $\chi_A(\lambda) = 0$.
C'est le cas si et seulement si $\ker(A - \lambda I) \neq \{0\}$.

Par division euclidienne on a $\mu_A = P \cdot (X - \lambda) + r$ où $r \in \mathbb{K}$, donc

$$0 = \mu_A(A) = P(A) \cdot (A - \lambda I) + rI.$$

Appliqué à un vecteur non nul $v \in \ker(A - \lambda I)$ ceci donne $0 = rv$.
On conclut que $r = 0$, donc λ est une racine de μ_A . □

Polynômes minimal et caractéristique : racines simples

Supposons que le polynôme caractéristique factorise comme

$$\chi_A = (X - \lambda_1)^{n_1} \cdots (X - \lambda_k)^{n_k}$$

avec des racines distinctes $\lambda_1, \dots, \lambda_k$ de multiplicité $n_1, \dots, n_k \geq 1$.

Alors le polynôme minimal factorise comme

$$\mu_A = (X - \lambda_1)^{m_1} \cdots (X - \lambda_k)^{m_k}$$

avec les mêmes racines $\lambda_1, \dots, \lambda_k$ et de multiplicités $1 \leq m_i \leq n_i$.

Corollaire

Si χ_A n'a que des racines simples, alors $\mu_A = \chi_A$. □

Si l'on choisit $A \in \mathbb{C}^{n \times n}$ de manière aléatoire, alors avec probabilité 1 le polynôme caractéristique χ_A aura des racines simples et $\mu_A = \chi_A$.

Néanmoins, le phénomène $\deg \mu_A < \deg \chi_A$ arrive dans de nombreuses applications (où A n'est pas du tout aléatoire).

Comment calculer le polynôme minimal ?

On considère les matrices $I = A^0, A = A^1, A^2, \dots, A^n$ comme vecteurs dans $\mathbb{K}^{(n^2)}$. On peut les écrire comme vecteurs lignes : Ceci donne une matrice M à $n + 1$ lignes et n^2 colonnes.

Si les k premières lignes de M sont linéairement dépendantes, on obtient alors on relation non triviale

$$a_0 I + a_1 A + a_2 A^2 + \dots + a_k A^k = 0.$$

Ceci se détermine en appliquant l'algorithme de Gauss.

Toute telle relation donne un polynôme annulateur

$$P = a_0 + a_1 X + a_2 X^2 + \dots + a_k X^k.$$

Si k est minimal, alors $a_k \neq 0$ et on obtient $\mu_A = P/a_k$.



Cette méthode marche bien quand la dimension n est petite.



Cette méthode est souvent trop coûteuse pour n grand, car il faudra réduire une matrice M ayant n^2 lignes et $n + 1$ colonnes.

Un algorithme probabiliste plus rapide

Pour calculer μ_A il existe une méthode probabiliste moins coûteuse.

On calcule le polynôme minimal μ_A^v par rapport à un vecteur $v \neq 0$:

- On calcule les vecteurs $v_0 = v, v_1 = Av, \dots, v_n = A^n v$.
- Dans \mathbb{K}^n on cherche une relation non triviale

$$a_0 v_0 + a_1 v_1 + \dots + a_k v_k = 0$$

avec k minimal. On peut ensuite se ramener à $a_k = 1$.

- Ceci définit un polynôme $\mu_A^v = a_0 + a_1 X + \dots + a_k X^k$.
Il vérifie $\mu_A^v(A)v = 0$, mais pas forcément $\mu_A^v(A) = 0$.

Avantage : les calculs sont beaucoup moins lourds !

La construction garantit que μ_A^v divise μ_A , qui lui divise χ_A .

On effectue ce calcul pour un vecteur v choisi au hasard.

Si, par chance, μ_A^v est de degré n , alors $\mu_A^v = \mu_A = \chi_A$.

Sinon, on peut essayer quelques autres vecteurs,
puis calculer le ppcm des polynômes ainsi obtenus.

Si l'on obtient un polynôme de degré n on conclut comme avant.

Sinon on peut tester si ce polynôme annule la matrice A .

Dans ce cas favorable on a trouvé le polynôme minimal μ_A .

Diagonalisation de matrices : critères

On considère une matrice $A \in \mathbb{K}^{n \times n}$,
typiquement à coefficients dans $\mathbb{K} = \mathbb{R}$ ou $\mathbb{K} = \mathbb{C}$.

A est **diagonalisable**

⇔^{Déf} Il existe une base $v_1, \dots, v_n \in \mathbb{K}^n$ formée de vecteurs propres, c'est-à-dire $Av_1 = \lambda_1 v_1, \dots, Av_n = \lambda_n v_n$.

⇔ Dans la base (v_1, \dots, v_n) l'endomorphisme $x \mapsto Ax$ s'écrit comme une matrice diagonale, $TAT^{-1} = \text{diag}(\lambda_1, \dots, \lambda_n)$. Ici T est la matrice de passage de (e_1, \dots, e_n) à (v_1, \dots, v_n) .

⇔ Le polynôme minimal de A n'a que des racines simples, c'est-à-dire que $\text{pgcd}(\mu_A, \mu'_A) = 1$.

⇐ Le polynôme caractéristique de A n'a que des racines simples, c'est-à-dire que $\text{pgcd}(\chi_A, \chi'_A) = 1$.

 Il peut être difficile de calculer les polynômes μ_A ou χ_A .
À ce propos revoir les algorithmes discutés plus haut.

 Il peut être difficile de trouver les racines de μ_A ou de χ_A .

 Le critère $\text{pgcd}(P, P')$ est beaucoup plus facile à appliquer : pour savoir si les racines sont simples on n'a pas besoin de les calculer !

La méthode de la puissance : idée

Considérons une matrice $A \in \mathbb{C}^{n \times n}$ à coefficients complexes.

Alors A admet n valeurs propres $\lambda_1, \lambda_2, \dots, \lambda_n \in \mathbb{C}$.

Supposons que $|\lambda_1| > |\lambda_2| \geq \dots \geq |\lambda_n|$.

Objectif : Approcher un vecteur propre $v_1 \in \mathbb{C}^n$ associé à λ_1 .

Heuristique : Supposons que $A = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$.

Pour tout $x = (x_1, x_2, \dots, x_n) \in \mathbb{C}^n$ on trouve

$$\begin{aligned}Ax &= (\lambda_1 x_1, \lambda_2 x_2, \dots, \lambda_n x_n), \\A^k x &= (\lambda_1^k x_1, \lambda_2^k x_2, \dots, \lambda_n^k x_n).\end{aligned}$$

La première coordonnée croît le plus vite ! (On suppose $x_1 \neq 0$.)

Méthode : On commence par un vecteur aléatoire $u_0 \in \mathbb{C}^n$ non nul.

De manière itérative on calcule $u_{k+1} = \frac{Au_k}{|Au_k|}$ pour $k = 0, 1, 2, \dots$

On s'arrête si $Au_k \approx \lambda u_k$ en approximation suffisante pour un $\lambda \in \mathbb{C}$.

Dans ce cas $u_k \approx v_1$ et $\lambda \approx \lambda_1$ répondent au problème.

Vitesse de convergence : L'erreur diminue comme k^n où $k = \frac{|\lambda_2|}{|\lambda_1|}$.

Plus λ_1 dépasse les autres valeurs $\lambda_2, \dots, \lambda_n$, mieux c'est !

La méthode de la puissance : algorithme

La normalisation $u_{k+1} = \frac{Au_k}{|Au_k|}$ assure que $|u_{k+1}| = 1$.

Il reste encore un facteur $\frac{\lambda_1}{|\lambda_1|}$, un peu embêtant.

L'algorithme suivant normalise aussi ce facteur.

On garantit ainsi que $u_k \rightarrow v_1$ tel que $Av_1 = \lambda_1 v_1$.

Algorithme 1 méthode de la puissance

Entrée: une matrice $A \in \mathbb{C}^{n \times n}$ et une précision $\varepsilon > 0$

Sortie: un vecteur $u \in \mathbb{C}^n$ et une valeur $\lambda \in \mathbb{C}$ tels que $Au \approx \lambda u$

On choisit un vecteur $u \in \mathbb{C}^n$ au hasard.

répéter

$$u' \leftarrow Au, \quad u \leftarrow u'$$

Soit $k \in \{1, \dots, n\}$ l'indice minimal tel que $|u_k| = \max\{|u_1|, \dots, |u_n|\}$.

$$u \leftarrow u/u_k \quad // \text{ Ceci assure } |u| = 1 \text{ et } u_k = 1.$$

jusqu'à $|u - u'| \leq \varepsilon$

$$u' \leftarrow Au, \quad \lambda \leftarrow u'_k$$

retourner (u, λ)

La méthode des itérations inverses

Supposons que A admet des valeurs propres $\lambda_1, \lambda_2, \dots, \lambda_n \in \mathbb{C}$.
La méthode de la puissance permet d'approcher un vecteur propre associé à la valeur propre de module maximal (supposée unique).

Objectif : Approcher un vecteur propre $v_k \in \mathbb{C}^n$ associé à λ_k .
Ici on suppose $|\lambda_1| \geq \dots \geq |\lambda_{k-1}| > |\lambda_k| > |\lambda_{k-1}| \geq \dots \geq |\lambda_n|$.

Méthode : D'abord on approche λ_k par une valeur $\lambda \in \mathbb{C}$.
(Rappel : localisation puis approximation d'une racine de χ_A .)

$A - \lambda I$ a des valeurs propres $\lambda_1 - \lambda, \dots, \lambda_k - \lambda, \dots, \lambda_n - \lambda$.

En supposant $\lambda \approx \lambda_k$ mais $\lambda \neq \lambda_k$, cette matrice est inversible :

$B = (A - \lambda I)^{-1}$ a des valeurs propres $\frac{1}{\lambda_1 - \lambda}, \dots, \frac{1}{\lambda_k - \lambda}, \dots, \frac{1}{\lambda_n - \lambda}$.

Les vecteurs propres sont les mêmes pour A puis $A - \lambda I$ puis B .

Avantage : Ici la valeur propre de module maximal est $\frac{1}{\lambda_k - \lambda}$.
On applique alors la méthode de la puissance à la matrice B .

Inconvénient : Pour calculer B il faut d'abord inverser $A - \lambda I$.

Matrices hermitiennes et symétriques

Supposons que $A \in \mathbb{C}^{n \times n}$ est **hermitienne**, $A^t = \bar{A}$.

Cas particulier : $A \in \mathbb{R}^{n \times n}$ est **symétrique**, $A^t = A$.

Sur \mathbb{C}^n on utilise le produit scalaire usuel $\langle u | v \rangle = \bar{u}^t v$.

La matrice A est hermitienne/symétrique si et seulement si

$$\langle u | Av \rangle = \langle Au | v \rangle \quad \text{pour tout } u, v \in \mathbb{C}^n.$$

Toutes les valeurs propres de A sont réelles : si $Av = \lambda v$, alors

$$\lambda \langle v | v \rangle = \langle v | \lambda v \rangle = \langle v | Av \rangle = \langle Av | v \rangle = \langle \lambda v | v \rangle = \bar{\lambda} \langle v | v \rangle$$

Puisque $v \neq 0$ on a $\langle v | v \rangle \neq 0$. On conclut que $\lambda = \bar{\lambda}$.

Les espaces propres de A sont orthogonaux : si $Au = \lambda' u$, alors

$$\lambda \langle u | v \rangle = \langle u | \lambda v \rangle = \langle u | Av \rangle = \langle Au | v \rangle = \langle \lambda' u | v \rangle = \lambda' \langle u | v \rangle$$

Ainsi $(\lambda - \lambda') \langle u | v \rangle = 0$. Pour $\lambda \neq \lambda'$ on conclut que $\langle u | v \rangle = 0$.

Élimination des valeurs propres

Soit A une matrice hermitienne/symétrique à valeurs propres $\lambda_1, \lambda_2, \dots, \lambda_n \in \mathbb{R}$. On suppose que $|\lambda_1| > |\lambda_2| > \dots > |\lambda_n| > 0$.

Pour la valeur propre λ_1 nous savons approcher un vecteur propre v_1 , tel que $Av_1 = \lambda_1 v_1$, par la méthode de la puissance. Quitte à passer à $\frac{v_1}{|v_1|}$ nous pouvons supposer v_1 normé, $|v_1| = 1$.

La matrice $B = A - \lambda_1 v_1 \bar{v}_1^t$ possède les mêmes valeurs et vecteurs propres que A , sauf (λ_1, v_1) qui est remplacé par $(0, v_1)$. En effet,

$$Bv_1 = Av_1 - \lambda_1 v_1 \bar{v}_1^t v_1 = \lambda_1 v_1 - \lambda_1 v_1 = 0$$

Puisque les espaces propres de A sont orthogonaux, on trouve

$$Bv_k = Av_k - \lambda_1 v_1 \bar{v}_1^t v_k = Av_k = \lambda_k v_k$$

À partir de la matrice B on peut donc approcher v_2 par la méthode de la puissance, puis éliminer (λ_2, v_2) , et ainsi de suite.

Avantage : La matrice B est facile à calculer ; on évite l'inversion.

Que fait un moteur de recherche ?

 Pour une présentation détaillée voir ma page web www-fourier.ujf-grenoble.fr/~eiser/enseignement#google ou bien taper « Comment fonctionne Google ? » sur Google.

Que fait un moteur de recherche ?

- Fouille de données : pour un ou plusieurs mots-clés donnés on cherche des pages web associées, si possible pertinentes.
- Classement des résultats : puisque les pages trouvées sont (souvent trop) nombreuses, il faut les trier par importance.

Derrières les coulisses, plusieurs problèmes se présentent :

- La quantité des données à gérer est énorme.
- Toute requête doit être traitée en temps réel.

Solutions : réseau de quelques milliers de PC, stockage des données bien préparées, algorithmes de recherche hautement optimisés, etc.

Finalement, il y a un problème encore plus délicat :

- Comment trier les pages par ordre d'importance ?

Il est hors de question de les classer manuellement, par des êtres humains : ce serait trop coûteux, trop lent, et donc jamais à jour.

L'importance d'une page doit être déterminée de manière automatisée, par un algorithme. Comment est-ce possible ?

Le succès de Google

La grande innovation apportée par Google en 1998 est qu'il trie intelligemment des pages par ordre d'importance. Son étonnante efficacité a fait le succès de Google . . . et la fortune de ses fondateurs, Sergey Brin et Lawrence Page.

 D. Vise, M. Malseed : *Google story*, Dunod, Paris, 2006.

Voici quelques chiffres pour se faire une idée de l'ordre de grandeur :

Cerveaux : environ 19 000 employés (mars 2008)

Ordinateurs : plus de 60 000 PC en réseau sous Linux (2006)

Mémoire vive : plus de 130 000 Go de RAM pour les calculs (2006)

Disques durs : plus de 5 000 000 Go pour le stockage (2006)

Trafic en ligne : quelques milliers de requêtes par secondes (2006)

Il va sans dire que c'est devenu un marché gigantesque :

Cotation boursière : \$140 000 000 000 (avril 2008)

Chiffre d'affaires : \$16 593 000 000 en 2007

Bénéfices net : \$4 203 000 000 en 2007

Part du marché : plus de 50% dans de nombreux pays

Le web est un graphe

Dans une première approximation nous allons négliger le contenu des pages et ne tenir compte que de la structure du graphe.

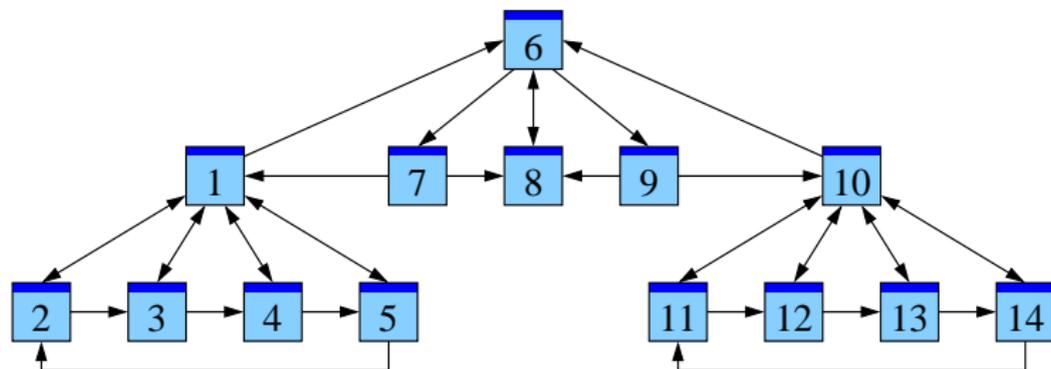


FIG.: Le web vu comme un graphe

Dans la suite on numérote les pages par $1, 2, 3, \dots, n$.

On écrit $j \rightarrow i$ si la page j pointe vers la page i . Exemple :

$1 \rightarrow 2, 3, 4, 5, 6$; $2 \rightarrow 1, 3$; $3 \rightarrow 1, 4$; $4 \rightarrow 1, 5$; $5 \rightarrow 1, 2$;

$6 \rightarrow 7, 8, 9$; $7 \rightarrow 8, 1$; $8 \rightarrow 6$; $9 \rightarrow 8, 10$; $10 \rightarrow 6, 11, 12, 13, 14$;

$11 \rightarrow 10, 12$; $12 \rightarrow 10, 13$; $13 \rightarrow 10, 14$; $14 \rightarrow 10, 11$.

Comment mesurer l'importance d'une page web ?

Première idée : comptage des liens.

Il est plausible qu'une page importante reçoit beaucoup de liens.

Avec un peu de naïveté, on croira aussi l'affirmation réciproque : si une page reçoit beaucoup de liens, alors elle est importante.

Ainsi on pourrait définir l'importance μ_i de la page i comme suit :

$$(1) \quad \mu_i = \sum_{j \rightarrow i} 1.$$

Exemple : Les pages 1 et 10 reçoivent 5 liens chacune, alors que la page 6 n'en reçoit que 3. Ainsi $\mu_1 = \mu_{10} = 5$ mais seulement $\mu_6 = 3$.

Inconvénient : La mesure μ ainsi définie ne correspond pas à l'importance ressentie : elle sous-estime l'importance de la page 6.

Manipulation : On peut artificiellement augmenter l'importance d'une page i en créant des pages « vides de sens » pointant vers i .

Comment mesurer l'importance d'une page web ?

Seconde idée : comptage pondéré.

Certaines pages j émettent un grand nombre ℓ_j de liens : ceux-ci sont donc moins spécifiques et leur « poids » est plus faible. Ainsi on pourrait définir une mesure d'importance plus fine par

$$(2) \quad \mu_i = \sum_{j \rightarrow i} \frac{1}{\ell_j}.$$

Exemple : Dans notre exemple, on trouve $\mu_1 = \mu_{10} = 2.5$ et $\mu_6 = 1.4$.

Inconvénient : La mesure μ ainsi définie ne correspond toujours pas bien à l'importance ressentie par les utilisateurs.

Manipulation : Comme avant, on peut artificiellement augmenter l'importance d'une page i en créant une foule de pages « vides » pointant vers i . De nouveau, la mesure n'est pas fiable.

Comment mesurer l'importance d'une page web ?

Troisième idée : définition récursive.

Le principe utilisé par Google : *une page i est importante si beaucoup de pages importantes pointent vers i .*

Ainsi on pose

$$(3) \quad \mu_i = \sum_{j \rightarrow i} \frac{1}{\ell_j} \mu_j.$$

Cette somme compte chaque lien reçu par i avec poids $\frac{1}{\ell_j} \mu_j$: ceci tient compte de l'importance μ_j de la page d'origine j , et du nombre ℓ_j des liens qui en sont émis.

Exemple : Dans notre exemple, on trouve $\mu_6 = 6$ et $\mu_1 = \mu_{10} = 5$ puis $\mu_8 = 4$. Les autres pages n'obtiennent que $\mu_i = 2$.

Plausibilité : Les pages 6, 1, 10, 8 sont effectivement repérées comme les plus importantes.

Robustesse : Si l'on ajoute des pages « vides de sens » elles recevront l'importance 0 et ne contribueront pas au calcul.

Interprétation en termes d'algèbre linéaire

Notre formule n'est rien autre qu'un système d'équations linéaires :

$$(4) \quad \mu_i = \sum_{j \rightarrow i} \frac{1}{\ell_j} \mu_j.$$

Plus explicitement, pour $i, j \in \{1, \dots, n\}$, on pose

$$(5) \quad a_{ij} := \begin{cases} \frac{1}{\ell_j} & \text{si } j \rightarrow i, \\ 0 & \text{sinon.} \end{cases}$$

Avec cette matrice $A = (a_{ij})$ notre équation (4) s'écrit comme

$$(6) \quad \mu = A\mu$$

C'est un système d'équations linéaires, que l'on peut résoudre par des méthodes adéquates. Remarquons qu'il se réécrit comme

$$(7) \quad (A - I)\mu = 0.$$

Nous cherchons un vecteur propre $\mu \in \mathbb{R}^n$ pour la valeur propre 1.

Interprétation comme marche aléatoire

Avant de calculer aveuglement, interprétons notre équation $A\mu = \mu$.

Par définition $A = (a_{ij})$ est une **matrice stochastique** :

$$\begin{aligned} a_{ij} &\geq 0 && \text{pour tout } i, j \text{ et} \\ \sum_i a_{ij} &= 1 && \text{pour tout } j. \end{aligned}$$

On interprète a_{ij} comme la probabilité d'aller de la page j à la page i en suivant un des ℓ_j liens, choisi au hasard.

Supposons que $x \in \mathbb{R}^n$ est un **vecteur stochastique** :

$$x_j \geq 0 \quad \text{pour tout } j \text{ et } \quad \sum_j x_j = 1,$$

On interprète x_j comme la probabilité de se trouver sur la page j .

Partant de la page j , on suit le lien $j \rightarrow i$ avec probabilité a_{ij} .

Ce chemin nous fait tomber sur la page i avec une probabilité $a_{ij}x_j$.

La probabilité d'arriver sur la page i , par n'importe quel chemin, est

$$y_i = \sum_j a_{ij}x_j.$$

Un pas dans la marche aléatoire correspond à l'application linéaire

$$T: \mathbb{R}^n \rightarrow \mathbb{R}^n, \quad x \mapsto y = Ax.$$

Mesure invariante

Une mesure de probabilité μ vérifiant $\mu = T(\mu)$ est appelée une **mesure invariante** ou une **mesure d'équilibre**.

En termes d'algèbre linéaire c'est un **vecteur propre** pour $\lambda = 1$.

En termes d'analyse, c'est un **point fixe** de l'application T .

Exemple : itérons la marche aléatoire avec une probabilité initiale u_0 :

temps	page 1	page 2	page 3	page 4	page 5	page 6	page 7	page 8	page 9	page 10	page 11	page 12	page 13	page 14
$t=0$	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000
$t=1$	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
$t=2$	0.000	0.000	0.000	0.000	0.000	0.000	0.333	0.333	0.333	0.000	0.000	0.000	0.000	0.000
$t=3$	0.167	0.000	0.000	0.000	0.000	0.333	0.000	0.333	0.000	0.167	0.000	0.000	0.000	0.000
$t=4$	0.000	0.033	0.033	0.033	0.033	0.400	0.111	0.111	0.111	0.000	0.033	0.033	0.033	0.033
$t=5$	0.122	0.017	0.017	0.017	0.017	0.111	0.133	0.244	0.133	0.122	0.017	0.017	0.017	0.017
$t=6$	0.100	0.033	0.033	0.033	0.033	0.293	0.037	0.170	0.037	0.100	0.033	0.033	0.033	0.033
$t=7$	0.084	0.036	0.036	0.036	0.036	0.210	0.098	0.135	0.098	0.084	0.036	0.036	0.036	0.036
$t=8$	0.122	0.035	0.035	0.035	0.035	0.168	0.070	0.168	0.070	0.122	0.035	0.035	0.035	0.035
$t=9$	0.105	0.042	0.042	0.042	0.042	0.217	0.056	0.126	0.056	0.105	0.042	0.042	0.042	0.042
...														
$t=28$	0.125	0.050	0.050	0.050	0.050	0.151	0.050	0.100	0.050	0.125	0.050	0.050	0.050	0.050
$t=29$	0.125	0.050	0.050	0.050	0.050	0.150	0.050	0.100	0.050	0.125	0.050	0.050	0.050	0.050
$t=30$	0.125	0.050	0.050	0.050	0.050	0.150	0.050	0.100	0.050	0.125	0.050	0.050	0.050	0.050

On observe un phénomène de diffusion, plausible après réflexion.

Convergence : Après 30 itérations, on est très proche (à 10^{-3} près) de la solution μ déjà exhibée. Il ne s'agit pas de l'équiprobabilité : certaines pages sont plus fréquentées que d'autres !

Raffinement du modèle

Notre modèle a encore un défaut, illustré par l'exemple suivant :

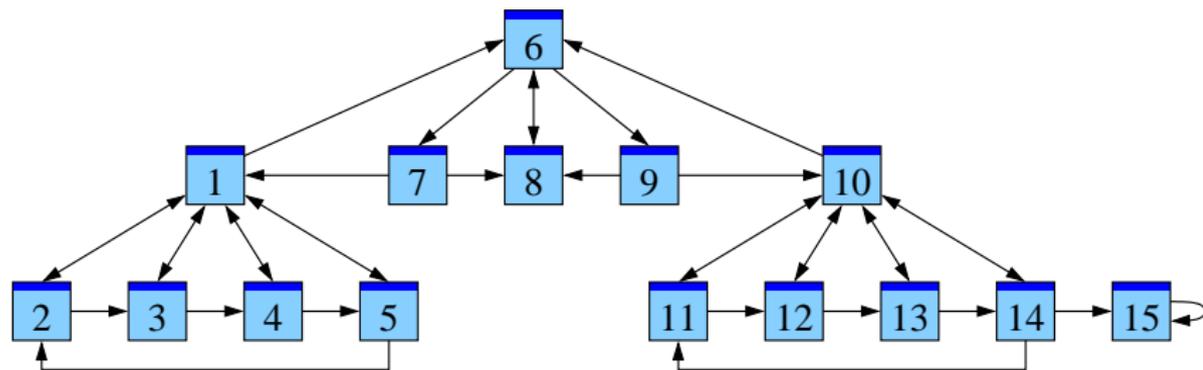


FIG.: Une variante du graphe initial

Google utilise un modèle raffiné dépendant d'un paramètre $c \in [0, 1]$:

- Avec probabilité c , le surfeur abandonne la page actuelle et recommence sur une des n pages du web, choisie au hasard.
- Avec probabilité $1 - c$, le surfeur suit un des liens de la page actuelle j , choisi de manière équiprobable, comme discuté avant.

Cette astuce évite de se faire piéger par une page sans issue.

Elle garantit d'arriver n'importe où dans le graphe, connexe ou non.

Le modèle PageRank utilisé par Google

Le modèle raffiné se formalise comme l'application affine

$$T: \mathbb{R}^n \rightarrow \mathbb{R}^n, \quad x \mapsto c\varepsilon + (1 - c)Ax.$$

Ici A est la matrice stochastique définie ci-dessus.

Le vecteur $\varepsilon = (\frac{1}{n}, \dots, \frac{1}{n})$ correspond à l'équiprobabilité.

La constante $c \in [0, 1]$ est un paramètre du modèle.

Interprétation du paramètre : La valeur $\frac{1}{c}$ est le **nombre moyen** de pages visitées (= liens suivis plus 1) avant de recommencer sur une page aléatoire. En général, on choisira la constante c positive mais proche de zéro. Par exemple, $c = 0.15$ correspond à suivre environ 6 liens en moyenne. (On pourrait argumenter que ceci correspond empiriquement au comportement des utilisateurs... à débattre.)

Exercice : Si vous vous y connaissez en probabilité, prouvez la remarque précédente.

Existence, unicité, et calcul de la solution

Sur \mathbb{R}^n on considère la norme $|x| = \sum_{k=1}^n |x_k|$.

Théorème

Soit $A \in \mathbb{R}^{n \times n}$ une matrice stochastique, soit $c \in]0, 1]$ une constante, et soit $T: \mathbb{R}^n \rightarrow \mathbb{R}^n$ l'application affine définie par

$$T(x) = c\varepsilon + (1 - c)Ax$$

Alors T est contractante de rapport $k = 1 - c < 1$.

Elle admet une unique mesure invariante $\mu = T(\mu)$ et pour tout vecteur initial u_0 la suite itérative $u_{m+1} = T(u_m)$ converge vers μ .

Méthode de la puissance : Partant d'un vecteur stochastique initial, on itère l'application T . La proposition garantit la convergence.

Vitesse. On a $|u_m - \mu| \leq k^m |u_0 - \mu|$, la convergence vers μ est donc au moins aussi rapide que celle de la suite géométrique k^m vers 0.

Démonstration du théorème

Si $x \in \mathbb{R}^n$ est stochastique, alors son image $z = Tx$ l'est aussi : toute coordonnée $z_i = \frac{c}{n} + (1 - c) \sum_j a_{ij} x_j$ vérifie $z_i \geq 0$ puis

$$\begin{aligned} |z| &= \sum_i |z_i| = c + (1 - c) \sum_i \sum_j a_{ij} x_j = c + (1 - c) \sum_j \sum_i a_{ij} x_j \\ &= c + (1 - c) \sum_j \left(\sum_i a_{ij} \right) x_j = c + (1 - c) \sum_j x_j = c + (1 - c) = 1. \end{aligned}$$

Regardons deux vecteurs $x, y \in \mathbb{R}^n$ et essayons de majorer la différence $z := Tx - Ty$ en fonction de $|x - y|$. On a

$$z = [c\varepsilon + (1 - c)Ax] - [c\varepsilon + (1 - c)Ay] = kA(x - y)$$

donc $z_i = k \sum_j a_{ij} (x_j - y_j)$ pour tout $i = 1, \dots, n$. Ainsi

$$\begin{aligned} |Tx - Ty| &= |z| = \sum_i |z_i| = \sum_i \left| k \sum_j a_{ij} (x_j - y_j) \right| \\ &\leq k \sum_i \sum_j |a_{ij} (x_j - y_j)| = k \sum_j \sum_i a_{ij} |x_j - y_j| \\ &= k \sum_j \left(\sum_i a_{ij} \right) |x_j - y_j| = k|x - y|. \end{aligned}$$

Ceci prouve que $T: \mathbb{R}^n \rightarrow \mathbb{R}^n$ est contractante de rapport $k = 1 - c$. La suite de l'énoncé découle du théorème du point fixe. \square