République Algérienne Démocratique Et Populaire
Université Mohamed Boudiaf De M'sila
Faculté Des Mathématiques Et De L'informatique
Département De Mathématiques

MEMOIRE DE FIN D'ETUDE

Présenté pour l'obtention du Diplôme de **MASTER**

Domaine : Mathématiques et Informatiques
Filière: Mathématiques
Option: Analyse Mathématique et Numérique

Par
SERAICHE Fatima

Sujet

Approximation des valeurs propres et moteur de recherche Google

Date de soutenance : 20 Juin 2018

Devant le jury :

Mr.GAGUI Bachir	MCA. Univ de M'sila	Président
Mr. LAKEHALI Belkacem	MCA. Univ de M'sila	Rapporteur
Mr. BENSALOUA Cheniti	MCB. Univ de M'sila	Examinateur

Promotion : 2017 / 2018

Dédicace

Au nom de ALLAH chément et le miséricordieux

Je dédie ce travail

A mon père Ahmed

Vraiment aucune dédicace ne saurait exprimer mon amour et mon affection je vous offre ce modeste travail en témoignage de tous les sacrifices et l'immense tendresse dont vous m'avez toujours su me compiler

Puisse dieu tout puissant tu garder et tu procurer santé et bonheur

A mes très chers frères

A mes très chères sœurs

et surtout à ma sœur Samira

A tout ma famille SERAICHE sans exception

A mes amis Hadjira, Nadia et Zahra

A tous mes collègues de ma promotion et du département

A tous qui m'ont apporté du soutien toute ma vie

A tous mes enseignants.

Fatima

Table des matières

Remerciement

En préambule à ce mémoire, j'adresse ces quelques mots pour remercier notre Dieu tout puissant pour exprimer ma reconnaissance envers sa grande générosité. Dieu m'a donné la volonté, la patience, la santé et la confiance durant toutes mes années d'études.

Je tiens à remercier particulièrement mon directeur de thèse Monsieur **LAKEHALI Belkasem**, pour tout l'aide qu'il ma apporté et leur patience, leurs conseils et pour avoir guidé ce travail avec beaucoup d'intérêt.

Je lui suis également reconnaissant

pour la confiance qu'il ma accordée.

Ma sincère reconnaissance à tous les membres du jury pour l'honneur qu'ils me font en acceptant de présider et examiner ce travail.

Je souhaite aussi adresser mes remerciements les plus sincères aux personnes qui m'ont apporté leurs aides et qui ont contribué à l'élaboration de ce mémoire.

En effet, je voudrai remercier ma famille, mon encadreur et tous ceux qui ont participé de prés ou de loin à la réalisation de ce mémoire.

Également, un remerciement à tous mes collègues de promotion 2018 pour les bons moments qui nous avons passé ensemble.

Enfin, j'adresse mes plus sincères remerciements à tous mes proches et amis, qui m'ont toujours soutenu et encouragé au cours de la réalisation de ce mémoire.

Merci à tous et à toutes.

Introduction

En mathématique, et plus particulièrement en algèbre linéaire, le concept de valeur propre est une notion algébrique s'appliquant à une application linéaire d'un espace dans lui même. Son utilisation, cependant, dépasse maintenant de loin le cadre de ce présent mémoire. Il intervient aussi bien en mathématiques pures qu'appliquées.

L'approximation numérique des valeurs propres est un problème important rencontré dans le traitement d'équations différentielles, EDP et d'équation intégrales. Ainsi que l'étude des propriétés algébriques d'une matrice comme le calcul du conditionnement et le spectre. Cette approche permet aussi de résoudre de multiples problèmes de vibration, l'étude de l'équation de Schrödinger en mécanique quantique, la théorie des graphes, l'analyse des structures et le classement des pages web par les moteurs de recherche Google, Yahoo, Ask, Bing ....Ceci est le but de mon travail.

Les méthodes de calcul sont des algorithmes itératifs car le calcul exact des valeurs propres en général est impossible pour les matrices d'ordre $n \geq 5$. Nous étudierons deux types de méthodes: méthodes partiels fournissant seulement certaines valeurs propres et méthodes globales qui rendent compte de l'ensemble de tout le spectre.

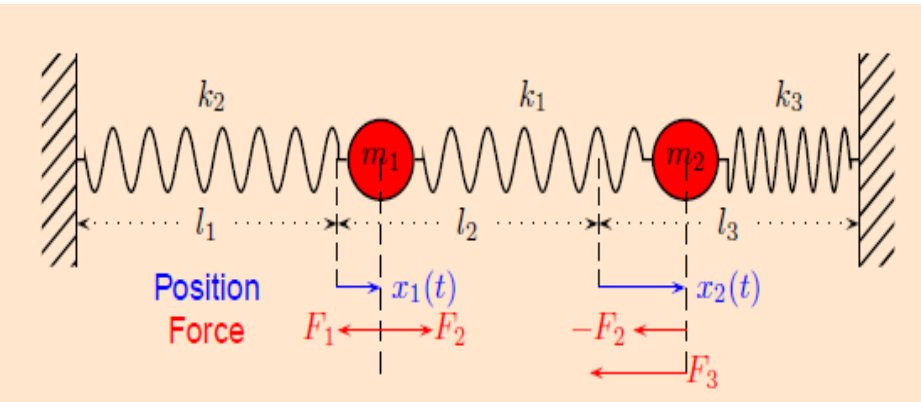
Chapitre 1

Motivation

Les valeurs propres des matrices ou des opérateurs linéaires jouent un rôle dans un très grand nombre d'applications, à la fois théoriques et pratiques. Nous allons essayer de transmettre une idée de l'étendue des applications en citant des exemples.

1.1 Mouvement des ressorts

Considérons un système de deux billes de masse unité reliées par trois ressorts de raideur unité. Notons $x_1(t)$ et $x_2(t)$ les positions des deux billes au temps t, par rapport à leur position d'équilibre. Soient $F_1(t)$, $F_2(t)$ et $F_3(t)$ les forces appliquées sur les billes : ce sont les forces de rappel des trois ressorts. Nous allons voir que l'étude des positions des billes au cours du temps $x_1(t)$ et $x_2(t)$, se ramène à un calcul de valeurs propres.



Etude du mouvement de deux billes maintenues par trois ressorts.

Pour cela, nous écrivons d'abord les équations de **Newton** pour les deux billes :

masse $\times$ accélération= somme des forces extérieures.

Ce qui se traduit par :

$$\begin{cases} m_1 x_1^{"}(t) = F_1(t) + F_2(t), \\ m_2 x_2^{"}(t) = -F_2(t) + F_3(t), \end{cases}$$

où

$$\begin{cases} F_1(t) = -k_1 x_1(t), \\ F_2(t) = k_2(x_2(t) - x_1(t)), \\ F_3(t) = -k_3 x_2(t). \end{cases}$$

Donc

$$\begin{cases} x_1''(t) = -\dfrac{k_1 + k_2}{m_1} x_1(t) + \dfrac{k_2}{m_1} x_2(t), \\ x_2''(t) = \dfrac{k_2}{m_2} x_1(t) - \dfrac{k_2 + k_3}{m_2} x_2(t). \end{cases}$$

Sous forme matricielle :

$$\overrightarrow{x}''(t) = -A\overrightarrow{x}(t) \qquad \text{avec} \quad A = \begin{pmatrix} \dfrac{k_1 + k_2}{m_1} & -\dfrac{k_2}{m_1} \\ -\dfrac{k_2}{m_2} & \dfrac{k_2 + k_3}{m_2} \end{pmatrix}.$$

Les positions des billes s'écrivent sous la forme :

$$\begin{cases} x_1(t) = C_1 cos(\omega t), \\ x_2(t) = C_2 cos(\omega t), \end{cases}$$

où les grandeurs C_1, C_2 et ω restant à déterminer.

En substituant ces relations dans les équations du mouvement, et après simplification par $cos(\omega t)$, nous avons:

$$A\begin{pmatrix} C_1 \\ C_2 \end{pmatrix} = \omega^2 \begin{pmatrix} C_1 \\ C_2 \end{pmatrix}.$$

Le problème revient donc à chercher $C_1 \neq 0$, $C_2 \neq 0$ et ω tels que les deux équations ci-dessus soient satisfaites.

Cela revient à chercher les valeurs propres de la matrice A.

1.2 Dynamique des populations

Divers modèles mathématiques ont été proposés pour prédire l'évolution de certaines espèces (humaines ou animales). Le modèle le plus simple, introduit par Lotka en 1920 et formalisé 20 ans plus tard par Leslie, est basé sur le taux de mortalité et de fécondité pour différentes tranches d'âge i = 0, . . ., n.

Soit $x_i^{(t)}$ le nombre de femelles (les mâles n'interviennent pas dans ce modèle) dont l'âge au temps t appartient à la i-ème tranche. On suppose que les valeurs de $x_i^{(0)}$ sont données. Notons s_i le taux de survie des femelles de la i-ème tranche, et m_i le nombre moyen de femelles engendrées par des femelles de la i-ème tranche d'âge.

Le modèle de Lotka et Leslie est défini par les équations

$$\begin{aligned} x_{i+1}^{(t+1)} &= x_i^{(t)} \quad si\ i = 0, ..., n-1, \\ x_0^{(t+1)} &= \sum_{i=0}^{n} x_i^{(t)} m_i. \end{aligned}$$

Les n premières équations décrivent le développement de la population, la dernière sa reproduction. Sous forme matricielle, cela donne

$$x^{(t+1)} = Ax^{(t)},$$

où $x^{(t)} = (x_0^{(t)}, ..., x_n^{(t)})^t$ et A est la matrice de Leslie

$$A = \begin{pmatrix} m_0 & m_1 & . & . & . & . & . & m_n \\ s_0 & 0 & . & . & . & . & . & 0 \\ . & s_1 & 0 & . & . & . & . & 0 \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & 0 & . \\ 0 & 0 & . & . & . & . & s_{n-1} & 0 \end{pmatrix}.$$

La dynamique de cette population est déterminée par la valeur propre de module maximal de A: λ_1, tandis que la distribution des individus dans les différentes tranches d'âge (normalisée par la population totale), est obtenue comme la limite de $x^{(t)}$ pour $t \to \infty$ et vérifie $Ax = \lambda_1 x$ [7].

Chapitre 2

Les valeurs propres

2.1 Rappel sur les systèmes linéaires

Le système $Mx = b$ avec M matrice carrée admet :

- Soit une solution unique si $\det(M) \neq 0$.
- Soit pas ou infinité des solutions si $\det(M) = 0$.

Dans le cas $\det(M) = 0$ on dit que M est singulière.

Puisque $x = 0$ est toujours solution de $Mx = 0$ et que l'on cherche des solutions $x \neq 0$, on se place dans le cas où $M = A - \lambda I$ est singulière.

Autrement dit on cherche les valeurs propres de A en résolvant l'équation :

$$\det(A - \lambda I) = 0.$$

Théoriquement la solution de $Ax = \lambda x$ se fait en 3 étapes systématiques:

1. Calcul du déterminant $|A - \lambda I|$.
2. Détermination de racines du polynôme caractéristique obtenu en écrivant :

$$\det(A - \lambda I) = 0.$$

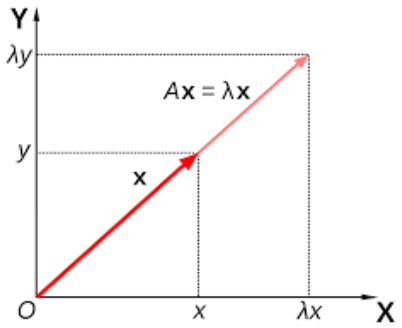
3. Pour chaque racine (chaque valeur propre) on doit résoudre le système linéaire

$$Ax = \lambda x,$$

afin de déterminer le où les vecteurs propres associés.

En pratique, cette démarche n'est pas exploitable pour les systèmes de taille supérieure à 4 ou 5 car le calcul formel de $\det(A - \lambda I)$ est rapidement monstrueux.

On préfère donc les méthodes itératives dans lesquelles on s'approche pas à pas à des valeurs propres.



La matrice étire le vecteur x sans changer sa direction. x est un vecteur propre pour la valeur propre λ.

2.2 Définition et propriétés

Définition 2.2.1 *Pour une matrice carrée A d'ordre n on dit que $\lambda \in \mathbb{C}$ est une* ***valeur propre*** *de A s'il existe un vecteur $v \neq 0 \in \mathbb{C}^n$ tels que :*

$$Av = \lambda v.$$

L'élément v est appelé **vecteur propre** de A associé à λ. Ce dernier n'est pas unique, en effet tous les vecteurs αx avec $\alpha \neq 0$, réel ou complexe, sont aussi des vecteurs propres associés à λ.

Quelques propriétés:

- Les valeurs propres peuvent être complexes même si la matrice est réelle.
- Les valeurs propres d'une matrice symétrique sont toutes réelles.
- Si A est diagonale, les valeurs propres de A sont égales aux éléments diagonaux.
- Deux matrices semblables ont les mêmes valeurs propres.
- Si λ valeur propre alors $\lambda - \mu$ et $\dfrac{1}{\lambda - \mu}$ sont respectivement valeurs propres de $A - \mu I$ et $(A - \mu I)^{-1}$ pour tout $\mu \in \mathbb{R}$.
- Si toutes les valeurs propres d'une matrice carrée sont strictement positives On dit que cette matrice est symétrique définie positive (noté $A \succ 0$).
- Pour une matrice carrée $A_{(n)} = (a_{ij})_{1 \leq i,j \leq n}$, on a :

  (i) $tr(A) = \sum_{i=1}^{n} a_{ii} = \sum_{i=1}^{n} \lambda_i.$

  (ii) $det(A) = \prod_{i=1}^{n} a_{ii} = \prod_{i=1}^{n} \lambda_i.$

2.3 Polynôme caractéristique

La première méthode que nous proposons était déjà utilisé par Joseph Luis Lagrange(1736-1813) pour calculer les valeurs propres d'une matrice. Elle consiste à calculer les racines du polynôme caractéristique.

Le polynôme caractéristique d'une matrice A est le polynôme défini par :

$$p(\lambda) = \det(A - \lambda I).$$

Les valeurs propres de A sont ses racines.

Si A est de dimension inferieur à 3 on peut faire le calcul en arithmétique exact, cet algorithme peut être très utile. En revanche, si nous faisons le calcul des erreurs d'arrondi, cet algorithme peut donner de mauvais surprises.

On sait depuis **Galois(1811-1832)** et **Abel(1802-1829**) qu'on ne peut pas calculer par opérations élémentaires les racines d'un polynôme quelconque de degré supérieur ou égal à 5. On préfère donc les méthodes itératives dans lesquelles on approche pas à pas des valeurs propres.

2.4 Polynôme Wilkinson

Pour illustrer le problème avec le calcul des racines d'un polynôme, on peut prendre l'exemple classique du polynôme de Wilkinson(1757-1825) :

$$w(x) = \prod_{i=1}^{20}(x-i) = (x-1)(x-2)...\,(x-19)\,(x-20).$$

Ce polynôme de degré 20 est exprimé sous forme factorisé, et les racines sont $x_i = 1, ..., 20$. Si le problème des racines d'un polynôme est bien conditionné, on s'attend à ce qu' un petit changement dans les coefficients du polynôme ramène à un petit changement dans les valeurs des racines. Malheureusement ce n'est pas le cas: le polynôme x^2 a une racine double en $x = 0$, mais $x^2 - \varepsilon$ a une paire de racine $x = \pm\sqrt{\varepsilon}$. Pour epsilon petit, $\varepsilon \ll \sqrt{\varepsilon}$ et donc les racines sont très "loin" de 0 même quand les deux polynômes (x^2 et $x^2 - \varepsilon$) sont proches.

Ce type de problème peut se produire même quand les racines du polynôme sont bien espacés, comme dans le cas du polynôme de Wilkinson. Le coefficient de x^{19} du polynôme de Wilkinson est -210. Si on modifie le coefficient de $\varepsilon = 2^{-23}$, les racines du polynôme peuvent dévier significativement. Par exemple, la onzième racine devient $10,09549 + 0,64215i$, au lieu de 11.

La perturbation des coefficients provoque une grande erreur dans le calcul des racines du polynôme[16].

Chapitre 3

Approximation des valeurs propres

Comme on l'a vu dans les problèmes précédents, la connaissance du spectre de A n'est pas toujours nécessaire. Souvent, seules important les valeurs propres extrémales, c'est-à-dire celles ayant le plus grands et plus petits modules. Ils peut être utilisé pour analyser des méthodes numériques (la vitesse de convergence pour la méthode de gradient conjugue), ils permettent aussi de calculer la valeur optimale du paramètre d'accélération de la méthode de Richardson et le conditionnement d'une matrice symétrique définie positive ($cond(A) = \frac{\lambda_{\min}}{\lambda_{\max}}$). La méthode de la puissance et puissance inverse permettant de calculer cette quantité.

Nous présentons d'abord quelques résultats intermédiaires sur la localisation des valeurs propres.

3.1 Localisation des valeurs propres

Les valeurs propres de A étant les racines du polynôme caractéristique $p_A(\lambda)$, on ne peut les calculer qu'avec des méthodes itératives quand $n \geq 5$. Il est donc utile de connaître leur localisation dans le plan complexe pour accélérer la convergence.

Une première estimation est donnée par la proposition suivante:

Proposition 3.1.1 *Si $\|.\|$ est une norme matricielle consistante alors*

$$\rho(A) \leq \|A\| \qquad \forall A \in \mathbb{C}^{n\times n},$$

où $\rho(A) = max\{|\lambda|, \lambda$ est valeur propre de $A\}$ est appelé le rayon spectral de A [8].

Preuve. Si λ est une valeur propre de A alors il existe $v \neq 0$, vecteur propre de A, tel que $Av = \lambda v$. Ainsi, puisque $\|.\|$ est consistante:

$$|\lambda| \|v\| = \|\lambda v\| = \|Av\| \leq \|A\| \|v\|,$$

et donc $|\lambda| \leq \|A\|$.

Cette inégalité étant vraie pour toute valeur propre de A, elle est en particulier quand $|\lambda|$ est égal au rayon spectral.

Cela entraîne donc que le spectre de A est inclus dans le disque centré en 0 et de rayon $||A||$. ■

Le théorème suivant permet de définir une région plus restreinte que ce disque.

Théorème 3.1.1 ***Gershgorine***

Soit $A = (a_{ij})_{1 \leq i,j \leq n}$ une matrice carrée complexe de taille n. On appelle disque de ***Gershgorine*** *les n disques définies par :*

$$D_i = \left\{ z \in \mathbb{C} \ : |z - a_{ii}| \leqslant \sum_{i \neq j} |a_{ij}| \right\} \qquad i = 1, ..., n.$$

Si λ est une valeur propre de A, alors λ appartient à l'un des disques de ***Gershgorine*** *de la matrice A.*

Preuve. Soit λ une valeur propre de A et x un vecteur non nul associe à λ, c'est-à-dire:

$$Ax = \lambda x.$$

Soit $i \in \{1, ..., n\}$ tel que $|x_i| = \max_j |x_j|$,
on a alors $x_i \neq 0$ et on peut écrire

$$\begin{aligned} \sum_{j=1}^{n} a_{ij} x_j = \lambda x_i \quad &\Rightarrow \quad \lambda - a_{ii} = \sum_{i \neq j} a_{ij} \frac{x_j}{x_i}, \\ &\Rightarrow \quad |\lambda - a_{ii}| \leq \sum_{i \neq j} |a_{ij}| \frac{|x_j|}{|x_i|} \leq \sum_{i \neq j} |a_{ij}|. \end{aligned}$$

Ce qui montre que $\lambda \in D_i$. ■

3.2 Méthode de la puissance

La méthode de la puissance est une méthode itérative qui fournit une très bonne approximation des valeurs propres extrémales d'une matrice quelconque A et des vecteurs propres associés. On notera λ_1 et λ_n les valeurs propres ayant respectivement le plus grand et le plus petit module.

Le principe de la méthode repose sur le fait qu'en appliquant un grand nombre de fois la matrice sur un vecteur initial quelconque, les vecteurs successifs vont prendre une direction qui se rapproche du vecteur propre de la grande valeur propre (en valeur absolue).

Supposons que les valeurs propres de A soient rangées comme suit :

$$|\lambda_1| \ > |\lambda_2| \geqslant .... \geqslant |\lambda_n| .$$

Remarquer, on particulier, que $|\lambda_1|$ est distinct des autres modules des valeurs propres de A.

Supposons que les différents vecteurs propres forment une base de $\mathbb{C}^n$

$$v^{(1)}, v^{(2)}, ..., v^{(n)}.$$

On prend un vecteur complexe quelconque $w^{(0)}$ de départ. On a

$$w^{(0)} = \alpha_1 v^{(1)} + \alpha_2 v^{(2)} + ... + \alpha_n v^{(n)},$$

avec $\alpha_i \, \epsilon \, \mathbb{R} \ \ , i = 1, ..., n.$

On calcul ensuite:

$$\begin{cases} w^{(1)} = Aw^{(0)}, \\ w^{(2)} = Aw^{(1)} = A^2 w^{(0)}, \\ \vdots \\ w^{(k)} = Aw^{(k-1)} = A^k w^{(0)}. \end{cases}$$

Finalement on a:

$$\begin{aligned} w^{(k)} &= A^k w^{(0)}, \\ &= A^k(\alpha_1 v^{(1)} + \alpha_2 v^{(2)} + ... + \alpha_n v^{(n)}), \\ &= \alpha_1 \lambda_1^k v^{(1)} + ... + \alpha_n \lambda_n^k v^{(n)}. \end{aligned}$$

Cela peut se réecrire comme

$$w^{(k)} = \lambda_1^k(\alpha_1 v^{(1)} + \alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k v^{(2)} + ... + \alpha_n \left(\frac{\lambda_n}{\lambda_1}\right)^k v^{(n)}).$$

Le quotient de Rayleigh satisfait(si $\alpha_1 \neq 0$):

$$\frac{(w^{(k)})^t A w^{(k)}}{(w^{(k)})^t w^{(k)}} \to \lambda_1.$$

Voir[6].

Comme $|\lambda_1| > |\lambda_j|$ pour tout $j \neq 1$, tous les termes $\left(\frac{\lambda_j}{\lambda_1}\right)^k$ tend vers 0 quand k tend vers l'infini.

Remarque 3.2.1 *Notons bien que d'un point de vue pratique la méthode de la puissance ne peut pas être implémentée en l'état car l'algorithme pourrait conduire à un "overflow", c'est-à-dire que les valeurs $w^{(k+1)}$ sont susceptibles de devenir de plus en plus grands. Pour remédier à cela et pour fixer la norme du vecteur limite de la suite $(w^{(k)})_{k\epsilon\mathbb{N}}$, nous ajoutons une étape de normalisation de sorte que*

$$z^{(k)} = \frac{w(k)}{\|w(k)\|} \qquad w^{(k+1)} = Az^{(k)}$$

c'est-à-dire que nous fixons la norme de chaque itéré égale à 1.

Proposition 3.2.1 ***(Formule de rayon spectral)*** *Soit A une matrice d'ordre n diagonalisable dont la valeur propre de plus grande module λ_1 est unique. Soit q_0 un vecteur de $\mathbb{R}^n$ qui n'est pas orthogonal au sous-espace propre à gauche associé à λ_1. Alors la suite définie par $x^{(k)} = Ax^{(k-1)}$ vérifie*

1 $q = \lim_{k\to\infty} \left(\frac{\lambda_1}{|\lambda_1|}\right)^k q^{(k)}$ *est un vecteur propre de norme 1 associé à λ_1,*

2 $\lim_{k\to\infty} \left\|Aq^{(k)}\right\| = |\lambda_1|$.

Preuve. Voir[11] ■

3.2.1 Condition de convergence

Si le vecteur initial $w^{(0)}$ n'appartient pas au sous-espace vectoriel engendré par les $(n-1)$ vecteurs propres de A alors la méthode converge.

Remarque 3.2.2 *Si* $|\lambda_1| \approx |\lambda_2|\,(...|\lambda_{n-1}| \approx |\lambda_n|)$ *alors la convergence sera trés lente car* $\dfrac{|\lambda_2|}{|\lambda_1|} \ll 1$ *doit être vrai pour avoir une convergence rapide* $[voir[3], page392]$.

Le choix du vecteur initial influe beaucoup sur la vitesse de convergence. Si le vecteur initial est près d'un vecteur propre alors les coefficients α_i $i \neq 1$ *seront petit par rapport à* α_1.

Exemple 3.2.1 $([5],\ page320)$.

Soit $A = \begin{pmatrix} 1.8 & 0.8 \\ 0.2 & 1.2 \end{pmatrix}$ $v_1 = (4,1)^t$ $x = (-0.5,1)^t$,

les valeurs propres de A *sont* $\lambda_1 = 2$ *et* $\lambda_2 = 1$.

Pour $k = 1,...,8$. *On calcule* $A^k x$.

Qu'est-ce qui se passe quand k *tend vers l'infini ?*

Les trois premiers expressions de $A^k x$ *sont:*

$$\begin{aligned} Ax &= \begin{pmatrix} 1.8 & 0.8 \\ 0.2 & 1.2 \end{pmatrix}\begin{pmatrix} -0.5 \\ 1 \end{pmatrix} = \begin{pmatrix} -0.1 \\ 1.1 \end{pmatrix}, \\ A^2x &= A(Ax) = \begin{pmatrix} 1.8 & 0.8 \\ 0.2 & 1.2 \end{pmatrix}\begin{pmatrix} -0.1 \\ 1.1 \end{pmatrix} = \begin{pmatrix} 0.7 \\ 1.3 \end{pmatrix}, \\ A^3x &= A(A^2x) = \begin{pmatrix} 1.8 & 0.8 \\ 0.2 & 1.2 \end{pmatrix}\begin{pmatrix} 0.7 \\ 1.3 \end{pmatrix} = \begin{pmatrix} 2.3 \\ 1.7 \end{pmatrix}, \end{aligned}$$

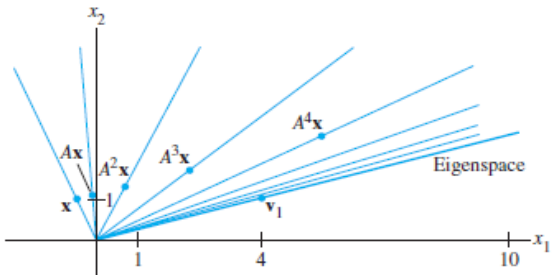
pour $k = 0,...,8$

k	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
$A^k x$	-0.5	-0.1	0.7	2.3	5.5	11.9	24.7	50.3	101.5
	1	1.1	1.3	1.7	2.5	4.1	7.3	13.7	26.5

les vecteurs

$$x, Ax, ..., A^k x, ...$$

sont montrés dans la figure en bas, les autres vecteurs deviennent trop longs à afficher, mais des segments de ligne montrent les directions de ces vecteurs. En fait les directions des vecteurs sont ce que nous voulons vraiment voir pas les vecteurs eux-mêmes. La ligne représentant l'espace v_1 plus précisément l'angle entre les lignes déterminé par $A^k x$ tend vers zéro quand $k \to \infty$.



Les directions déterminées par

$x, Ax, A^2x, ..., A^7x$

Exemple 3.2.2 *Appliquons la méthode de la puissance sur* $A = \begin{pmatrix} 6 & 5 \\ 1 & 2 \end{pmatrix}$,

avec le vecteur initial $x_0 = (0,1)^t$.

$$Ax_0 = \begin{pmatrix} 6 & 5 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 5 \\ 2 \end{pmatrix} \quad , x_1 = \frac{Ax_0}{\|Ax_0\|_\infty} = \begin{pmatrix} 1 \\ 0.4 \end{pmatrix},$$

$$Ax_1 = \begin{pmatrix} 6 & 5 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 0.4 \end{pmatrix} = \begin{pmatrix} 8 \\ 1.8 \end{pmatrix} \quad , x_2 = \frac{Ax_1}{\|Ax_1\|_\infty} = \begin{pmatrix} 1 \\ 0.225 \end{pmatrix},$$

$$Ax_2 = \begin{pmatrix} 6 & 5 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 0.225 \end{pmatrix} = \begin{pmatrix} 7.125 \\ 1.450 \end{pmatrix} \quad ,x_3 = \frac{Ax_2}{\|Ax_2\|_\infty} = \begin{pmatrix} 1 \\ 0.2035 \end{pmatrix}.$$

Les résultats de MATLAB pour les 5 premieres itérations avec $\mu_k = \max_i |(Ax_k)_i|$ sont:

k	0	1	2	3	4	5
x_k	0	1	1	1	1	1
	1	0.4	0.225	0.2035	0.2005	0.20007
Ax_k	5	8	7.125	7.0175	7.0025	7.00036
	2	1.8	1.450	1.4070	1.4010	1.40014
μ_k	5	8	7.125	7.0175	7.0025	7.00036

On remarque que $\{x_k\}$ approche à $(1, 0.2)^t$ et μ_k est pproche de 7, donc $(1, 0.2)^t$ est le vecteur propre et 7 est la plus grand valeur propre. Il est facillement vérifier que:

$$A \begin{pmatrix} 1 \\ 0.2 \end{pmatrix} = \begin{pmatrix} 6 & 5 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 0.2 \end{pmatrix} = \begin{pmatrix} 7 \\ 1.4 \end{pmatrix} = 7 \begin{pmatrix} 1 \\ 0.2 \end{pmatrix}.$$

Si on choisit $x_0 = (-5, 1)^t$ la méthode ne converge pas car $x_0 = (-5, 1)^t$ est un vecteur propre.

3.2.2 Avantages

On n'utilise que des produits de matrices par vecteurs.

Le calcul, même avec des matrices de grande taille, ne prend pas de place en mémoire.

3.2.3 Inconvénients

Si les valeurs propres sont très proches, la convergence sera lente.

L'intérêt de la méthode des puissances itérées est de ne pas nécessiter d'hypothèses très particulières sur la matrice, comme être symétrique. En revanche, elle demande une configuration particulière des valeurs propres qui n'est pas toujours vérifiée. De plus, elle ne permet de calculer qu'une seule des valeurs propres de A.

Si la matrice est inversible, on peut modifier la méthode pour la valeur propre de plus petit module.

3.3 Méthode de la puissance inverse

La méthode de la puissance inverse est un algorithme permettant de calculer la valeur propre de plus petit module d'une matrice, ou plus généralement, la valeur propre la plus proche d'un nombre complexe donné. Elle est basée sur la méthode de la puissance itérée appliquée à l'inverse de la matrice A.

$$Av = \lambda v \quad \Leftrightarrow \quad A^{-1}v = \frac{1}{\lambda}v.$$

Étant donné un vecteur initial $x^{(0)}$, on pose

$$y^{(0)} = \frac{x^{(0)}}{\|x^{(0)}\|},$$

et on calcule pour $k = 1, 2, ...$

$$\begin{aligned} x^{(k)} &= A^{-1}y^{(k-1)} \\ y^{(k)} &= \frac{x^{(k)}}{\|x^{(k)}\|} \\ \mu^{(k)} &= (y^{(k)})^t A^{-1} y^{(k)} \end{aligned}$$

Si les vecteurs propres de A sont linéairement indépendants, et s'il n'y a qu'une valeur propre λ_n de module minimal, alors

$$\lim_{k \dashrightarrow \infty} \mu^{(k)} = 1/\lambda_n,$$

i.e. $(\mu^{(k)})^{-1}$ tend vers λ_n pour $k \to \infty$.

A chaque étape k, on doit résoudre un système linéaire de la forme

$$Ax^{(k)} = y^{(k-1)}.$$

Il est donc commode d'effectuer une factorisation LU de A une fois pour toute, afin de n'avoir à résoudre que deux systèmes triangulaires à chaque itération.

Rappelons que la commande lu (par MATLAB) peut également effectuer la décomposition LU pour des matrices complexes.

Une autre généralisation de la méthode de la puissance permet de calculer une approximation de la valeur propre (inconnue) la plus proche d'un μ donné (réel ou complexe).

Notons λ_μ une telle valeur propre et définissons la matrice translatée $A_\mu = A - \mu I$, dont les valeurs propres sont

$$\lambda(A_\mu) = \lambda(A) - \mu.$$

Pour approcher λ_μ, on peut d'abord estimer $\lambda_{min}(A_\mu)$, valeur propre de plus petite norme de A_μ, en appliquant la méthode de la puissance inverse à A_μ, puis calculer $\lambda_\mu = \lambda_{min}(A_\mu) + \mu$.

Cette technique est connue sous le nom de **méthode de la puissance avec décalage** ou avec **translation** (shift en anglais), et le nombre μ est appelé décalage (ou shift).

En général, pour résoudre à chaque étape le système, on ne calcule pas l'inverse de la matrice, mais on réalise sa décomposition LU.

La suite (x_k) converge vers l'inverse de la plus grande valeur propre de A^{-1}, donc vers la valeur propre de A de plus petit module.

Exemple 3.3.1 *Soit* $A = \begin{pmatrix} 10 & -8 & -4 \\ -8 & 13 & 4 \\ -4 & 4 & 4 \end{pmatrix}$.

Les valeurs propres sont $\lambda_1 = 21$, $\lambda_2 = 3.3$ *et* $\lambda_3 = 1.9$.

Utilisant la méthode de puissance inverse pour déterminer la plus petite valeur propre de la matrice A.

Les deux petites valeurs sont proches entre eux, alors on applique la méthode de puissance inverse à $(A - 1,9I)$, *les résultats de calcul MATLAB sont présentés dans le tableau qui en bas.*

Le choix du vecteur initial x_0 *est arbitraire, et* $\mu_k = \max_i(y_k)_i$

k	0	1	2	3	4
	1	0.5736	0.5054	0.5004	0.50003
x_k	1	0.0646	0.0045	0.0003	0.00002
	1	1	1	1	1
	4.45	5.0131	5.0012	5.0001	5.000006
y_k	0.50	0.0442	0.0031	0.0002	0.000015
	7.76	9.9197	9.9949	9.9996	9.999975
μ_k	7.76	9.9197	9.9949	9.9996	9.999975
v_k	2.03	2.0008	2.00005	2.000004	2.0000002

Voir [[5], *page* 323].

3.4 Méthode de Jacobi

Il s'agit d'une méthode, assez ancienne, s'emploie lorsque l'on cherche toutes les valeurs propres d'une matrice symétrique A. La méthode revient à effectuer une suite de transformation de type rotation planaire qui permet d'annuler un élément (p, q), où p et q sont deux entiers de la matrice A. Chaque rotation élémentaire fait intervenir une matrice orthogonale P_{pq}. On construit ainsi une suite des matrices symétriques $A^{(k)}$ qui tend vers la matrice diagonale semblable à A.

On pose $A^{(1)} = A$ et, à chaque transformation k, on construit la matrice $A^{(k+1)}$ défnie par :

$A^{(k+1)} = P_{pq}^{(k)} A^{(k)} P_{pq}^{(k)}$

Les éléments de la matrice symétrique $A^{(k+1)}$ sont donnés par :

$$\begin{cases} a_{ij}^{(k+1)} = a_{ij}^{(k)} & \text{pour } i \neq p, q \text{ et } \ j \neq p, q. \\ a_{ip}^{(k+1)} = a_{ip}^{(k)} cos\theta - a_{iq}^{(k)} sin\theta & \text{pour } i \neq p, q. \\ a_{iq}^{(k+1)} = a_{ip}^{(k)} sin\theta + a_{iq}^{(k)} cos\theta & \text{pour } \ i \neq p, q. \\ a_{pp}^{(k+1)} = a_{pp}^{(k)} \cos 2\theta + a_{qq}^{(k)} \sin 2\theta - 2a_{pq}^{(k)} cos\theta sin\theta. \\ a_{qq}^{(k+1)} = a_{pp}^{(k)} sin2\theta + a_{qq}^{(k)} cos2\theta + 2a_{pq}^{(k)} cos\theta sin\theta. \end{cases}$$

où l'angle $\theta \in \left]-\frac{\pi}{4}, 0\right[\sqcup \left]0, \frac{\pi}{4}\right[$ vérifié : $tan2\theta = \dfrac{2a_{pq}^{(k)}}{a_{qq}^{(k)} - a_{pp}^{(k)}}$.

Deux méthodes pour le choix des entiers p et q :

1. Les deux entiers peuvent être choisis de façon cyclique: $p = 1$, $q = 2$ puis ($p = 1$, $q = 3$), etc..

2. Les deux entiers peuvent être choisis tel que l'élément $a_{pq}^{(k)}$ soit la plus grande valeur hors éléments de la diagonale.

Cas de la dimension deux

$$P = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}.$$

Cas de la dimension trois

$$P = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{pmatrix}.$$

Cas général

$$(P_{pq})_{i,j} = \begin{cases} 1 & \text{si } i = j, \text{avec } \ j \neq p, \ j \neq q. \\ \cos\theta & \text{si } i = j = p \ ou \ i = j = q. \\ \sin\theta & \text{si } \ i = p \ \text{ et } j = q. \\ -\ sin\ \theta & \text{si } i = q \ \text{ et } \ j = p. \\ 0 & \text{sinon.} \end{cases}$$

Nous avons aussi

$$P = \begin{pmatrix} 1 & 0 & ... & ... & ... & 0 & 0 \\ 0 & \cos\theta & ... & ... & \sin\theta & ... & 0 \\ ... & 0 & 1 & 0 & ... & ... & ... \\ ... & ... & ... & ... & ... & ... & ... \\ ... & -\sin\theta & ... & ... & \cos\theta & ... & ... \\ 0 & ... & ... & ... & ... & ... & 0 \\ 0 & 0 & ... & ... & ... & 0 & 1 \end{pmatrix} \begin{matrix} \\ \\ \leftarrow p \\ \\ \\ \leftarrow q \\ \\ \end{matrix}$$

.

Exemple 3.4.1 *Voir[14]*

Appliquons la méthode de Jacobi sur la matrice A:

$$A^{(1)} = \begin{pmatrix} 0 & 1 & 6 & 0 & 0 & 0 \\ 1 & 0 & 2 & 7 & 0 & 0 \\ 6 & 2 & 0 & 3 & 8 & 0 \\ 0 & 7 & 3 & 0 & 4 & 9 \\ 0 & 0 & 8 & 4 & 0 & 5 \\ 0 & 0 & 0 & 9 & 5 & 0 \end{pmatrix} \Longrightarrow A^{(31)} = \begin{pmatrix} 16.61 & 0 & 0 & 0 & 0 & 0 \\ 0 & 5.94 & 0 & 0 & 0 & 0 \\ 0 & 0 & -2.46 & 0 & 0 & 0 \\ 0 & 0 & 0 & -12.13 & 0 & 0 \\ 0 & 0 & 0 & 0 & -10.06 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2.20 \end{pmatrix}$$

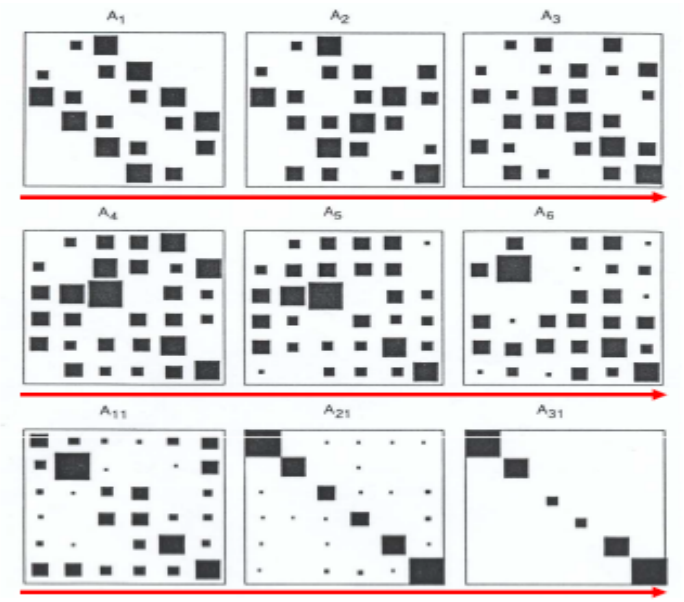


Illustration de la méthode de Jacobi.

3.5 Méthode de QR

Nous présentons dans cette section une méthode itérative pour approcher simultanément toutes les valeurs propres d'une matrice A. L'idée de base est de transformer A en une matrice semblable pour laquelle le calcul des valeurs propres est plus simple, voir[8].

La méthode QR est la méthode la plus couramment utilisée pour le calcul de l'ensemble des valeurs propres d'une matrice quelconque de taille moyenne, soit $n < 100$, notamment non symétrique. De plus, étant basée sur la décomposition QR, où Q est une matrice orthogonale (unitaire en complexes) et R une matrice triangulaire supérieure, puis on multiplie Q et R dans l'ordre inverse: $A^{(1)} = QR$. Cette nouvelle matrice est similaire à A.

L'idée consiste à construire une suite des matrices $A^{(k)}$, toutes semblables à A. Après avoir posé $A^{(0)} = A$, de plus, étant basée sur la décomposition QR, pour calculer les matrices carrées $Q^{(k+1)}$ et $R^{(k+1)}$ pour $k = 0, 1, ...$telles que

$$Q^{(k+1)}R^{(k+1)} = A^{(k)},$$

puis on pose $A^{(k+1)} = R^{(k+1)}Q^{(k+1)}$.

Les matrices $A^{(k)}$, $k = 1, 2, ...$ sont toutes semblables, elles ont donc les mêmes valeurs propres que A.

Théorème 3.5.1 *Soit A une matrice carrée d'ordre n (réelle ou complexe). Il existe une matrice unitaire Q et une matrice triangulaire supérieure R telles que*

$$A = QR.$$

De plus, on peut s'arranger pour que les éléments diagonaux de la matrice R soient tous positifs ou nuls.

Si la matrice A est inversible, la factorisation $A = QR$ correspondante est unique.

Preuve. On suppose pour simplifier que A est inversible. Donc les vecteurs colonnes de A forment une base de $\mathbb{R}^n$.

On effectue l'orthonormalisation de cette base par le procédé de **Gram-Schmidt**. On obtient une matrice orthogonal Q dont les vecteurs colonnes sont formés de cette base orthonormée.

On veut $A = QR$, donc on calcule $R = Q^tA$. Or $R_{ij} = \langle e_i, a_j \rangle$ qui est nul par définition dès que $i > j$. Donc R est triangulaire supérieure.

Soit $A = A^{(1)}$ une matrice carrée quelconque. On écrit sa factorisation QR, soit $A^{(1)} = Q^{(1)}R^{(1)}$, puis on forme la matrice $A^{(2)} = R^{(1)}Q^{(1)}$ et on recommence. Sous des hypothèses assez restrictives, on montre que les matrices $A^{(k)}$ deviennent triangulaires supérieures, au sens où les coefficients sub-diagonaux tendent vers 0. De plus, les termes diagonaux tendent vers les valeurs propres de la matrice. Cependant, les matrices $A^{(k)}$ ne convergent pas car on ne sait rien du comportement des coefficients sur-diagonaux. ■

Remarque 3.5.1 *1. La méthode **QR** permet d'approcher toutes les valeurs propres d'une matrice A.*

2. Dans sa version de base, on a un résultat de convergence si A est à coefficients réels et a des valeurs propres distinctes.

3. Sa vitesse de convergence asymptotique dépend du plus grand quotient des modules de deux valeurs propres successives.

Proposition 3.5.1 ***(Convergence de la méthode QR)***

Soit $A \in \mathbb{R}^{n\times n}$ une matrice telle que

$$|\lambda_1| > |\lambda_2| > ... > |\lambda_n|.$$

Alors

$$\lim_{k\to+\infty} T^{(k)} = \lim_{k\longrightarrow\infty} A^{(k)} = T = \begin{bmatrix} \lambda_1 & t_{12} & ... & t_{1n} \\ 0 & . & . & . \\ . & . & \lambda_{n-1} & t_{n-1,n} \\ 0 & ... & 0 & \lambda_n \end{bmatrix}.$$

La vitesse de décroissance vers zéro des coefficients triangulaires inférieurs, $a_{i,j}^{(k)}$,$i > j$, quand k tend vers l'infini, dépend de $\max_i |\lambda_{i+1}/\lambda_i|$.

En pratique, on stoppe les itérations quand $\max_{i>j}|a_{i,j}^{(k)}| \leq \varepsilon$, où $\varepsilon > 0$ est une tolérance fixée.

Si de plus A est symétrique, la suite $\{A_{(k)}\}$ converge vers une matrice diagonale.

Si les valeurs propres de A, bien que distinctes, ne sont pas bien séparées, on déduit que la convergence de $T^{(k)}$ vers une matrice triangulaire peut être assez lente[8].

3.5.1 Transformation de Householder

•Une transformation orthogonale permet en un seul coup de faire apparaître des zéros sur toute une colonne: c'est la transformation de Householder.

•Pour x vecteur quelconque de taille m, on définit:

$$H = I - 2\frac{vv^t}{\|v\|^2} \qquad \text{avec } v = x + \|x\| e_1, \text{où } e_1 = \underbrace{(1,0,0,...,0)^t}_{\text{m composantes}}.$$

•On vérifie facilement que $H = H^t$ puis que $HH^t = I$. Par suite, on obtient $H^{-1} = H^t$ et donc H est bien orthogonale en plus d'être symétrique.

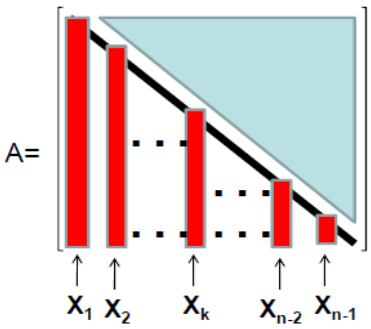
•Un calcul simple permet alors de vérifier que

$$Hx = -\|x\| e_1 = (-\|x\|, 0, 0, ..., 0)^t.$$

Voir[14].

3.5.2 Factorisation QR

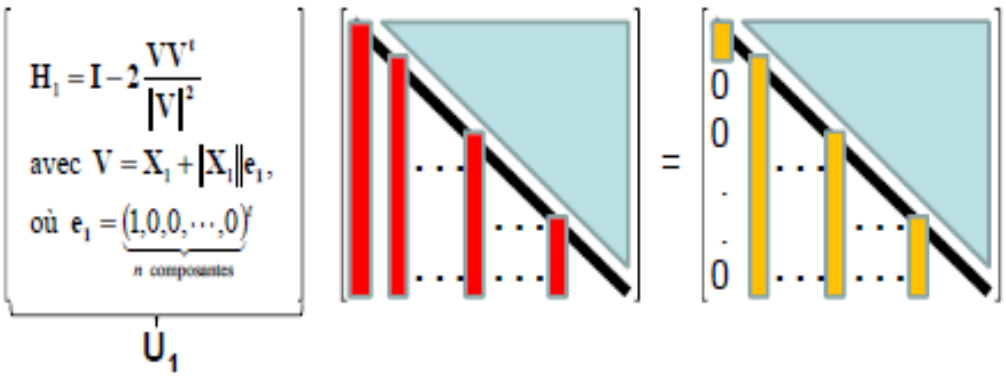
Partant d'une matrice A quelconque, on peut faire apparaître les vecteurs colonnes suivants



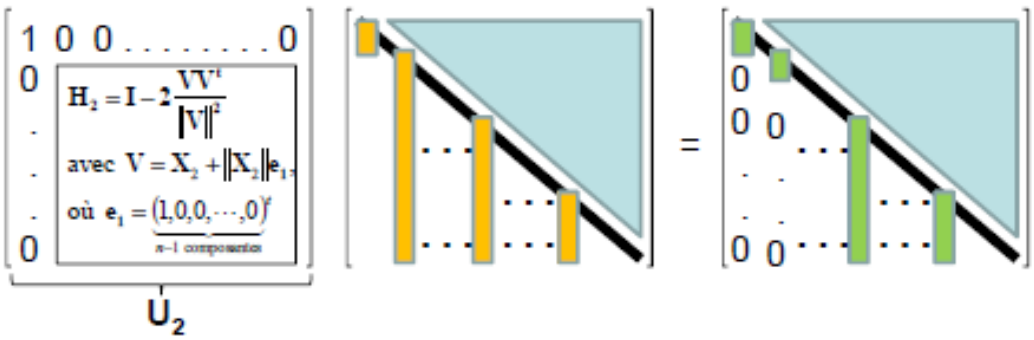
Les vecteurs colonnes X_1, X_2, ..., X_k, et X_{n-1} ont pour taille: $n, n-1, ..., n-k+1, ...2$ respectivement.

L'idée est d'appliquer une transformation de Householder adéquate successivement à chacun de ces vecteurs afin d'annuler tous les termes sous diagonaux.

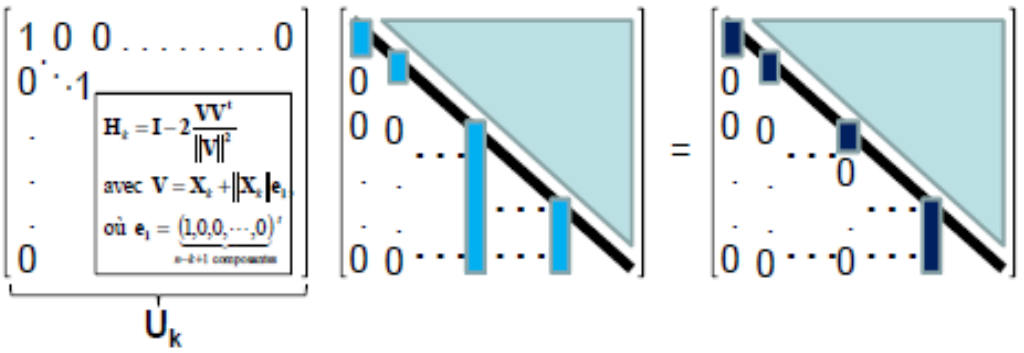
Plus précisément, pour annuler les $(n-1)$ dernières composantes de X_1



Pour anuler les $(n-2)$ dernières composantes de X_2



Pour annuler les $(n-k)$ dernières composantes du nouveau X_k



Au final, après avoir appliqué les transformations de Householder bien choisies on obtient (en posant $Q^t = U_{n-1}U_{n-2}...U_2U_1$) la factorisation QR de la matrice A:

$$A = QR,$$

avec Q matrice orthogonale et R matrice triangulaire supérieur, voir[14].

Exemple 3.5.1 $[Voir[14], page25]$.

On fait d'abord apparaitre des zéros sur la première colonne de la matrice A:

$$A = \begin{pmatrix} 1 & 3 & 2 & 1 \\ 1 & 2 & 1 & 2 \\ 2 & -1 & 2 & 1 \\ -1 & -2 & 3 & 4 \end{pmatrix}$$

$$x_1 = (1,1,2,-1)^t \Rightarrow v_1 = x_1 + \|x_1\| \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 4.6458 \\ 1 \\ 2 \\ -1 \end{pmatrix}.$$

$$U_1 = H_1 = I - 2\frac{v_1 v_1^t}{\|v_1\|^2} = \begin{pmatrix} -0.7559 & -0.3780 & 0.3780 & 0.3780 \\ -0.3780 & 0.9186 & 0.0814 & 0.0814 \\ 0.3780 & 0.0814 & 0.9186 & -0.0814 \\ 0.3780 & 0.0814 & -0.0814 & 0.9186 \end{pmatrix}.$$

$$U_1A = \begin{pmatrix} -2.6458 & -4.1576 & 0 & 0.3780 \\ 0 & 0.4593 & 0.5695 & 1.8661 \\ 0 & 0.5407 & 2.4305 & 1.1339 \\ 0 & -0.4593 & 3.4305 & 4.1339 \end{pmatrix}$$

.

On procède de la même manière pour les colonnes 2 et 3 des vecteurs:

$x_2 = (0.4593, 0.5407, -0.4593)^t$, $x_3 = (2.1945, 3.6310)^t$

$$v_2 = x_2 + \|x_2\| \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1.3045 \\ 0.5407 \\ -0.4593 \end{pmatrix} \quad H_2 = I - 2\frac{v_2 v_2^t}{\|v_2\|^2}.$$

$$U_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & & & \\ 0 & & H_2 & \\ 0 & & & \end{pmatrix} \qquad U_2U_1A = \begin{pmatrix} -2.6458 & -4.1576 & 0 & 0.3780 \\ 0 & -0.8452 & 0 & 0.5071 \\ 0 & 0 & 2.1945 & 0.5706 \\ 0 & 0 & 3.6310 & 4.6124 \end{pmatrix}.$$

$$v_3 = x_3 + \|x_3\| \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 6.4371 \\ 3.6310 \end{pmatrix}.$$

$$H_3 = I - 2\frac{v_3 v_3^t}{\|v_3\|^2}.$$

$$U_3 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & & \\ 0 & 0 & & H_3 \end{pmatrix} \qquad \underbrace{U_3U_2U_1}_{Q^t}A = \underbrace{\begin{pmatrix} -2.6458 & -4.1576 & 0 & 0.3780 \\ 0 & -0.8452 & 0 & 0.5071 \\ 0 & 0 & -4.2426 & -4.2426 \\ 0 & 0 & 0 & 1.8974 \end{pmatrix}}_{R}.$$

Exemple 3.5.2 *[7]*

Considérons la matrice

$$A = \begin{pmatrix} 30 & 2 & 3 & 13 \\ 5 & 11 & 10 & 8 \\ 9 & 7 & 6 & 12 \\ 4 & 14 & 15 & 1 \end{pmatrix},$$

et appelons le Programme ***qrbasic*** *pour calculer ses valeurs propres:*

$D = qrbasic(A, 1.e-14, 100)$.

% *cette instruction donne les valeurs propres de la matrice A.*

$D =$

39.3960

17.8208

-9.5022

0.2854.

La méthode converge en 56 *itérations.*

Chapitre 4

Applications

Dans ce chapitre nous proposons d'expliquer l'idée de base proposé par **Larry Page** et **Sergey Brin** pour mettre au point le moteur de recherche Google en 1998.

4.1 Moteur de recherche

Un moteur de recherche est une application web permettant de trouver des ressources à partir d'une requête sous forme des mots. Les ressources peuvent être des pages web, des articles de forums usenent, des images, des vidéos, des fichiers, etc. Certains sites web offrent un moteur de recherche comme principale fonctionnalité, on appelle "moteur de recherche" le site lui-même.

Ce sont des instruments de recherche sur le web sans intervention humaine, ce qui les distingue des annuaires. Ils sont basés sur des "robots", encore appelés "bots" "spiders" "crawlers" ou "agents " qui parcourent les sites à intervalles réguliers et de façon automatique pour découvrir de nouvelles adresses(URL). Ils suivent les liens hypertextes qui relient les pages les unes aux autres, les uns après les autres. Chaque page identifiée est alors indexée dans une base de données, accessible ensuite par les internautes à partir de mots-clés.

4.2 Google

Google (gooooooooogle) est une entreprise américaine de services technologiques fondée en 1998 dans la Silicon Valle, en Californie, par **Larry Page** et **Sergueï Brin**, créateurs du moteur de recherche Google.

Les fondements de l'histoire de l'entreprise Google commencent par la rencontre de deux étudiants de l'université Stanford en 1995. Sergueï Brin alors âgé de vingt-trois ans et Larry Page de vingt-quatre ans sont « pratiquement en désaccord sur tout ». Cela ne les empêche pourtant pas, en janvier 1996, de commencer à travailler sur un nouveau moteur de recherche.

Les fondateurs de Google, alors doctorants, observent la façon dont sont classés les résultats lorsqu'ils effectuent des recherches dans les bases de données scientifiques. Ils constatent que l'exploitation des informations contenues dans la structure hypertextuelle dépend de la nature des liens entre ces documents. Ainsi l'idée d'analyser les liens inter-documents reposait sur l'observation d'une caractéristique de la littérature scientifique et aux modes d'organisation du Sciences Citation Index (SCI), qui consiste à attribuer une valeur à une publication scientifique proportionnellement au nombre de publications qui la cite. Ce principe revient aux travaux de Jon Kleinberg qui avait mis au point pour IBM l'algorithme HITS (Hyperlink –Induced Topic Search) qui prenait en considération l'autorité d'une page en fonction du nombre des liens pointant vers elle. C'est donc en s'inspirant des travaux de Jon Kleinberg que les deux étudiants mettent au point l'algorithme de classement des pages web appelé "Pagerank". Cet algorithme prend en considération les liens pointant vers une page comme un vote pour cette page. Plus une page recevrait de vote plus elle serait considérée comme étant pertinentes et plus le vote de cette page, lorsqu'elle pointerait elle-même vers d'autres pages aurait de la valeur.

4.3 PageRank

Le PageRank ou "PR" est l'algorithme d'analyse des liens concourant au système de classement des pages web utilisé par le moteur de recherche Google. Il mesure quantitativement la popularité d'une page web. Le PageRank n'est qu'un indicateur parmi d'autres dans l'algorithme qui permet de classer les pages du web dans les résultats de recherche de Google. Ce système a été inventé par Larry Page, cofondateur de Google. Ce mot est une marque déposée.

Le PageRank se rencontre habituellement sous la forme d'une note de 0 à 10.

PageRank sous forme d'une note de 0 à 10.

Le PageRank n'est pas le seul critère qui va déterminer votre position dans les résultats des moteur de recherche. Ne dépensez pas toute votre énergie à tenter de l'augmenter. La pertinence est aujourd'hui le paramètre le plus important pour votre classement. Le PageRank n'est désormais utilisé que comme paramètre complémentaire permettant de départager

des page ayant des pertinences comparables pour une recherche particulière.

Illustration du Pagerank.

La notion de PageRank s'appliquent aux pages, et non pas aux sites.

4.4 Algorithme de *Google*

À un moment donné, on peut considérer que le web est une collection de $N \epsilon \mathbb{N}$ pages, avec N très très grand (de l'ordre de 10^{10} en octobre 2005). La plupart de ces pages incluent des liens hypertextes vers d'autres pages. On dit qu'elles *pointent* vers ces autres pages. L'idée de base utilisée par les moteurs de recherche pour classer les pages par ordre de pertinence décroissante consiste à considérer que plus une page est la cible de liens venant d'autres pages, c'est-à-dire plus il y a de pages qui pointent vers elle, plus elle a de chances d'être fiable et intéressante pour l'utilisateur final, et réciproquement. Il s'agit donc de quantifier cette idée, c'est-à-dire d'attribuer un rang numérique ou un score de pertinence à chaque page.

On se donne donc un ordre arbitraire sur l'ensemble des pages que l'on numérote ainsi de $i = 1,...,\mathbb{N}$. La structure de connectivité du web peut alors être représentée par une matrice $C \epsilon \mathbb{N} * \mathbb{N}$ telle que:

$$c_{ij} = \begin{cases} 1 & \text{si la page } j \text{ pointe sur la page } i. \\ 0 & \text{sinon.} \end{cases}$$

Les liens d'une page sur elle-même ne sont pas significatifs, on pose donc $c_{ii} = 0$. On observe que la ligne i contient tous les liens significatifs qui pointent sur la page i, alors que la colonne j contient tous les liens significatifs présents sur la page j.

On souhaite attribuer à chaque page i un score $r_i \in \mathbb{R}_+^*$ de façon à pouvoir classer l'ensemble des pages par score décroissant et présenter à l'utilisateur une liste ainsi classée des pages correspondant à sa requête. L'algorithme "PageRank" part du principe qu'un lien de la page j pointant sur la page i contribue positivement au score de cette dernière, avec une pondération par le score r_j de la page dont est issu le lien une page ayant un score élevé a ainsi plus de poids qu'une n'ayant qu'un score médiocre et par le nombre total de liens présents sur ladite page $N_j = \sum_{k=1}^{N} c_{kj}$. On introduit donc la matrice Q définie par

$$q_{ij} = \begin{cases} \frac{c_{ij}}{N_j} \text{ si } N_j \neq 0 \\ 0 \text{ sinon} \end{cases}$$

La somme des coefficients des colonnes non nulles de Q vaut toujours 1. L'application des principes ci-dessus conduit donc à une équation pour le vecteur $r \in \mathbb{R}^{\mathbb{N}}$ des scores des pages de la forme

$$r_i = \sum_{j=1}^{\mathbb{N}} q_{ij} r_j \text{ c'est à dire } r = Qr.$$

Le problème du classement des pages du web se trouve ainsi ramené à la recherche d'un vecteur propre d'une énorme matrice, associé à la valeur propre 1 .

Il peut arriver que la matrice Q n'admette pas la valeur propre 1 ce qui invalide quelque peu la philosophie originale de l'algorithme. Pour remédier à ce défaut, on considère alors $e = (1, 1,...,1)^t \in \mathbb{R}^{\mathbb{N}}$ et $d \in \mathbb{R}^{\mathbb{N}}$tel que:

$$d_j = \begin{cases} 1 & \text{si } N_j = 0 \\ 0 & \text{sinon} \end{cases}$$

La matrice

$$P = Q + \frac{1}{N} e^t d,$$

est maintenant la transposée d'une matrice stochastique : ses coefficients sont tous positifs et la somme des coefficients de chaque colonne vaut 1(remarquons qu'il s'agit d'une « petite » correction à Q car N est très grand). Du fait que $e^t P = e^t$, on voit que P admet bien la valeur propre 1.

Comme cette valeur propre est en général multiple, on effectue une dernière modification en choisissant un nombre $0 < \alpha < 1$ et en posant

$$A = \alpha P + (1 - \alpha)\frac{1}{N} e^t e.$$

Notons que pour Google $\alpha = 0,85$ est optimal. On pourra admettre qu'une telle matrice admet 1 comme plus grande valeur propre, que cette valeur propre est simple et que l'on peut choisir un vecteur propre correspondant à composantes toutes positives. Finalement, PageRank calcule un tel vecteur propre $r \in \mathbb{R}^{\mathbb{N}}$, normalisé d'une façon ou d'une autre,

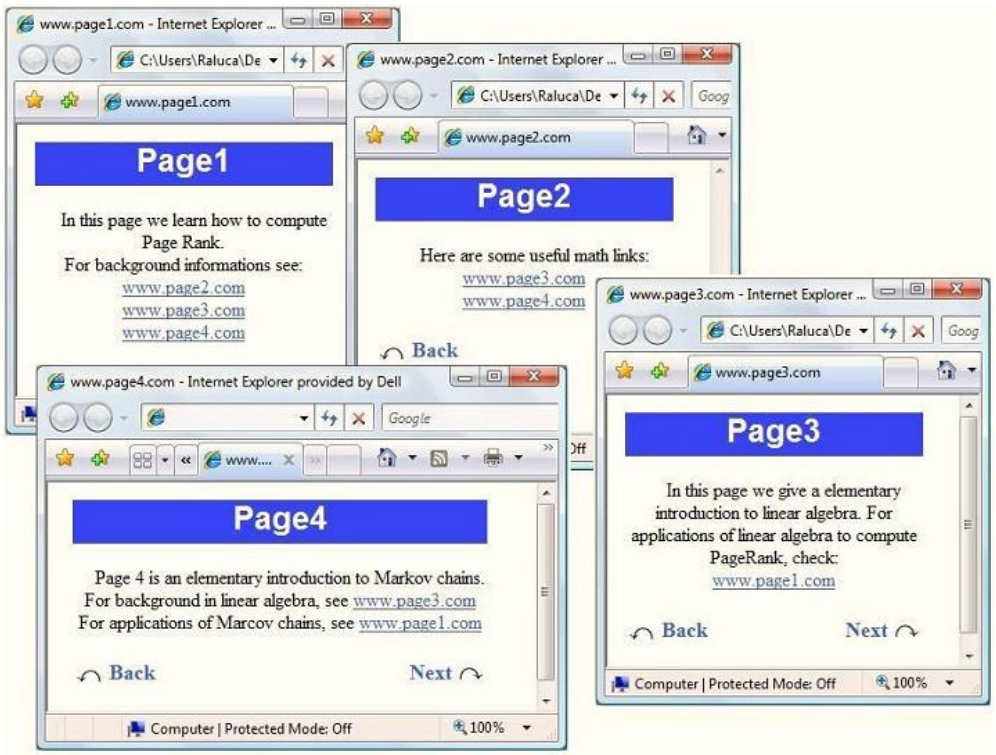
$$r = Ar,$$

dont les N composantes fournissent le classement recherché des pages du web. On sait combien cette stratégie s'est révélée efficace, puisque Google a totalement laminé les moteurs de recherche de première génération, comme Altavista, lesquels ont essentiellement disparu du paysage.

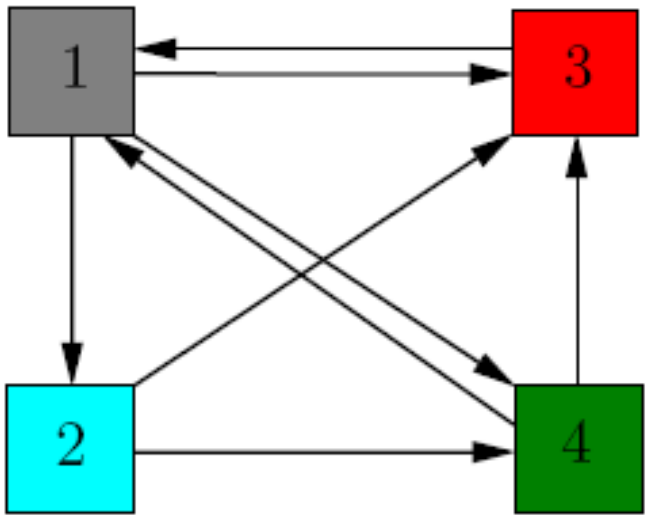
Ce système n'est en général pas facile à résoudre pour $n = 4$, et encore moins lorsqu'il s'agit de tenir compte de l'ensemble des pages web et de l'ensemble des requêtes. Il faut donc résoudre un problème aux valeurs propres avec un grand nombre d'inconnues et pratiquement de manière instantanée.

Pour ce faire, nous commençons par représenter le réseaux web comme un graphe orienté, avec des nœuds représentés par des pages web et des arêtes représentées par des liens entre eux.

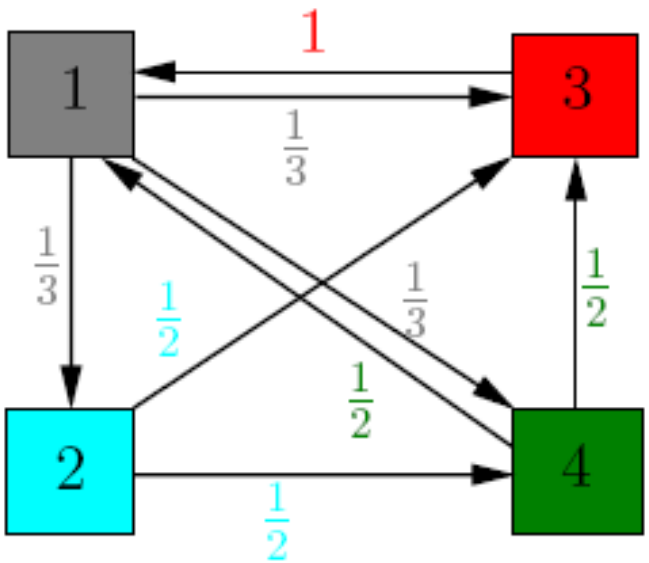
Supposons, par exemple, que nous ayons un résaux de 4 pages de site web: www.page1.com, www.page2.com, www.page3.com, www.page4.com :



Nous "traduisons" l'image en un graphique orienté avec 4 nœuds, un pour chaque site web. Lorsque le site web i fait référence à j, nous ajoutons une flèche orientée dirigé entre le nœud i et le nœud j dans le graphique. Dans le but de calculer leur rang de page, nous ignorons tous les liens de navigation tels que le dos, les boutons suivants, car nous nous soucions seulement des connexions entre les différents sites web. Par exemple, Page1 est liée par toutes les autres pages, de sorte que le nœud 1 du graphique aura des flèches sortantes vers tous les autres nœuds. Page3 a un seul lien avec la page 1, donc le nœud 3 aura un flèche sortant au nœud 1. Après l'analyse de chaque page web, nous obtenons le graphique suivant:



Dans notre modèl, chaque page doit transférer son importance aux pages vers lesquelles elle est liée. Le nœud 1 a 3 flèches sortantes, donc il passera de son importance à chacun des 3 autres nœuds. Le nœud 3 n'a qu'une seul flèche sortante, donc il passera toute son importance au nœud 1. En général, si un nœud a k arêtes sortantes, il transmettra son importance à chacun des nœuds auxquels il est lié. Pour mieux visualiser le processus en affectant des poids à chaque flèche.



Notons A la matrice de transition du graphe $A = \begin{pmatrix} 0 & 0 & 1 & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{pmatrix}$.

Point de vue systèmes dynamiques:

Supposons qu'initialement, l'importance soit uniformément répartie entre les 4 nœuds, chacun recevant $\frac{1}{4}$. Notons v le vecteur de rang initial, ayant des cordonnées égales à $\frac{1}{4}$. Chaque lien entrant augmente l'importance d'une page web, donc à l'étape 1, nous mettons à jour le rang de chaque page en ajoutant à la valeur actuelle l'importance des liens entrants. C'est la même chose que multiplier la matrice A avec v.

A l'étape 1: le nouveau vecteur d'importance est $v_1 = Av$. Nous pouvons itérer le processus, donc à l'étape 2: le vecteur d'importance mis à jour est $v_2 = A(Av) = A^2v$. Les calculs numériques donnent:

$$v = \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix}.$$

$$Av = \begin{pmatrix} 0.37 \\ 0.08 \\ 0.33 \\ 0.20 \end{pmatrix}.$$

$$A^2v = A(Av) = A \begin{pmatrix} 0.37 \\ 0.08 \\ 0.33 \\ 0.20 \end{pmatrix} = \begin{pmatrix} 0.43 \\ 0.12 \\ 0.27 \\ 0.16 \end{pmatrix}.$$

$$A^3v = \begin{pmatrix} 0.35 \\ 0.14 \\ 0.29 \\ 0.20 \end{pmatrix}.$$

$$A^4v = \begin{pmatrix} 0.39 \\ 0.11 \\ 0.29 \\ 0.19 \end{pmatrix}.$$

$$A^5v = \begin{pmatrix} 0.39 \\ 0.13 \\ 0.28 \\ 0.19 \end{pmatrix}.$$

$$A^6v = \begin{pmatrix} 0.38 \\ 0.13 \\ 0.29 \\ 0.19 \end{pmatrix}.$$

$$A^7v = \begin{pmatrix} 0.38 \\ 0.12 \\ 0.29 \\ 0.19 \end{pmatrix}.$$

$$A^8v = \begin{pmatrix} 0.38 \\ 0.12 \\ 0.29 \\ 0.19 \end{pmatrix}.$$

Nous remarquons que les suites d'itérations

$$v, Av, ..., A^kv, ...$$

tendent vers la valeur d'équilibre v^* = Vecteur d'équilibre. Nous appelons cela le vecteur PageRank de notre graphique web.

Le vecteur de Page Rank v^* que nous avons calculé par différentes méthodes, indique que la page 1 est la page la plus pertinente. Cela peut sembler surprenant puisque la page 1 a 2 backlinks, alors que la page 3 a 3 backlinks. Si nous jetons un coup d'oeil au graphique, nous voyons que le nœud 3 n'a qu'un seul front sortant vers le nœud 1, donc il transfère toute son importance au nœud 1. De même, une fois un internaute qui suit uniquement les hyperliens visites page 3, il ne peut que aller à la page 1. Notez également comment le rang

de chaque page n'est pas trivialement juste la somme pondérée des arêtes qui entrent dans le nœud. Intuitivement, à l'étape 1, un nœud reçoit un vote d'importance de ses voisins directs, à l'étape 2 des voisins de ses voisins, et ainsi de suite.

Point de vue de l'algèbre linéaire:

Notons x_1, x_2, x_3 et x_4 l'importance des quatre pages. En analysant la situation à chaque nœud, nous obtenons le système :

$$\begin{cases} x_1 = 1x_3 + \frac{1}{2}x_4 \\ x_2 = \frac{1}{3}x_1 \\ x_3 = \frac{1}{3}x_1 + \frac{1}{2}x_2 + \frac{1}{2}x_4 \\ x_4 = \frac{1}{3}x_1 + \frac{1}{2}x_2 \end{cases}$$

Cela équivaut à demander les solutions des équations $A.\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}$

$$où \quad A = \begin{pmatrix} 0 & 0 & 1 & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{pmatrix}.$$

On montre d'abord que $\lambda = 1$ est une valeur propre pour A^t, avec le vecteur propre correspondant $v = (1, 1, 1, 1)^t$.

$$A^t = \begin{pmatrix} 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 1 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \end{pmatrix}.$$

Ceci est vrai puisque $A^t v = 1v$. Ensuite, à partir du toute matrice A a les mêmes valeurs propres que sa transposée A^t, 1 est une valeur propre pour la matrice A.

Ensuite nous souhaitons trouver un vecteur propre correspondant à cette valeur propre pour A. Soit u un tel vecteur propre. Afin de vous trouver explicitement, nous écrivons l'équation

$$\begin{pmatrix} 0 & 0 & 1 & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{pmatrix} . \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = 1. \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix},$$

tel que $u = (x_1, x_2, x_3, x_4)^t$, avec x_1, x_2, x_3 et x_4 sont tous des nombres réels que nous ne connaissons pas encore. Si nous multiplions terme par terme, nous constatons que:

$$\begin{cases} x_1 = 1x_3 + \frac{1}{2}x_4 \\ x_2 = \frac{1}{3}x_1 \\ x_3 = \frac{1}{3}x_1 + \frac{1}{2}x_2 + \frac{1}{2}x_4 \\ x_4 = \frac{1}{3}x_1 + \frac{1}{2}x_2 \end{cases}.$$

On substituant x_2 dans x_3 et x_4 :

$$\begin{cases} x_1 = 1x_3 + \frac{1}{2}x_4 \\ x_2 = \frac{1}{3}x_1 \\ x_3 = \frac{1}{3}x_1 + \frac{1}{2}\frac{1}{3}x_1 + \frac{1}{2}x_4 = \frac{1}{2}x_1 + \frac{1}{2}x_4 \\ x_4 = \frac{1}{3}x_1 + \frac{1}{2}\frac{1}{3}x_1 = \frac{1}{2}x_1 \end{cases}.$$

Remplacons $x_4 = \frac{1}{2}x_1$ dans la troixème relation on obtient: $x_3 = \frac{3}{4}x_1$ et donc le vecteur u est sous la forme :

$$u = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} x_1 \\ \frac{1}{3}x_1 \\ \frac{3}{4}x_1 \\ \frac{1}{2}x_1 \end{pmatrix} = x_1 \begin{pmatrix} 1 \\ \frac{1}{3} \\ \frac{3}{4} \\ \frac{1}{2} \end{pmatrix} = \frac{x_1}{12} \begin{pmatrix} 12 \\ 4 \\ 9 \\ 6 \end{pmatrix}$$

Comme x_1 est juste un nombre réel (d'où un scalaire) nous pouvons prendre $x_1 = 12$ et nous venons de prouver que le vecteur dont les entrées sont 12, 4, 9 et 6 (de haut en bas) est un vecteur propre pour A.

Conclusion

Le PageRank n'est pas le seul critère qui va déterminer votre position dans les résultats des moteurs de recherche. Ne dépensez pas toute votre énergie à tenter de l'augmenter. La pertinence est aujourd'hui le paramètre le plus important pour votre classement. Le PageRank n'est désormais utilisé que comme paramètre complémentaire permettant de départager des pages ayant des pertinences comparables pour une recherche particulière.

Bibliographie

[1] S. Brin, L. Page: "*The Anatomy of a Large-Scale Hypertextual Web Search Engine*", Stanford University, 1998.

[2] P.G.Ciarlet: "*Introduction à l'analyse numérique matricielle et à l'optimisation*". Masson.

[3] W.Ford: "*Numerical Linear Algebra with Applications", 1 ed, Elsevier, 2015.*

[4] G.H.Golub,C.F Van Loan: "*Matrix computations*" John Hopkins University press, Baltimore, 2013.

[5] D.Lay: "*Linear Algebra and its applications", 4th ed.*

[6] B.N. Parlett (1980): "*The Symmetric Eigenvalue Problem*". Prentice-Hall, Englewood Cliffs, NJ.

[7] A.Quarteroni, P.Gervasio ,F.Saleri: "*Calcul Scientifique*", cours, exercices corrigés et illustration en MATLAB et Octave, Springer-verlag Italia 2010.

[8] A.Quarteroni, R.Sacco ,F.Saleri: "*Méthodes numériques pour le calcul scientifique*", Springer, 2007.

[9] Y.Saad: "*Numerical Methods for large eigenvalue problem*", Revised ed,SIAM,Philadelphia 2011.

[10] J.H.Wilkinson: "*The Algebraic Eigenvalue Problem*", Oxford University Press, New York, 1965.

[11] A.Yger, Jacques-Arthur weil: "*L3 Mathématiques appliquées*", Pearson, 2009.

[12] Wikipedia: PageRank, en.wikipedia.org/wiki/PageRank et fr.wikipedia.org/wiki/PageRank.

[13] https://www.unige.ch/~wanner/teaching/Numi/Numi5.

[14] www.math.univ-montp2.fr/~nicoud/Cours/CSIChap3.

[15] www-pequan.lip6.fr/.../polycopies/cours5-puissanceiteree.

[16] math.univ-lyon1.fr/.../numalg/algebre_notes_de_cours_6.

[17] www.irisa.fr/sage/bernard/publis/AF1224-FINAL.

4

Annexe

Programme powerm: Méthode de la puissance

```
function[lambda,x,iter,relres]=powerm(A,z0,tol,nmax)
%POWERM Méthode de la puissance
%[LAMBDA,X,ITER,RELRES]=POWERM(A,Z0,TOL,NMAX)calcule la valeur propre
%LAMBDA de plus grand module de la matrice A et un vecteur propre correspondant
%X de la norme un. TOL est la tolérence de la méthode.
%NMAX est le nombre maximum d'itérations. Z0 est la donnée initiale.
% ITER est l'itération à laquelle la solution X a été calculée.
q=z0/norm(z0); q2=q;
relres=tol+1; iter=0; z=A*q;
while relres(end)>=tol & iter<=nmax
q=z/norm(z); z=A*q;
lambda=q'*z; x=q;
z2=q2'*A; q2=z2/norm(z2); q2=q2';
y1=q2; costheta=abs(y1'*x);
if costheta >= 5e-2
iter=iter+1;
temp=norm(z-lambda*q)/costheta;
relres=[relres; temp];
else
```

```
fprintf('Valeur propre multiple'); break;
end
x
iter
relres
end
return
```

Programme invpower: Méthode de la puissance inverse

```
function [sigma,x,iter,relres]=invpower(A,z0,mu,tol,nmax)
%INVPOWER Méthode de la puissance inverse
% [SIGMA,X,ITER,RELRES]=INVPOWER(A,Z0,MU,TOL,NMAX) calcule la
% valeur propre LAMBDA de plus petit module de la matrice A et un vecteur
% propre correspondant X de norme un. TOL est la tolérance de la méthode.
% NMAX est le nombre maximum d'itérations. z0 est la donnée initiale.
% MU est le shift. ITER est l'itération à laquelle la solution X a été calculée.
M=A-mu*eye(size(A)); [L,U,P]=lu(M);
q=z0/norm(z0); q2=q'; sigma=[];
relres=tol+1; iter=0;
while relres(end)>=tol & iter<=nmax
iter=iter+1;
b=P*q;
y=L\b; z=U\y;
q=z/norm(z); z=A*q; sigma=q'*z;
b=q2'; y=U'\b; w=L'\y;
q2=w'*P; q2=q2/norm(q2); costheta=abs(q2*q);
if costheta>=5e-2
temp=norm(z-sigma*q)/costheta; relres=[relres,temp];
else
fprintf('Valeur propre multiple'); break;
end
x=q;
end
return
```

Programme gershcicles: disques de Gershgorin

```
function gershcircles(A)
% GERSHCIRCLES(A) trace les disques de Gershgorin
% pour la matrice carrée A et sa transposée.
n = size(A);
if n(1) ~= n(2)
error('Matrices carrées seulement');
else
n=n(1); circler=zeros(n ,201); circlec=circler;
end
center = diag(A);
radiic = sum(abs(A-diag(center )));
radiir = sum(abs(A'-diag(center )));
one = ones (1,201); cosisin = exp(i*[0:pi/100:2* pi]);
figure (1); title('Disques des lignes ');
xlabel('Re'); ylabel('Im');
figure (2); title('Disques des colonnes ');
xlabel('Re'); ylabel('Im');
for k = 1:n
circlec(k,:) = center(k)*one + radiic(k)*cosisin ;
figure (1);
patch(real(circler (k,:)) ,imag(circler (k,:)) ,'red');
hold on
plot(real(circler(k,:)) ,imag(circler(k,:)) ,'k-' ,...
real(center(k)),imag(center(k)),'kx');
figure (2);
patch(real(circlec (k,:)) ,imag(circlec (k,:)) ,'green');
hold on
plot(real(circlec(k,:)) ,imag(circlec(k,:)) ,'k-' ,...
real(center(k)),imag(center(k)),'kx');
```

```
end
for k = 1:n
figure (1);
plot(real(circler(k,:)) ,imag(circler(k,:)) ,'k-' ,...
real(center(k)),imag(center(k)),'kx');
figure (2);
plot(real(circlec(k,:)) ,imag(circlec(k,:)) ,'k-' ,...
real(center(k)),imag(center(k)),'kx');
end
figure (1); axis image; hold off;
figure (2); axis image; hold off
return
```

Programme qrbasic: Méthode de QR

```
function D=qrbasic (A,tol ,nmax)
%QRBASIC calcule les valeurs propres de la matrice A.
% D=QRBASIC(A,TOL ,NMAX) calcule par itérations QR
% toutes les valeurs propres de A avec une tolérance
% TOL en NMAX itérations au maximum. La convergence
% de cette méthode n'est pas toujours garantie .
[n,m]=size(A);
if n ~= m, error('Matrices carrées seulement'); end
T = A; niter = 0; test = norm(tril(A,-1),inf);
while niter <= nmax & test >= tol
[Q,R]=qr(T); T = R*Q;
niter = niter + 1;
test = norm(tril(T,-1),inf);
end
if niter > nmax
warning(['La méthode ne converge pas dans le '
'nombre d"itérations maximum voulu\n']);
else
fprintf(['La methode converge en ' ...
'%i itérations\n'],niter);
end
D = diag(T);
return
```

ملخص:

في هذا العمل حاولنا تقديم بعض الطرق لحساب القيم التقريبية للقيم الذاتية لمصفوفة، وعلى سبيل المثال نذكر طريقة الأس والأس العكسي.
في البداية قدمنا تذكير لبعض المفاهيم الأساسية: قيمة ذاتية، شعاع ذاتي، نظام خطي، كثير حدود مميز، ومن ثم قمنا بتقديم بعض طرق حساب القيم الذاتية وشروط تقاربها بالإضافة إلى أمثلة توضيحية.
وأخيرا كتطبيق، حاولنا شرح الكيفية التي يستخدمها غوغل لترتيب الصفحات .

كلمات مفتاحية :

القيم الذاتية، الأشعة الذاتية، طيف، تقارب، طريقة الأس والأس العكسي، جكوبي، طريقة QR، طرق ترددية.

Résumé:

Dans ce mémoire, on a essayé de présenter quelques méthodes d'approximation des valeurs propres d'une matrice telles que méthode de la puissance et puissance inverse.
Au début, on a donné des rappelles et des concepts de base: valeur propre, système linéaire et polynôme caractéristique. Ensuite, on a présenté un aperçu sur les méthodes de calcul les valeurs propres avec des conditions de convergences en illustrant par des exemples.
Enfin, on a appliqué cette théorie sur le moteur de recherche Google pour le classement des pages web.

Mots clés:

Valeurs propres, vecteurs propres, spectre, convergence, méthode de puissance et puissance inverse, Jacobi, QR, méthode itérative…

Abstract:

In this memory, we try to present a basic approximate methods for compute eigenvalues of matrice such that power and inverse method.
At the beginning, we gave overview and basic concepts: eigenvalue, linear system and characteristic polynomial. Next, we gave an overview on methods for compute eigenvalues with convergence conditions and examples.
As is known, we gave an application to Google how classified pages web.

Key words:

Eigenvalues, eigenvectors, spectrum, convergence, power and power inverse methods, Jacobi, QR, iterative method…