ÉCOLE INTERNATIONALE DES SCIENCES
DU TRAITEMENT DE L'INFORMATION

MATHÉMATIQUES APPLIQUÉES

Page Rank

Bara DIOUKHANE
Quentin DUCASSE

Semestre 2

Promotion 2022

2019

Table des matières

1 Présentation du Pagerank

Le pagerank est un critère de pertinence (représenté par un chiffre compris entre 0 et 10) qui analyse la popularité d'une page Web. Le pagerank permet donc de calculer la qualité et la quantité des liens pointant vers chaque page Web permettant ainsi par la suite de classer les pages Web dans les résultats de recherche Google.
Pour calculer le pagerank d'une page Web, on prend en compte 3 principaux facteurs :

- Le nombre de liens pointant vers la page (liens entrants et sortants)
- La qualité des pages (liens entrants)
- Le pagerank de chaque lien

Il existe deux principaux types de pagerank qui sont le pagerank amorti et non amorti.

Lorqu'une page Web possède de nombreux liens qui se redirigent vers une même page principale, le pagerank diminue des liens secondaires selon un facteur p appelé facteur d'amortissement. On appelle ce pagerank le pagerank amorti. Dans le cas contraire, c'est à dire si on ne prend pas en compte ce facteur d'amortissement, on parle alors de pagerank non amorti.

2 Calcul direct du pagerank (non amorti)

2.1 Méthode algèbrique

Suppsosons que l'on souhaite calculer le pagerank de notre site internet composé de 5 pages Web. On représente notre site internet par un schéma sur lequel on peut visualiser tous les liens entre chaque page Web du site.

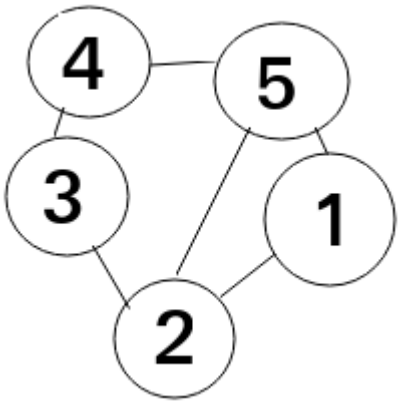


FIGURE 1 – Architecture des différentes pages web du site

On note r_1, r_2, r_3, r_4 et r_5 les pageranks des pages respectives 1, 2 ,3, 4 et 5. On remarque sur ce schéma que la page 1 est reliée par la page 2 et par la page 5. De plus la page 2 est reliée par 3 autres pages qui sont les pages 1, 3 et 5. La page 5 est reliée elle aussi par 3 pages qui sont les pages 1, 4 et 2.
Donc $r_1 = \frac{1}{3}r_2 + \frac{1}{3}r_5$
On suit le même raisonnement pour les cinq pages Web du site et on obtient le système suivant :

$$\begin{cases} r_1 = & \frac{1}{3}r_2 + \frac{1}{3}r_5 \\ r_2 = & \frac{1}{2}r_1 + \frac{1}{2}r_3 + \frac{1}{3}r_5 \\ r_3 = & \frac{1}{3}r_2 + \frac{1}{2}r_4 \\ r_4 = & \frac{1}{2}r_3 + \frac{1}{3}r_5 \\ r_5 = & \frac{1}{2}r_1 + \frac{1}{3}r_2 + \frac{1}{2}r_4 \end{cases}$$

On pose : $r = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \end{pmatrix}$

On identifie facilement une matrice $M = \begin{pmatrix} 0 & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & \frac{1}{3} \\ 0 & \frac{1}{3} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & 0 & \frac{1}{2} & 0 \end{pmatrix}$

Donc :

$$\begin{cases} r_1 = & \frac{1}{3}r_2 + \frac{1}{3}r_5 \\ r_2 = & \frac{1}{2}r_1 + \frac{1}{2}r_3 + \frac{1}{3}r_5 \\ r_3 = & \frac{1}{3}r_2 + \frac{1}{2}r_4 \\ r_4 = & \frac{1}{2}r_3 + \frac{1}{3}r_5 \\ r_5 = & \frac{1}{2}r_1 + \frac{1}{3}r_2 + \frac{1}{2}r_4 \end{cases}$$

$\iff r = Mr$

$$\iff \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \end{pmatrix} = \begin{pmatrix} 0 & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & \frac{1}{3} \\ 0 & \frac{1}{3} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & 0 & \frac{1}{2} & 0 \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \end{pmatrix}$$

On sait que la somme de toute les probabilités est égale a 1. Donc :
$r_1 + r_2 + r_3 + r_4 + r_5 = 1$

On cherche donc l'unique vecteur r tel que $r = Mr$ et tel que la somme des composantes de r est égale a 1.

$$
\begin{aligned}
r = Mr \iff & \quad r(M - Id_5) = 0_E \\
\iff & \quad (M - Id_5) = 0_E \text{ (car r est non nul)} \\
\iff & \quad r \in Ker(M - Id_5) = 0_E \\
\iff & \quad (M - Id_5) \times r = 0_E \\
\iff & \quad \left(\begin{pmatrix} 0 & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & \frac{1}{3} \\ 0 & \frac{1}{3} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & 0 & \frac{1}{2} & 0 \end{pmatrix} - \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \right) \times \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \\
\iff & \quad \begin{pmatrix} -1 & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ \frac{1}{2} & -1 & \frac{1}{2} & 0 & \frac{1}{3} \\ 0 & \frac{1}{3} & -1 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & -1 & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & 0 & \frac{1}{2} & -1 \end{pmatrix} \times \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \\
\iff & \quad \begin{cases} -r_1 + \frac{1}{3}r_2 + \frac{1}{3}r_5 = 0 \\ \frac{1}{2}r_1 - r_2 + \frac{1}{2}r_3 + \frac{1}{3}r_5 = 0 \\ \frac{1}{3}r_2 - r_3 + \frac{1}{2}r_4 = 0 \\ \frac{1}{2}r_3 - r_4 + \frac{1}{3}r_5 = 0 \\ \frac{1}{2}r_1 + \frac{1}{3}r_2 + \frac{1}{2}r_4 - r_5 = 0 \end{cases} \\
\iff & \quad r = \begin{pmatrix} \frac{2}{3}r_1 \\ r_2 \\ \frac{2}{3}r_3 \\ \frac{2}{3}r_4 \\ r_5 \end{pmatrix} \text{ (Après calcul)} \\
\iff & \quad r \in vect(\tfrac{2}{3}, 1, \tfrac{2}{3}, \tfrac{2}{3}, 1)
\end{aligned}
$$

La somme des composantes de r est égale à 1 donc on divise chaque composante de vect par la somme des composantes :

$$r = \begin{pmatrix} \frac{\frac{2}{3}}{\sum_{i=1}^{5} r_i} \\ \frac{1}{\sum_{i=1}^{5} r_i} \\ \frac{\frac{2}{3}}{\sum_{i=1}^{5} r_i} \\ \frac{\frac{2}{3}}{\sum_{i=1}^{5} r_i} \\ \frac{1}{\sum_{i=1}^{5} r_i} \end{pmatrix} = \begin{pmatrix} \frac{2}{3\times4} \\ \frac{1}{4} \\ \frac{2}{3\times4} \\ \frac{2}{3\times4} \\ \frac{1}{4} \end{pmatrix} = \begin{pmatrix} \frac{1}{6} \\ \frac{1}{4} \\ \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{4} \end{pmatrix}$$

Donc nous avons trouvé les pageranks r_1, r_1, r_1, r_1, r_1 des pages Web 1, 2, 3, 4 et 5 de notre site Web.
- La page 1 a un pagerank égale à 1/6
- La page 2 a un pagerank égale à 1/4
- La page 3 a un pagerank égale à 1/6
- La page 4 a un pagerank égale à 1/6
- La page 5 a un pagerank égale à 1/4

2.2 Méthode numérique

Présentation de la méthode

La méthode numérique permet de calculer les pageranks d'un nombre quelconque de pages internet reliées à un nombre quelconque de liens hypertextes variés. L'algorithme ci-dessous calcule le couple (P,D) qui constitue la diagonalisation de la matrice M avec P une matrice de passage et D la matrice diagonale calculée tel que $M = PDP^{-1}$. (Si M est diagonalisable). On sait que la matrice D diagonale est une matrice diagonale ayant pour diagonales les valeurs propres de la matrice M.
Pour rappel, une valeur propre λ de M est un scalaire tel que det$(M - \lambda id_E) = 0$. Comme on a démontré précédemment que det$(M - 1 \times id_E) = 0$, il suffit de rechercher le vecteur propre associé à la valeur propre 1. Pour cela, on parcourt chaque vecteur colonne de la matrice P jusqu'à trouver le vecteur correspondant à la valeur propre 1. Ce vecteur est appelé sous-espace propre de la valeur propre 1 et est noté E_1. Il correspond à $vect$ calculé précédemment.
Dans notre exemple $E_1 = vect(\frac{2}{3}, 1, \frac{2}{3}, \frac{2}{3}, 1)$. Les étapes qui suivent sont exactement les mêmes que celles décrites précédemment.

On retrouve donc $r = (\frac{1}{6}, \frac{1}{4}, \frac{1}{6}, \frac{1}{6}, \frac{1}{4})$

Choix algorithmiques

Voici le code pour calculer le vecteur r contenant tous les pageranks de chaque page du site.
On utilise la fonction *spec* implémentée dans scilab qui renvoie donc la matrice de passage P et la matrice diagonale D tel que $M = PDP^{-1}$.
La fonction prend en paramètre d'entrée la matrice M et renvoie la pagerank de chaque page.

```
function [r]= pagerank (M)
  [m,n]=size(M);
  I=zeros(m,n);
  s=0;
  pos=0;
 [P,D]=spec(M);

 for i=1:n
     if D(i,i)==1
         pos=i;
     end
 end

 for i=1:n
     for j=1:n

      r(i)=P(i,pos)
    end
 end

for i=1 : n
    s=s+r(i);
end

for i=1 : n
    r(i)=r(i)/s
end
endfunction
```

FIGURE 2 – Algorithme permettant de calculer le vecteur r pour un site d'architecture quelconque de manière directe

Exemple illustant l'algorithme

Soit $M = \begin{pmatrix} 0 & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & \frac{1}{3} \\ 0 & \frac{1}{3} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & 0 & \frac{1}{2} & 0 \end{pmatrix}$

La fonction spec renvoie donc les résultats suivants :

```
--> [P,D]=spec(M)
 D  =

   1.   0.          0.    0.          0.
   0.   0.3333333   0.    0.          0.
   0.   0.         -0.5   0.          0.
   0.   0.          0.   -0.8333333   0.
   0.   0.          0.    0.          0.

 P  =

   0.3651484   0.5345225   0.6666667   0.    0.
   0.5477226   0.2672612  -0.5        -0.5  -0.5883484
   0.3651484  -0.5345225   0.1666667   0.5  -0.3922323
   0.3651484  -0.5345225   0.1666667  -0.5   0.3922323
   0.5477226   0.2672612  -0.5         0.5   0.5883484
```

On sélectionne le vecteur propre associé à la valeur propre 1. Le vecteur propre est encadré en rouge ci-dessous :

```
--> [P,D]=spec(M)
 D  =

   1.   0.          0.    0.          0.
   0.   0.3333333   0.    0.          0.
   0.   0.         -0.5   0.          0.
   0.   0.          0.   -0.8333333   0.
   0.   0.          0.    0.          0.

 P  =

   0.3651484   0.5345225   0.6666667   0.    0.
   0.5477226   0.2672612  -0.5        -0.5  -0.5883484
   0.3651484  -0.5345225   0.1666667   0.5  -0.3922323
   0.3651484  -0.5345225   0.1666667  -0.5   0.3922323
   0.5477226   0.2672612  -0.5         0.5   0.5883484
```

Ensuite, il suffit de diviser chaque composante du vecteur par la somme des composantes du vecteur.

On obtient le résultat suivant :

```
r  =

    0.1666667
    0.25
    0.1666667
    0.1666667
    0.25
```

On retouve donc bien $r = (\frac{1}{6}, \frac{1}{4}, \frac{1}{6}, \frac{1}{6}, \frac{1}{4})$ caculé précédemment par la méthode algébrique

3 Calcul par marche aléatoire du pagerank

La détermination des pageranks, par la méthode précédente, suppose de calculer un vecteur propre associé à une matrice dont la taille est égale au nombre de pages internet. Ceci suppose de manipuler des matrices dont la taille est de plusieurs milliards au carré, ce qui n'est pas faisable même par les plus puissants ordinateurs. La technique de la marche aléatoire, présentée dans la suite, va permettre d'approcher la valeur des pageranks avec beaucoup moins de calculs.

3.1 Cas du pagerank non amorti

Méthode par le calcul

On considère un utilisateur se déplaçant de page en page en choisissant la page suivante de manière aléatoire et équiprobable dans le graphe de la figure 1. Pour tout entier naturel n, on note X_n, la variable aléatoire réelle, égale au numéro de la page consultée par l'utilisateur après n déplacements.

Soit $n \in \mathbf{N}$
Le nombre de pages Web distinctes dans notre exemple est de 5 donc le numéro des pages Web varie entre 1 et 5.
D'où $X_n(\Omega) = [\![1, 5]\!]$

Soit $i \in [\![1, 5]\!]$
D'après la formule des probabilités totales au système complet d'évenements associé à X_n, on obtient le système suivant :

$$\begin{cases} P(X_{n+1} = 1) = & \frac{1}{3}P(X_n = 2) + \frac{1}{3}P(X_n = 5) \\ P(X_{n+1} = 2) = & \frac{1}{2}P(X_n = 1) + \frac{1}{2}P(X_n = 3) + \frac{1}{3}P(X_n = 5) \\ P(X_{n+1} = 3) = & \frac{1}{3}P(X_n = 2) + \frac{1}{2}P(X_n = 4) \\ P(X_{n+1} = 4) = & \frac{1}{2}P(X_n = 3) + \frac{1}{3}P(X_n = 5) \\ P(X_{n+1} = 5) = & \frac{1}{2}P(X_n = 1) + \frac{1}{3}P(X_n = 2) + \frac{1}{2}P(X_n = 4) \end{cases}$$

Comme chaque probabilité est équiprobable, alors

$$\begin{pmatrix} P(X_{n+1}=1) \\ P(X_{n+1}=2) \\ P(X_{n+1}=3) \\ P(X_{n+1}=4) \\ P(X_{n+1}=5) \end{pmatrix} = \begin{pmatrix} 0 & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & \frac{1}{3} \\ 0 & \frac{1}{3} & 0 & \frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & 0 & \frac{1}{2} & 0 \end{pmatrix} \times \begin{pmatrix} P(X_n=1) \\ P(X_n=2) \\ P(X_n=3) \\ P(X_n=4) \\ P(X_n=5) \end{pmatrix}$$

Soit $i \in [\![1,5]\!]$

$\forall j \in [\![1,5]\!]$
$P(X_{n+1}=i)_{1\leq i\leq 5} = M \times P(X_n=j)_{1\leq j\leq 5}$
On note $\Pi_n = P(X_n=j)_{1\leq j\leq 5} = P(X_n=i)_{1\leq i\leq 5}$
Donc $\Pi_{n+1} = M\Pi_n$ où M est la matrice donné ci dessus

Pour le vecteur initial, on peut supposer que l'on part d'une page k, auquel cas le vecteur Π_0 aura une k-ième composante égale à 1 et les autres composantes nulles.

On pose $\Pi_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$

Donc $\Pi_{n+1} = M \times \Pi_n$ $\Pi_1 = M\Pi_0$
$\Pi_2 = M\Pi_1 = M(M\Pi_0) = M^2\Pi_0$
$\Pi_3 = M\Pi_2 = M(M^2\Pi_0) = M^3\Pi_0$

De manière générale, on remarque que $\Pi_n = M^n\Pi_0$

On peut montrer que la suite de matrices $(M^n)_{n\in\mathbf{N}}$ converge vers la matrice d'une projection sur $E_1(M) = SEP(M,1)$, sous espace propre associé à la valeur 1 pour M.
Pour cela, nous allons calculer M^2, M^3, M^4, etc pour pouvoir approcher une limite finie de chaque valeur de la matrice.

Après calcul :

$$M^2 = \begin{pmatrix} 0.3333333 & 0.1111111 & 0.1666667 & 0.1666667 & 0.1111111 \\ 0.1666667 & 0.4444444 & 0. & 0.4166667 & 0.1666667 \\ 0.1666667 & 0. & 0.4166667 & 0. & 0.2777778 \\ 0.1666667 & 0.2777778 & 0. & 0.4166667 & 0. \\ 0.1666667 & 0.1666667 & 0.4166667 & 0. & 0.4444444 \end{pmatrix}$$

$$M^3 = \begin{pmatrix} 0.1111111 & 0.2037037 & 0.1388889 & 0.1388889 & 0.2037037 \\ 0.3055556 & 0.1111111 & 0.4305556 & 0.0833333 & 0.3425926 \\ 0.1388889 & 0.287037 & 0. & 0.3472222 & 0.0555556 \\ 0.1388889 & 0.0555556 & 0.3472222 & 0. & 0.287037 \\ 0.3055556 & 0.3425926 & 0.0833333 & 0.4305556 & 0.1111111 \end{pmatrix}$$

$$M^4 = \begin{pmatrix} 0.2037037 & 0.1512346 & 0.1712963 & 0.1712963 & 0.1512346 \\ 0.2268519 & 0.3595679 & 0.0972222 & 0.3865741 & 0.1666667 \\ 0.1712963 & 0.0648148 & 0.3171296 & 0.0277778 & 0.257716 \\ 0.1712963 & 0.257716 & 0.0277778 & 0.3171296 & 0.0648148 \\ 0.2268519 & 0.1666667 & 0.3865741 & 0.0972222 & 0.3595679 \end{pmatrix}$$

$$M^5 = \begin{pmatrix} 0.1512346 & 0.1754115 & 0.1612654 & 0.1612654 & 0.1754115 \\ 0.2631173 & 0.1635802 & 0.373071 & 0.1319444 & 0.3243313 \\ 0.1612654 & 0.248714 & 0.0462963 & 0.2874228 & 0.087963 \\ 0.1612654 & 0.087963 & 0.2874228 & 0.0462963 & 0.248714 \\ 0.2631173 & 0.3243313 & 0.1319444 & 0.373071 & 0.1635802 \end{pmatrix}$$

$$M^6 = \begin{pmatrix} 0.1512346 & 0.1754115 & 0.1612654 & 0.1612654 & 0.1754115 \\ 0.2631173 & 0.1635802 & 0.373071 & 0.1319444 & 0.3243313 \\ 0.1612654 & 0.248714 & 0.0462963 & 0.2874228 & 0.087963 \\ 0.1612654 & 0.087963 & 0.2874228 & 0.0462963 & 0.248714 \\ 0.2631173 & 0.3243313 & 0.1319444 & 0.373071 & 0.1635802 \end{pmatrix}$$

On constate que le premier vecteur colonne de M^6 est environ égal r le vecteur calculé ci-dessus dans la partie précédente

On verifie avec n très grand. On pose n=300

$$M^{300} = \begin{pmatrix} 0.1666667 & 0.1666667 & 0.1666667 & 0.1666667 & 0.1666667 \\ 0.25 & 0.2508425 & 0.2487362 & 0.2512638 & 0.2491575 \\ 0.1666667 & 0.1658241 & 0.1679305 & 0.1654029 & 0.1675092 \\ 0.1666667 & 0.1675092 & 0.1654029 & 0.1679305 & 0.1658241 \\ 0.25 & 0.2491575 & 0.2512638 & 0.2487362 & 0.2508425 \end{pmatrix}$$

On remarque effectivement que la matrice $(M^n)_{n\in\mathbf{N}}$ tend vers une projection sur $E_1(M) = SEP(M, 1)$

Comme $\Pi_n = M^n\Pi_0$, $(\Pi_n)_{n\in\mathbf{N}}$ converge vers $P = \begin{pmatrix} \frac{1}{6} \\ \frac{1}{4} \\ \frac{1}{6} \\ \frac{1}{6} \\ \frac{1}{4} \end{pmatrix} = (0.167, 0.25, 0.167, 0.167, 0.25)$

On retrouve le vecteur calculé précédemment par la méthode directe.

Choix algorithmiques

La fonction $markov$ prend en paramètre la matrice de transition M, le nombre de déplacemnts n et le vecteur intial P0. Cette fonction permet de calculer le vecteur $P = \Pi_n$ pour ainsi connaître chaque pagerank de chaque page Web du site.
La fonction applique la formule $\Pi_n = M\Pi_{n-1}$

```
function [P] =markov (M,n,P0)
    [m,m]=size(M);
    P=P0;
    for i=1:n
        P=M*P;

    end

 P

endfunction
```

On obtient le vecteur P suivant pour n=5 :

On remarque que seules 3 valeurs sont distinctes.

```
P  =

  0.1512346
  0.2631173
  0.1612654
  0.1612654
  0.2631173
```

La fonction $markovPlot$ de même paramètre que la fonction $plot$ génère un graphe contenant l'évolution des probabilités de chaque page au cours des n déplacements pour une architecture quelconque.

L'algorithme est le suivant :

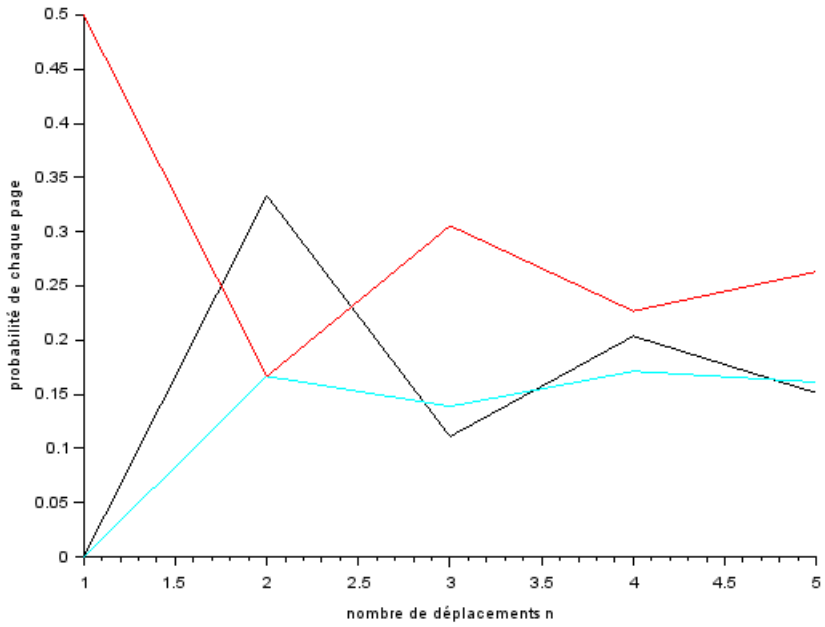
```
function markov_plot(M,n,P0)
 [m,m]=size(M);
    P=P0;

 E=zeros(m,n);
 E(:,1)=P

    for i=1:n
        P=M*P
        E(:,i)=P
    end
scf()
x=1:1:n

for j=1:m
        plot2d(x',(E(j,:)'),j )
    end
endfunction
```

Dans l'exemple de notre matrice M, on prend n=5. On obtient le graphe ci-dessous :



On remarque sur le graphe que seulement trois probabilités distinctes évoluent. Les résultats concordent avec la remarque précédente. En effet, les deux courbes représentant l'évolution des deux probabilités restantes sont confondues avec les trois courbes du graphe : on observe que la courbe bleue et la courbe noire tendent vers 0.16 tandis que la courbe rouge tend vers 0.25. On retrouve les pageranks correspondant aux résultats précédents.

3.2 Cas du pagerank amorti

Méthode algébrique

La méthode présentée jusqu'ici suppose que l'architecture étudiée est en vase clos, autrement dit que chaque page est atteignable à partir d'une autre en un nombre fini d'étapes.
On dit alors que le graphe est connexe. Il restera alors à traiter le cas des pages qui n'ont pas de lien hypertexte entrant ou sortant, ainsi que celui des groupes de pages qui ont des liens entre elles mais pas avec le reste.
Dès la soumission de l'algorithme original, Larry PAGE propose d'introduire un facteur d'amortissement qui est la probabilité p qu'à chaque déplacement, l'utilisateur choisisse de rester dans l'architecture de départ.
Ce qui veut dire surtout qu'il y a une probabilité (1− p) qu'il se téléporte sur n'importe qu'elle autre page du Web. Cette téléportation aléatoire se fait de manière équiprobable entre toutes les pages. En ajoutant le facteur d'amortissement à l'équation $\Pi_{n+1} = M\Pi_n$:

$$\Pi_{n+1} = P(X_{n+1} = i)_{1\leq i\leq 5} = M(P(X_n = j)_{1\leq j\leq 5}) = \underbrace{p(M \times \Pi_n)}_{\text{architecture de départ}} + \underbrace{(1-p)M_{5,5}(1/5) \times \Pi_n}_{\text{téléportation aléatoire}}$$

On peut remarquer qu'il s'agit bien d'une chaine de Markov homogène car l'expression de Π_{n+1} sachant Π_n ne dépend pas de n.

En réutilisant l'expression obtenue dans le cas du pagerank non amorti calculé précédemment, on a $\Pi_{n+1} = M\Pi_n$.
En factorisant l'expression Π_{n+1} trouvée un peu plus haut,
$\Pi_{n+1} = (pM + (1-p)M_{5,5}(1/5)) \times \Pi_n$
$\iff \Pi_{n+1} = G(p)\Pi_n$ avec $G(p) = (pM + (1-p) \times M_{5,5}(1/5)) = (pM + \frac{1-p}{5} \times (M_{5,5}(1)))$

En généralisant, soit une matrice de taille (m,n), on a donc pour un procédé quelconque l'égalité suivante :
$\Pi_{n+1} = G(p)\Pi_n$
avec $G(p) = (pM + \frac{1-p}{m} \times (M_{m,n}(1)))$

Choix algorithmiques

La fonction Googlematrix prend en paramètre la matrice de transition et le facteur d'amortissement p. Dans l'article original, Sergey BRIN et Larry PAGE proposent un facteur d'amortissement égale à 0,85. Donc en cas de non-précision du paramètre p ou de valeur illogique entrée par l'utilisateur, on prendra cet valeur en compte.
On applique donc l'expression de G(p) que l'on a obtenue précédemment, tout en gérant le cas où p n'est pas compris entre 0 et 1 et le cas où l'utilisateur n'a pas entré de paramètre :

```
function G =google_matrix(M, p)
[m,n]=size(M);
  if exists("p", "l") == 0 then
      p = 0.85;
    end
  if(0>p | p>1) then
          p=0.85;
      end
G=((1-p)/m)*ones(m,n)+p*M;
endfunction
```

Voici le cas où le coefficient d'amortissement p est égale à 1 :

```
-> google_matrix(M,1)
ans  =

  0.    0.3333333   0.    0.    0.3333333
  0.5   0.          0.5   0.    0.3333333
  0.    0.3333333   0.    0.5   0.
  0.    0.          0.5   0.    0.3333333
  0.5   0.3333333   0.    0.5   0.
```

On remarque dans ce cas où la probabilité de changement page aléatoire est nulle, on retombe bien sur la Matrice de transition de départ et que le coefficient d'amortissement n'influe pas sur le pagerank.
Ensuite, nous pouvons modifier la fonction pagerank en prenant en compte le coefficient d'amortissement, cette fonction prend donc désormais en paramètre la matrice de transition M et le coefficient d'amortissement p que l'on considérera à 1 en cas de non-précision par l'utilisateur. On renvoie ensuite le vecteur r des pageranks :

```
function [r] = pagerank (M,p)
    [m,n]=size(M);
    I=zeros(m,n);
    s=0;

    for i=1 : n
        I(i,i)=1;
    end

    M=google_matrix(M, p)-I;

r=kernel(M);

    for i=1 : n
        s=s+r(i);
    end

    for i=1 : n
        r(i)=r(i)/s
    end

endfunction
```

Voici quelques tests effectués avec l'architecture de départ utilisée dans l'exercice :

Pour p=0.9 :

```
--> pagerank(M,0.9)
 ans  =

   0.1674877
   0.2458128
   0.1704433
   0.1704433
   0.2458128
```

Pour p=0.7 :

```
 -> pagerank(M,0.7)
 ans  =

   0.1706924
   0.2371981
   0.1774557
   0.1774557
   0.2371981
```

Pour p=0.5 :

```
-> pagerank(M,0.5)
ans  =

  0.176
  0.228
  0.184
  0.184
  0.228
```

On remarque que plus on diminue le coefficient d'amortissement, plus chaque valeur du vecteur semble tendre vers une valeur similaire. Ce qui signifie que plus (1-p) augmente, plus le côté aléatoire du page rank augmente. Ce qui est confirmé par le vecteur que l'on obtient avec p=0.
Pour p=0 :

```
-> pagerank(M,0)
ans  =

  0.2
  0.2
  0.2
  0.2
  0.2
```

Lorsque (1-p)=1 et que le pageranking ne dépend plus que de l'aléatoire, on a bien équiprobabilité de téléportation sur toutes les pages.

Ainsi, la méthode du pagerank amorti permet de réduire le nombre de calculs sur la matrice nécessaires à la determination du pagerank en incluant un coefficient aléatoire qui permet d'approximer la valeur du pagerank, ce qui permet de réduire considérablement l'utilisation de données. Cependant, il faut noter que plus on réduit le coefficient d'amortissement, moins le calcul du pagerank sera précis et plus il dépendra de l'aléatoire.
Le coefficient idéal serait p=0.85 qui permettrait d'avoir un équilibre entre la précision (celle de la pertinence de la page) et l'utilisation de données (nombre de calculs) idéale pour approximer le pagerank d'une page.

4 Conclusion

Pour conclure, nous avons pu voir qu'il existe deux méthodes pour déterminer le pagerank d'une page.
Tout d'abord, le calcul direct du pagerank (non amorti) qui permet de calculer les pageranks d'un nombre quelconque de pages internet reliées à un nombre quelconque de liens hypertextes variés.
Cependant, cette méthode, à grandes échelles, peut entrainer un très grand nombre de calculs qui pourraient être difficile à gérer par nos ordinateurs. Donc, on utilise le calcul par marche aléatoire qui permet d'approximer la valeur du pagerank. Cette deuxième méthode fonctionne pour un pagerank non amorti ou pour un pagerank amorti, c'est à dire un pagerank incluant un coefficient d'amortissement qui correspond à un changement de page aléatoire. Cette méthode, bien que moins précise pour déterminer le pagerank d'une page, permettra de réduire considérablement le nombre de calculs, ce qui peut s'avérer très utile pour approximer le pagerank de pages à grandes échelles (pour des matrices gigantesques).
Ainsi la méthode du calcul directe sera utilisée lors des traitements de matrices de petites tailles tandis que la méthode du calcul par marche aléatoire donnera des résultats moins précis mais s'utilisera sur des matrices de très grandes tailles.