INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Module Projet 2MIC :

A la découverte de PageRank de Google

OLOUGOUNA Axel

CHOUKRI Ayoub

Promo : 58

Module Projet 2MIC :

A la découverte de PageRank de Google

OLOUGOUNA Axel

CHOUKRI Ayoub

Promo : 58

PLAN

Introduction

Il y a plus de 2100 ans, le roi d'Egypte Ptolémée II(309-308 av. J-C, 246 av. J-C) a décidé de réaliser une idée extraordinaire qui correspond à stocker toutes les connaissances du monde dans une seule et unique bibliothèque : « la fameuse Bibliothèque d'Alexandrie ». Malheureusement, quelques années plus tard, la bibliothèque pris feu et le rêve de Ptolémée II fût réduit à néant. Il y a quelques années, en 1996, deux élèves de l'université de Stanford (Larry Page et Sergey Brin) décidèrent d'indexer tous les liens des pages Web (connaissances). Ils fondèrent ainsi la société BackRub, renommée Google en 15 septembre 1997. Le nom Google vient du symbole mathématique googol, qui représente le nombre 10^{100}. En effet, Google a pu indexer jusqu'à maintenant environ 10^{14} pages dans le rêve d'arriver un jour au fameux nombre 10^{100}.

Aujourd'hui, Google est devenu le moteur de recherche le plus utilisé au monde. Ce succès obtenu est dû à leur efficacité dans le classement des pages selon leur importance. Cela a été rendu possible par l'algorithme du PageRank créé par les fondateurs de Google. Cet algorithme novateur, en pensant le web comme un graphe orienté, a permis de calculer pour chaque site un indice de pertinence. Il a donc permis de fournir aux utilisateurs la meilleure expérience possible en leur évitant de visiter des centaines de sites avant de trouver celui qui correspond le mieux. Ainsi, Google s'est démarqué de la concurrence des premiers moteurs de recherche tels que Altavista ou Yahoo qui ne faisaient que renvoyer les pages contenant le ou les mots clés recherchés. Le nombre de répétitions du mot dans une page la faisait monter dans le classement, ce qui n'est pas assez pertinent.

On se demande alors ce qui démarque l'algorithme du PageRank des autres algorithmes. Comment se fait-il qu'en moins d'une seconde, Google soit capable de nous renvoyer des millions de pages avec les plus intéressantes en haut du classement ?

Dans ce projet, nous essayerons de donner une approche pertinente de l'algorithme du PageRank utilisé par Google jusqu'à récemment. Dans un premier temps, nous procéderons à une approche heuristique de détermination de l'indice de pertinence en nous appuyant sur une modélisation mathématique du web. Dans un second temps, nous réaliserons l'algorithme découlant de cette approche et nous le vérifierons avec des applications concrètes.

I- Premières approches au calcul du PageRank

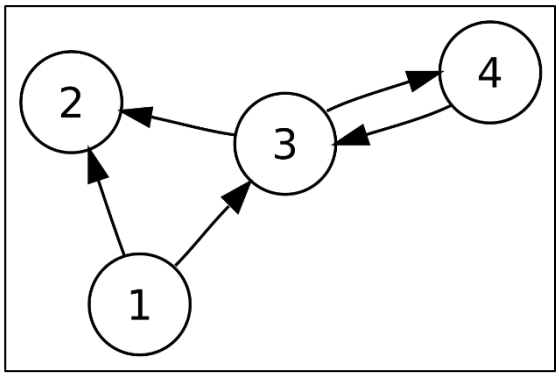
1- Modélisations mathématiques : Le web comme un graphe orienté

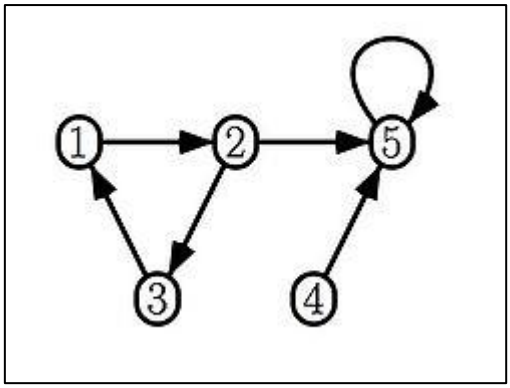
Le but de l'algorithme du PageRank est de classer par pertinence les différentes pages du web. Le web (**Word Wide Web**) est à la base un système basé sur internet permettant de consulter des pages sur des sites grâce à un navigateur. Il peut être vu comme une grande « toile d'araignée » reliant toutes ces pages grâce à des hyperliens.

Ainsi, dans le but de nous simplifier la tâche, nous modéliserons le web comme un graphe orienté **W** = (**S**, **A**) où chaque sommet de l'ensemble **S** (ensemble des sommets) représente une page du Web **W** et dont chaque flèche (arrête orientée) de l'ensemble **A (**ensemble des arrêtes) représente un hyperlien d'une page à une autre. L'ensemble des liens du graphe sera la base de l'algorithme du PageRank.

Par la suite, on notera $\Omega \in \mathcal{M}_n(\mathbb{R})$, **la matrice d'adjacence** du graphe **W**. Ce sera une matrice carrée de dimension **n** × **n** (avec n le nombre de pages du web), et dont chaque élément $\boldsymbol{\omega_{ij}}$ sera égal à **1** s'il y'a un lien de la page numéro i vers la page numéro j, 0 sinon. Toutefois, on ne considérera pas les liens d'une page vers elle-même, c'est à dire que $(\forall \boldsymbol{i} \in [\![\mathbf{1}, \boldsymbol{n}]\!]), \boldsymbol{\omega_{ii}} = \mathbf{0}$.

Par exemple, si on considère les deux mini-web W_1 et W_2 suivants :





Les matrices d'adjacence correspondantes seront :

$$\Omega_1 = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad \Omega_2 = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Le but de l'algorithme du PageRank est d'attribuer à chaque page du Web W un indice de pertinence en fonction de sa popularité. Ainsi, plus le nombre de pages qui pointent vers une page j est grand, plus son PageRank va être aussi important. En plus, la contribution d'un hyperlien entre une page j vers une page i va être d'autant plus importante que la page j est populaire. Finalement, la contribution d'un hyperlien d'une page j vers une page i va être d'autant plus important que le nombre liens sortants de la page j est petit.

On définit le vecteur ligne $\mu \in \mathbb{R}^n$, comme le vecteur contenant les indices de pertinence de toutes les pages du Web.

Dans un souci d'optimisation, on s'arrangera pour que $(\forall \boldsymbol{i} \in [\![\mathbf{1}, \boldsymbol{n}]\!]), \mu_i \in [\mathbf{0}, \mathbf{1}]$, et que la norme 1 du vecteur $\boldsymbol{\mu}$ soit égale à 1 : $\|\mu\|_1 = \mathbf{1}$.

2- Le comptage simple

a) Construction de l'approche

Une première approche du problème est de calculer le PageRank d'une page j du Web W en réalisant un comptage simple du nombre de liens pointant vers cette page j. Pour cela nous allons se baser sur la matrice d'adjacence Ω du web W. En effet, on considérera que le PageRank d'une page j est égale au nombre de liens qui pointent vers cette page j, ça veut dire :$(\forall j \in [\![\mathbf{1}, \boldsymbol{n}]\!]), \mu_j = \sum_{i=1}^{i=n} \boldsymbol{\omega_{ij}}$.

Toutefois, on remarque que $\|\mu\|_1 = \sum_{j=1}^{j=n} |\mu_j| = \sum_{j=1}^{j=n} \sum_{i=1}^{i=n} \boldsymbol{\omega_{ij}} = \boldsymbol{card(A)}$, avec A l'ensemble des arêtes du graphe W.

Ainsi, afin que $\|\mu\|_1 = \mathbf{1}$ **et** $(\forall \boldsymbol{i} \in [\![\mathbf{1}, \boldsymbol{n}]\!]), \mu_i \in [\mathbf{0}, \mathbf{1}]$, on divisera chaque indice μj par le nombre total de liens présents dans le web W, ça veut dire $(\forall \boldsymbol{j} \in [\![\mathbf{1}, \boldsymbol{n}]\!]), \mu_j = \sum_{i=1}^{i=n} \frac{\boldsymbol{\omega_{ij}}}{\boldsymbol{card(A)}}$.

En appliquant cette approche sur les deux graphes (Graphe 1, et Graphe 2) cités précédemment, on trouve :

On rappelle les deux matrices d'adjacence associées :

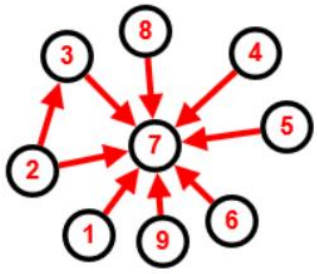
$$\Omega_1 = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad \Omega_2 = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

On trouve que $\mu^1 = \begin{bmatrix} 0 & \frac{2}{5} & \frac{2}{5} & \frac{1}{5} \end{bmatrix}$ et que $\mu^2 = \begin{bmatrix} \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & 0 & \frac{2}{5} \end{bmatrix}$

b) Les défauts de cette approche

Cette approche de comptage simple n'est pas sans défauts. En effet, dans un premier temps, si un développeur web commence à s'amuser en créant de multiples pages web vides qui pointent sur une seul et même page j. Le PageRank de cette page j sera de plus en plus grand.

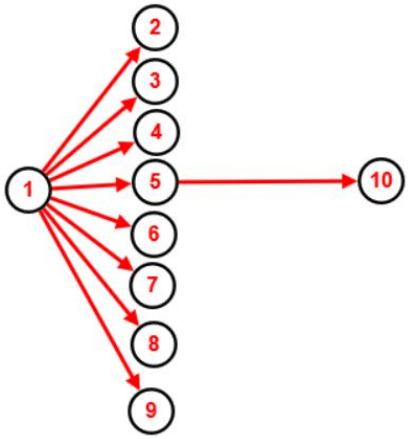
Par exemple, si on considère le Web suivant dans lequel toutes les pages autres que 7 et 2 sont des pages vides.
Le vecteur PageRank μ dans ce cas vaudra :



$$\mu = \begin{bmatrix} 0 & 0 & \frac{1}{9} & 0 & 0 & 0 & \frac{8}{9} & 0 & 0 & 0 \end{bmatrix}$$

On remarque bien que le PageRank de la page numéro 7 est le plus haut PageRank bien que tous les hyperliens visant cette page ne proviennent que de pages vides.

Dans un second temps si on considère le mini-web suivant :



La matrice d'adjacence associé est :

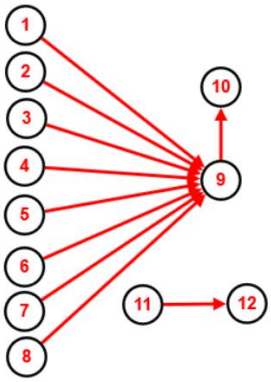
$$\Omega = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Le vecteur Page Rank dans ce cas sera :

$$\mu = \left[0 \quad \frac{1}{9} \quad \frac{1}{9} \quad \frac{1}{9} \quad \frac{1}{9} \quad \frac{1}{9} \quad \frac{1}{9} \quad \frac{1}{9} \quad \frac{1}{9} \quad \frac{1}{9}\right]$$

On remarque bien que $\mu_{10} = \mu_i, (\forall\, i \in [\![2,9]\!])$. Ce qui veut bien dire que la contribution de l'arrête $\{1,3\}$ pour μ_3 est exactement la même contribution de l'arête $\{5,10\}$ pour μ_{10}. Pourtant, la page numéro 1 pointe sur 8 différentes pages, alors que la page numéro 5 ne pointe que sur une seule et unique page qui est la page numéro 10. Ainsi, on peut dire que le problème posé par cette approche est le fait qu'elle considère un seul et même poids pour chaque arête.

Un troisième problème rencontré est le suivant :



En considérant le graphe suivant, le caluc du vecteur PageRank donne :

$$\mu = \left[0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad \frac{8}{10} \quad \frac{1}{10} \quad 0 \quad \frac{1}{10}\right]$$

On remarque que la page numéro 9 est la plus pertinente de ce Web, avec $\mu_9 = \frac{8}{10}$. Or, on remarque bien que $\mu_{10} = \mu_{12} = \frac{1}{10}$. Ça veut dire que l'arête $\{11,12\}$, et l'arête $\{9,10\}$ ont exactement le même poids, bien que la page numéro 9 est plus pertinente que la page numéro 11.

Ainsi, bien que le comptage soit très simple à implémenter, on remarque bien qu'il connait de nombreux défauts, ce qui va impacter le classement par pertinence des pages du web.

3- Le comptage pondéré

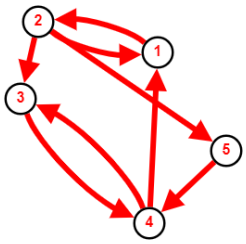
Ce comptage vient résoudre le problème du poids du vote d'une page. En effet, avec le comptage simple, on a pu constater que toutes les pages avaient le même poids, pourvu le nombre de pages vers lesquelles elle pointait. C'est à ce problème que vient le comptage pondéré.

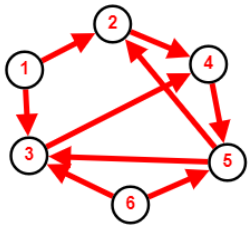
a) Construction de l'approche

Dans cette approche du problème, on divise le vote d'une page par le nombre de liens qui en sortent. Ainsi, le PageRank devient $(\forall j \in [\![1, n]\!]), \mu_j = \sum_{i=1}^{i=n} \frac{\omega_{ij}}{\sum_{k=1}^{k=n} \omega_{ik}}$.

On suppose dans un premier temps qu'une page pointe nécessairement vers une autre page. Ce qui qui veut dire $\sum_{k=1}^{k=n} \omega_{ik} \neq 0$.

On applique cette approche sur les deux graphes Ω_1 et Ω_2 suivants :





Les matrices d'adjacence associées sont :

$$\Omega_1 = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}, \quad \Omega_2 = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

On trouve que $\mu^1 = [0{,}16667 \quad 0{,}2 \quad 0{,}166667 \quad 0{,}4 \quad 0{,}06666]$ et que $\mu^2 = [0 \quad 0{,}25 \quad 0{,}16667 \quad 0{,}41667 \quad 0 \quad 0{,}16667]$

b) Les défauts de cette approche

Le comptage pondéré n'a pas résolu tous les problèmes signalés dans la partie du comptage simple. En effet, il n'a résolu que le problème du poids de vote d'une page pour une autre. Les problèmes 1 et 3 évoqués dans l'approche précédente ne sont toujours pas résolus. En effet, les problèmes numéros 3 et 1 requièrent le fait de pondérer par des poids qui dépendent de la page qui pointe.

D'où l'idée de considérer une approche récursive.

4- Le comptage récursif.

a) Construction de l'approche

La dernière approche (Comptage pondéré) calcule les indices de pertinence des pages μ_i à travers la formule : $(\forall j \in [\![1, n]\!]), \mu_j = \frac{1}{n} \times \sum_{i=1}^{i=n} \frac{\omega_{ij}}{\sum_{k=1}^{k=n} \omega_{ik}}$. Or pour remédier encore aux problèmes numéro 1 et 3 vu dans la partie du comptage simple, on propose de pondérer chaque terme par son indice de pertinence pour obtenir la formule récursive suivante :

$$(\forall j \in [\![1, n]\!]) \ \mu_j = \sum_{i=1}^{i=n} \mu_i \times \frac{\omega_{ij}}{\sum_{k=1}^{k=n} \omega_{ik}}.$$

Cette formule, bien concise mathématiquement, est en revanche très lourde en matière de complexité (Web constitué de presque 10^{10} pages). Pour cela, on va essayer d'écrire cette formule sous une forme matricielle.

On considère la matrice $\mathbf{D} \in \mathcal{M}_n(\mathbb{R})$, définie comme suit :

$$(\forall\, \boldsymbol{i} \in [\![\mathbf{1}, \boldsymbol{n}]\!]), (\forall\, \boldsymbol{j} \in [\![\mathbf{1}, \boldsymbol{n}]\!]), \boldsymbol{D_{ij}} = \frac{\omega_{ij}}{\sum_{k=1}^{k=n} \omega_{ik}}.$$

On aura alors $(\forall\, \boldsymbol{j} \in [\![\mathbf{1}, \boldsymbol{n}]\!])\ \mu_j = \sum_{i=1}^{i=n} \mu_i \times \boldsymbol{D_{ij}}$ i.e. $\mu = \mu \times \boldsymbol{D}$.

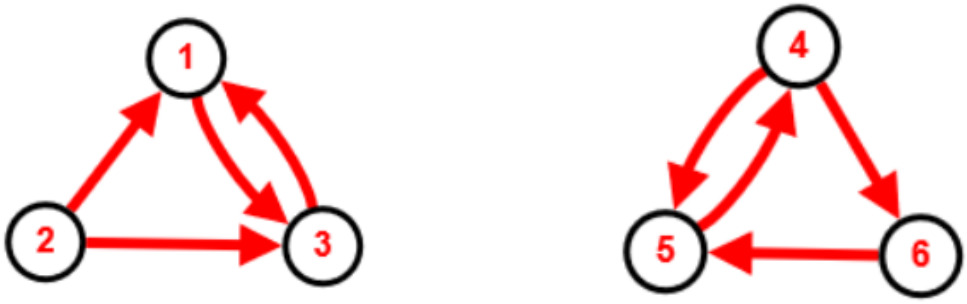
Ainsi, on remarque bien que μ est bien un vecteur propre associé à la valeur propre 1 de la matrice D. De ce fait, le calcul du vecteur μ d'indices de pertinence des pages du Web ne reposera finalement que sur la résolution de l'équation $\mu = \mu \times D$, ce qui veut dire, la détermination du vecteur propre à coefficients en 0 et 1, de norme 1 égale à 1 et associé à la valeur propre 1 de la matrice D.

Autrement dit, il faut chercher l'unique vecteur μ qui satisfait les critères suivants :

$$\mathbf{S:} \begin{cases} \mu = \mu \times \boldsymbol{D}. \\ \|\mu\|_1 = \mathbf{1} \\ (\forall \boldsymbol{i} \in [\![\mathbf{1}, \boldsymbol{n}]\!]), \mu_i \in [\mathbf{0}, \mathbf{1}] \end{cases}$$

b) Les défauts du comptage récursif : la non-unicité de la solution

La question qu'on peut se poser ici est à propos de l'unicité du vecteur μ qui satisfait tous les critères du système S. En effet, on peut prouver que dans certains cas, le vecteur μ n'est pas unique. Par exemple, si on considère le mini-web suivant :



La matrice D associé à ce web est :

$$\mathbf{D} = \begin{pmatrix} 0 & 0 & \frac{1}{2} & 0 & 0 & 0 \\ 1 & 0 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & 0 \end{pmatrix}$$

Si on essaie de déterminer l'ensemble des valeurs propres de cette matrice, on trouve que $\boldsymbol{Spec}(\mathrm{D}) = \left\{ \mathbf{1} \middle| \frac{-1}{2} + \frac{i}{2} \middle| \frac{1}{2} + \frac{i}{2} \right\}$, or $\boldsymbol{E_1} = \boldsymbol{Ker}(\mathbf{D} - \mathbf{1} * \boldsymbol{I_6}) = \boldsymbol{vect}(\, \boldsymbol{e_1} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{2} \\ \mathbf{2} \\ \mathbf{1} \end{bmatrix}, \boldsymbol{e_2} = \begin{bmatrix} \mathbf{0.5} \\ \mathbf{1} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix})$ ce qui veut dire $\boldsymbol{dim}\, \boldsymbol{E_1} = \mathbf{2}$. De ce fait, il suffit de prendre $\mu_1 = \frac{\boldsymbol{e1}}{\|\boldsymbol{e1}\|_1}, \boldsymbol{et}\ \mu_2 = \frac{\boldsymbol{e2}}{\|\boldsymbol{e2}\|_1}$, on voit bien que les deux vecteurs $\mu_1$et μ_2 vérifient bien le système S.

Toutefois, on remarque bien que la matrice possède une caractéristique très importante qui se manifeste dans la stochasticité des lignes **non nulles** de la matrice D.

c) Interprétation de la matrice D.

Comme le vecteur PageRank μ est un vecteur tel que $\|\mu\|_1 = \mathbf{1}$ et $(\forall \boldsymbol{i} \in [\![\mathbf{1}, \boldsymbol{n}]\!]), \mu_i \in [\mathbf{0}, \mathbf{1}]$. On peut dire que $(\forall \boldsymbol{i} \in [\![\mathbf{1}, \boldsymbol{n}]\!])$, μ_i est la probabilité qu'un internaute aléatoire se retrouve dans la page i.

Ainsi, plus μ_i est proche de 1 plus la page i est pertinente et vice versa.

De ce fait, par la formule des probabilités totales on peut dire que :

$$\mathbf{P}(\text{"Aller vers la page i"}) = \mu_i = \sum_{j=1}^{j=n} \mathbf{P}(\text{"Aller vers la page i"} \cap \text{"Etre dans la page j"})$$

Donc

$$\mu_i = \sum_{j=1}^{j=n} \mathbf{P}(\text{"Aller vers la page i"} \mid \text{"Etre dans la page j"}) \times \mathbf{P}(\text{"Etre dans la page j"})$$

Or on a $\mathbf{P}(\text{"Etre dans la page j"}) = \mu_j = \mathbf{P}(\text{"Aller vers la page j"})$. De ce fait, on aura :

$$\mu_i = \sum_{j=1}^{j=n} \mathbf{P}(\text{"Aller vers la page i"} \mid \text{"Etre dans la page j"}) \times \mu_j$$

Par analogie avec l'expression : $\mu_i = \sum_{j=1}^{j=n} \boldsymbol{D_{ji}} \times \mu_j$

On peut dire que $\boldsymbol{D_{ji}} = \mathbf{P}(\text{"Aller vers la page i"} \mid \text{"Etre dans la page j"})$. En d'autres termes $\boldsymbol{D_{ji}} = \boldsymbol{P}(\text{"}\boldsymbol{Passer\ de\ la\ page\ j\ vers\ la\ page\ i}\text{"})$.

En d'autres termes, on peut considérer que la matrice D est une matrice de transition.

Ceci dit, la modélisation du problème faite juste que là est une modélisation qui considère un internaute aléatoire qui se balade entre toutes les pages du web possibles à

atteindre. La pertinence d'une page j selon ce modèle repose sur la probabilité que l'internaute vient à la page j. Cette probabilité est calculée à travers la formule :

$$\mu_i = \sum_{j=1}^{j=n} \mathbf{P}(\text{"Aller vers la page i"} \mid \text{"Etre dans la page j"}) \times \mathbf{P}(\text{"Etre dans la page j"})$$

Autrement dit :

$$\mu_i = \sum_{j=1}^{j=n} \mathbf{P}(\text{"Aller vers la page i"} \mid \text{"Etre dans la page j"}) \times \mu_j$$

I.e.

$$\mu_i = \sum_{j=1}^{j=n} \boldsymbol{D_{ji}} \times \mu_j$$

Cependant, un problème se pose. En effet, suivant cette modélisation, un internaute, s'il se trouve coincé dans un fichier sans lien ne peut clairement plus continuer sa randonnée dans le Web. D'autre part, s'il se trouve dans un cycle de page de tel manière que l'on revienne infiniment et sans cesse sur ces mêmes pages, l'internaute ne pourra rien faire d'autre que de tourner en rond, visitant toujours les mêmes pages.

d) Un ajustement de la stochasticité

Pour remédier à ce problème, on pourra dire que l'internaute aura dans cette situation la possibilité de se rendre sur n'importe quelle page du Web et ce avec une parfaite équiprobabilité. Ce qui revient en fait à remplacer les lignes de zéro de la matrice D par des lignes de $\frac{1}{N}$. Ainsi on peut alors définir une nouvelle matrice S :

$$\boldsymbol{S} = \boldsymbol{D} + \frac{\mathbf{1}}{\boldsymbol{N}} \times \boldsymbol{P}$$

Avec $P \in \mathcal{M}_N(\mathbb{R})$, de tel façon que P est une matrice carrée de ligne i égale au vecteur-ligne unité si la page numéro i ne pointe vers aucune autre page.

L'utilité de cet ajustement sera perçue dans la prochaine approche.

II. Le PageRank de Google

1- La matrice G de Google

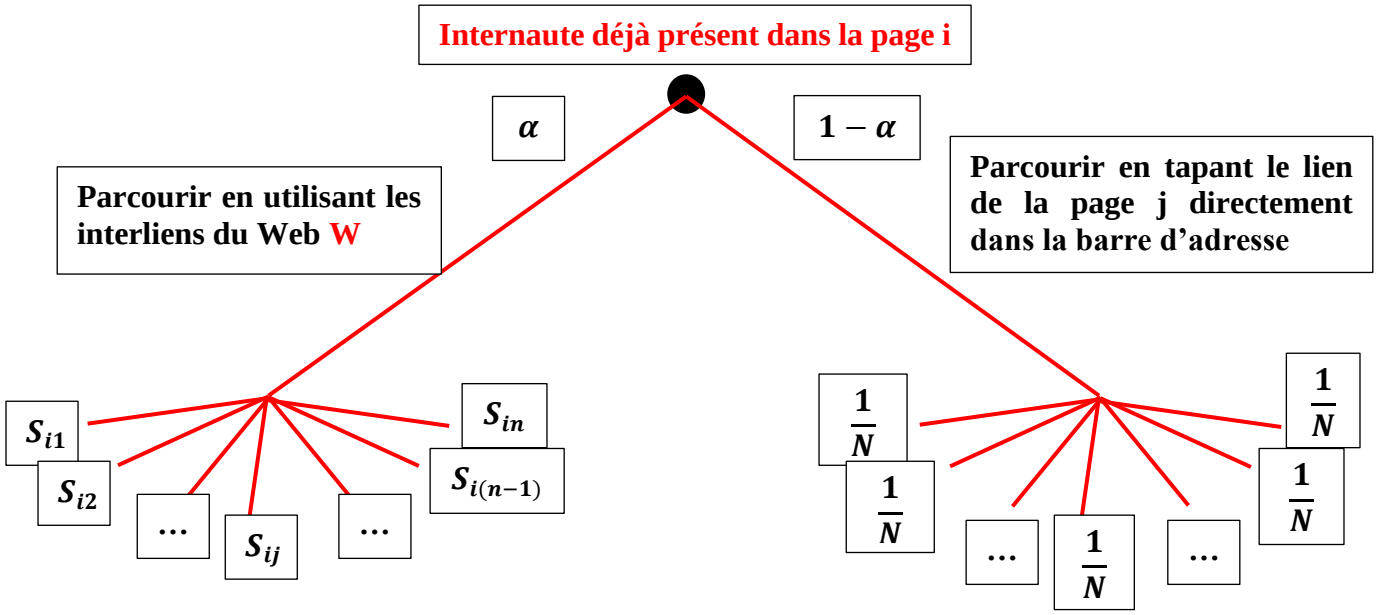
a) Le surfeur aléatoire

L'approche récursive précédente serait une très belle approche pour calculer l'indice de pertinence des pages d'un web W dans lequel un internaute aléatoire ne pourra parcourir ses pages qu'à partir de ses liens internes. En effet, cette approche récursive ne compte aucun autre facteur de navigation.

Il arrive très souvent qu'un internaute abandonne purement un site pour se rendre sur un autre site, sans aucun lien avec le précédent. Pour cela, il s'avère nécessaire de lister et prendre en compte tous les facteurs ou méthodes de navigation.

Dans cette dernière approche on ne considèrera que deux facteurs de navigation qu'on modélisera dans l'arbre de probabilité suivant :

En supposant que l'internaute est déjà présent dans une page i, on essayera de modéliser toutes les méthodes de parcours avec lesquels il pourra arriver à la page j.



b) La construction de la matrice G

D'après cet arbre de probabilité présenté ci-dessus on peut dire que :

« * » : $\mathbf{P("Aller\ vers\ la\ page\ j"\ |"Etre\ dans\ la\ page\ i")} = \boldsymbol{\alpha} \times S_{ij} + (1-\alpha) \times \frac{1}{N}$

Avec α, la probabilité que l'internaute utilise plutôt les interliens présents pour se déplacer vers la page j.

Ainsi suivant la formule des probabilités totales :

$$\mathbf{P}(\text{"Aller vers la page j"}) = \mu_j = \sum_{i=1}^{i=n} \mathbf{P}(\text{"Aller vers la page j"} \cap \text{"Etre dans la page i"})$$

Donc

$$\mu_j = \sum_{i=1}^{i=n} \mathbf{P}(\text{"Aller vers la page j"} \mid \text{"Etre dans la page i"}) \times \mathbf{P}(\text{"Etre dans la page i"})$$

Alors d'après « * » :

$$\mu_j = \sum_{i=1}^{i=n}(\alpha \times S_{ij} + (1-\alpha) \times \frac{1}{N}) \times \mu_i = \sum_{i=1}^{i=n} \alpha \times S_{ij} + \sum_{i=1}^{i=n}(1-\alpha) \times \frac{1}{N}) \times \mu_i \quad .$$

Donc

$$\mu_j = \sum_{i=1}^{i=n} \alpha \times S_{ij} \times \mu_i + \sum_{i=1}^{i=n}(1-\alpha) \times \frac{1}{N}) \times \mu_i = \sum_{i=1}^{i=n} \alpha \times S_{ij} \times \mu_i + (1-\alpha) \times \frac{1}{N} \times \sum_{i=1}^{i=n} \mu_i$$

Or $\|\mu\|_1 = 1$, donc

$$\mu_j = (1-\alpha) \times \frac{1}{N} + \sum_{i=1}^{i=n} \alpha \times S_{ij} \times \mu_i$$

En notant PR(A) : le PageRank de la page A, et en notant « vrs(A) » l'ensemble des pages qui pointent vers la page A, et L(A) le nombre d'interliens sortant de A :

On retrouve la formule de Google :

$$PR(A) = (1-\alpha) \times \frac{1}{N} + \alpha \times \sum_{B \in vrs(A)} \frac{PR(B)}{L(B)}$$

Or pour pouvoir résoudre cette équation analytiquement, il vaut mieux l'ecrire sous la forme d'une équation matricielle linéaire d'inconnu μ.

Pour faire ceci, on a qu'à prendre l'équation initiale :

$$\mu_j = \sum_{i=1}^{i=n}(\alpha \times S_{ij} + (1-\alpha) \times \frac{1}{N}) \times \mu_i$$

Et c'est là où intervient la fameuse et unique matrice de google G définit comme suit :

$$G_{ij} = \alpha \times S_{ij} + (1-\alpha) \times E_{ij} \text{ avec } E = \begin{pmatrix} \frac{1}{N} & \cdots & \frac{1}{N} \\ \vdots & \ddots & \vdots \\ \frac{1}{N} & \cdots & \frac{1}{N} \end{pmatrix}.$$

Ainsi chercher le vecteur de probabilité μ reviendrait juste à résoudre le système :

$$\mathbf{S:} \begin{cases} \mu = \mu \times G. \\ \|\mu\|_1 = 1 \\ (\forall i \in [\![1, n]\!]), \mu_i \in [0, 1] \end{cases}$$

Remarque :

Considérons une page i qui ne pointe vers aucune autre page :

Calculons pour une page donnée le coefficient de la matrice : $\boldsymbol{S_{ij}}$ dont on rappelle la formule établie après l'ajustement de stochasticité

$$\boldsymbol{S_{ij} = D_{ij} + \frac{1}{N} \times P_{ij} = \frac{1}{N} \times P_{ij} = \frac{1}{N}}$$

Donc d'après l'arbre de probabilité :

$\mathbf{P}$("**Aller vers la page j**" |"**Etre dans la page i**") $= \boldsymbol{\alpha \times S_{ij} + (1-\alpha) \times \frac{1}{N} = \alpha \times \frac{1}{N} + (1-\alpha) \times \frac{1}{N} = \frac{1}{N}}$.
Ce qui est bien logique. En effet, une fois l'internaute dans la page i, il ne lui restera aucun moyen de naviguer dans le web qu'en tapant l'adresse de la page web et ce avec $\frac{1}{N}$ comme probabilité. Toutefois, sans ajustement de stochasticité, cette probabilité vaudra $(\mathbf{1-\alpha}) \times \frac{1}{N}$: ce qui est absurde.

- **Stochasticité de la matrice G**

Par la définition de G : $(\forall \boldsymbol{i,j} \in [\![\mathbf{1}, \boldsymbol{n}]\!])\ \boldsymbol{G_{ij} = \alpha \times S_{ij} + (1-\alpha) \times E_{ij}}$

Soit $\boldsymbol{i} \in [\![\mathbf{1}, \boldsymbol{n}]\!]$, $\sum_{\mathbf{j=1}}^{\mathbf{j=n}} \mathbf{G_{ij}} = \sum_{\mathbf{j=1}}^{\mathbf{j=n}} (\boldsymbol{\alpha} \times \mathbf{S_{ij}} + (\mathbf{1-\alpha}) \times \mathbf{E_{ij}}) = \boldsymbol{\alpha} \times \sum_{\mathbf{j=1}}^{\mathbf{j=n}} \mathbf{S_{ij}} + (\mathbf{1-\alpha}) \times \sum_{\mathbf{j=1}}^{\mathbf{j=n}} \mathbf{E_{ij}}$

Donc $\sum_{j=1}^{j=n} \boldsymbol{G_{ij}} = \boldsymbol{\alpha} \times \mathbf{1} + (\mathbf{1-\alpha}) \times \sum_{j=1}^{j=N} \frac{\mathbf{1}}{\boldsymbol{N}} = \boldsymbol{\alpha} + (\mathbf{1-\alpha}) = \mathbf{1}$.
On conclut donc que G est bien une matrice stochastique.

2- Vers la recherche du vecteur PageRank

Il faut maintenant s'assurer que G admet bien 1 comme valeur propre. En effet, comme G est une matrice stochastique. I.e. la somme de chaque ligne est égale à 1.

En effet, en considérant $V = [1 \quad \cdots \quad 1]$:

$(\forall \boldsymbol{i} \in [\![\mathbf{1}, \boldsymbol{n}]\!])$, $\boldsymbol{(VG^t)_i} = \sum_{\mathbf{j=1}}^{\mathbf{j=n}} \mathbf{V_{1j}} \mathbf{G^t_{ji}} = \sum_{\mathbf{j=1}}^{\mathbf{j=n}} \mathbf{1} \times \mathbf{G^t_{ji}} = \sum_{\mathbf{j=1}}^{\mathbf{j=n}} \mathbf{G^t_{ji}} = \mathbf{1}$. Donc $\boldsymbol{VG^t = V}$. Donc $\boldsymbol{G^t}$ admets bien 1 comme valeur propre. Reste à savoir si 1 est valeur propre de la transposé de $\boldsymbol{G^t}$.
En effet, soit $\boldsymbol{x} \in \mathbb{C}, \boldsymbol{on\ a}\ \boldsymbol{\wp_{G^t}(x) = det(G^t - xI_n) = det((G - xI_n)^t) = det(G - xI_n) = \wp_G(x)}$. Donc G et $\boldsymbol{G^t}$ ont bien le même polynôme caractéristique, et donc 1 est bien valeur propre de G.

- Que signifie le paramètre α ?

On rappelle la formule de G : $\boldsymbol{G_{ij} = \alpha \times S_{ij} + (1-\alpha) \times E_{ij}}$

D'après l'arbre de probabilité établi ci-dessus, il est bien clair que α = P (« L'internaute aléatoire utilise les interliens du web pour naviguer »). Ainsi 1-α concerne bien la probabilité que l'internaute aléatoire utilise plutôt la barre d'adresse pour naviguer.

Cette probabilité α exercera une grande influence sur le vecteur PageRank μ. En effet, plus α est proche de 1, plus le calcul du PageRank inclue la matrice de transition G. Non seulement ça, mais cette probabilité influe beaucoup sur la vitesse de convergence du calcul du PageRank. Plus, elle est proche de 1, plus le calcul est beaucoup plus lent. Il faudra bien donc faire un compromis. Choisir un α assez proche de 1 pour inclure plus la matrice G dans le calcul et en même temps un α assez petit pour converger le plus rapidement possible.

Toutefois, après plusieurs études de statistiques, il s'est bien avéré que α = 0.85 est bien la probabilité qu'un internaute aléatoire exploite les interliens du Web pour naviguer.

La question qui se pose maintenant est l'existence ou non d'une solution unique pour le système S. Face à cette interrogation, il existe un théorème qui nous aide à répondre à cette question.

a) L'existence et l'unicité de la solution : Le théorème de Perron-Frobenius

Le théorème de Perron-Frobenius est un théorème donnant l'existence d'une valeur propre "dominante" pour une matrice primitive.

Définition : Une matrice est dite primitive si elle admet une puissance dont tous les termes sont strictement positifs.

Le théorème de Perron-Frobenius s'annonce comme suit :

Théorème de Perron-Frobenius :

Soit A une matrice primitive. Alors elle admet une valeur propre réelle strictement positive r>0 telle que

- pour toute autre valeur propre s de A, on a |s|<r (r est une valeur propre dominante);
- r est une valeur propre simple (son espace propre associé est de dimension 1);
- il existe un unique vecteur x+ de norme 1 à coordonnées strictement positives tel que Ax+=rx+

b) Démonstrations et preuves

La matrice G est bien une matrice primitive. En effet, $\boldsymbol{G_{ij} = \alpha \times S_{ij} + (1-\alpha) \times E_{ij} > 0}$, car $\boldsymbol{0 < \alpha < 1, et\ E_{ij} = \frac{1}{N} > 0, S_{ij} > 0}$.

On voit bien que toutes les conditions du théorème de Frobenius sont bien vérifiées.

Donc G admet une valeur propre réelle r strictement positive telle que :

- pour toute autre valeur propre s de A, on a |s|<r (r est une valeur propre dominante);
- r est une valeur propre simple (son espace propre associé est de dimension 1);

- il existe un unique vecteur μ de norme 1 à coordonnées strictement positives tel que $\boldsymbol{r} \times \mu = \mu \times \boldsymbol{G}$.

Il reste à démontrer que r = 1, pour prouver le résultat qu'on veut :

Soit $\Omega \in \boldsymbol{\mathcal{M}_n}(\mathbb{R})$ une matrice stochastique. Soit r une valeur propre de **Ω**. Soit $X \in Ker(\Omega - rI_n)$.

On a $\Omega X = rX$. Donc $\|\Omega X\|_\infty = \|rX\|_\infty = |r| \times \|X\|_\infty \leq |||\Omega||| \times \|X\|_\infty$.

Donc $|r| \leq |||\Omega|||$.

Soit $X \in \mathbb{R}^n$. On a $(\forall\, i \in [\![1, n]\!])\ (\Omega X)_i = \sum_{k=1}^{k=n} \Omega_{ik} X_{k1}$. Or $(\forall\, i \in [\![1, n]\!])\ X_i \leq \|X\|_\infty$.

Or du fait que **Ω** est une matrice stochastique i.e. la somme de chaque ligne vaut exactement 1.

$(\forall\, \boldsymbol{i} \in [\![\mathbf{1}, \boldsymbol{n}]\!])\ (\Omega \boldsymbol{X})_{\boldsymbol{i}} = \sum_{\boldsymbol{j=1}}^{\boldsymbol{j=n}} \Omega_{\boldsymbol{ik}} \boldsymbol{X_{k1}} \leq \sum_{\boldsymbol{j=1}}^{\boldsymbol{j=n}} \Omega_{\boldsymbol{ik}} \|\boldsymbol{X}\|_\infty = \|\boldsymbol{X}\|_\infty \times \sum_{\boldsymbol{k=1}}^{\boldsymbol{k=n}} \Omega_{\boldsymbol{ik}} = \|\boldsymbol{X}\|_\infty$.

Et puis par passage au sup on trouve : $\|\Omega X\|_\infty \leq \|X\|_\infty$.

Donc $(\forall\, \boldsymbol{X} \in \mathbb{R}^{\boldsymbol{n}} / \boldsymbol{X} \neq \mathbf{0})\ \frac{\|\Omega \boldsymbol{X}\|_\infty}{\|\boldsymbol{X}\|_\infty} \leq \mathbf{1}$. Ainsi par passage au sup sur $\mathbb{R}^n/\{0\}$. On trouve $|||\Omega||| \leq 1$.

Ainsi, on déduit que $|r| \leq |||\Omega||| \leq 1$.

D'autre part, comme 1 est une valeur propre de la matrice stochastique G, alors 1 est inférieur à la valeur propre dominante r, donc $1 \leq r$. Donc $r \leq 1\ et\ r \geq 1$. On conclut bien que r = 1.

Alors il existe un unique vecteur μ de norme 1 à coordonnées strictement positives tel que $\mu = \mu \times \boldsymbol{G}$.

Ce qui veut bien dire qu'il existe un unique vecteur μ qui vérifie :

$$\begin{cases} \mu = \mu \times \boldsymbol{G}. \\ \|\mu\|_{\mathbf{1}} = \mathbf{1} \\ (\forall \boldsymbol{i} \in [\![\mathbf{1}, \boldsymbol{n}]\!]), \mu_{\boldsymbol{i}} \in [\mathbf{0}, \mathbf{1}] \end{cases}$$

III. Le codage du PageRank

Jusqu'ici, on a déjà prouvé l'existence unique du vecteur de probabilité μ satisfaisant le système suivant :

$$\begin{cases} \mu = \mu \times \boldsymbol{G}. \\ \|\mu\|_1 = 1 \\ (\forall \boldsymbol{i} \in [\![1, \boldsymbol{n}]\!]), \mu_i \in [0, 1] \end{cases}$$

Ainsi, pour déterminer le vecteur, il s'avère nécessaire de déterminer l'espace propre caractéristique de la matrice G, associé à la valeur propre 1. Faire cela n'est pas du tout facile par un calcul direct, plusieurs méthodes s'offrent à nous (Décomposition QR, Décomposition LU, méthode de la puissance). Pour des soucis de simplification, on a opté pour la méthode de la puissance.

1- La méthode de la puissance

La méthode de la puissance repose sur le théorème suivant :

Théorème — Soit A une matrice carrée d'ordre n et $(\lambda_1,\lambda_2,...,\lambda_n)$ ses valeurs propres. On suppose :

$$|\lambda_1| > \max(|\lambda_2|, \ldots, |\lambda_n|).$$

On considère la somme directe $\mathbb{C}_n=E\oplus F$ où E est le sous-espace caractéristique de A associé à la valeur propre λ_1, et F est le sous-espace caractéristique de A associé aux autres valeurs propres.

Alors si $w^{(0)}\notin F$, la suite de vecteurs $(w^{(n)})$ définie par la relation de récurrence

$$\forall n \in \mathbb{N},\ w^{(n+1)} = \frac{1}{||Aw^{(n)}||} Aw^{(n)}$$

vérifie

- $w^{(n)}\neq 0$ pour tout $n\in\mathbb{N}$.
- $||Aw^{(n)}|| \longrightarrow |\lambda_1|$ lorsque $n\to\infty$.
- $\left(\frac{\overline{\lambda_1}}{|\lambda_1|}\right)^n w^{(n)} \longrightarrow v$ lorsque $n\to\infty$, où v est un vecteur unitaire de A associé à la valeur propre λ_1.
- Si j est une composante non nulle du vecteur v, alors $\frac{(Aw^{(n)})_j}{w_j^{(n)}} \longrightarrow \lambda_1$ lorsque $n\to\infty$.

A partir du théorème de Perron-Frobenius, on sait déjà que 1 est bien la valeur propre dominante de la matrice G. Il reste à chercher un vecteur $\boldsymbol{W^{(0)}}$ qui n'appartient pas à E orthogonal.

En prenant, $\boldsymbol{W^{(0)}} = [\mathbf{1} \ \cdots \ \mathbf{1}]$. Dire que $\boldsymbol{W^{(0)}} \in E^{\perp}$ équivaut à dire que $(\forall\ \boldsymbol{v} \in \boldsymbol{E}), \langle \boldsymbol{W^{(0)}} | \boldsymbol{v} \rangle_{\boldsymbol{C}} = \mathbf{0}$. Cependant, d'après le théorème de Frobenius, L'espace vectoriel E est de dimension 1. Donc, le vecteur de probabilité μ, vu qu'il n'est pas nul, constitue à lui seul une base de E.

Donc $\boldsymbol{W^{(0)}} \in \boldsymbol{E^{\perp}}$ équivaut à dire que $\langle \boldsymbol{W^{(0)}} | \mu \rangle_{\boldsymbol{C}} = \mathbf{0}$, **i.e.** $\sum_{k=1}^{k=N} \mu_k = \mathbf{0}$. Donc comme $(\forall \boldsymbol{i} \in [\![1, \boldsymbol{n}]\!]), \mu_i \in [0, 1]$. Cela revient à dire que $\sum_{k=1}^{k=N} |\mu_k| = \|\mu\|_1 = \mathbf{0}$. Ce qui est absurde.

On aboutit donc à la conclusion que $\boldsymbol{W^{(0)}} = [\mathbf{1} \quad \cdots \quad \mathbf{1}]$ n'appartient pas à E orthogonal.

Ainsi, d'après la méthode de la puissance, et par le procédé d'itération. La suite $\left(\frac{\overline{\lambda_1}}{|\lambda_1|}\right)^{\boldsymbol{n}} \times \boldsymbol{W^{(n)}} = \boldsymbol{W^{(n)}}$ $\boldsymbol{car}$ $\boldsymbol{\lambda_1 = 1}$ converge bien vers le vecteur PageRank μ. Reste à démontrer pourquoi ?

On supposera dans un premier temps que toutes les valeurs propres sont réelles.

Soit $M \in \mathcal{M}_m(\mathbb{R})$, et $(\boldsymbol{\lambda_1 \ldots\ldots \lambda_m})$ ses valeurs propres. Et soit $(\boldsymbol{e_1 \ldots\ldots e_m})$ $\boldsymbol{une\ base\ de\ vecteur\ propores\ orthonormés\ de}$ $\mathbb{R}^{\boldsymbol{m}}$ Soit $\boldsymbol{W^{(0)}}$ un vecteur tel que $\boldsymbol{W^{(0)}} \notin E^{\perp}$. I.e. $\boldsymbol{e_1}^*(\boldsymbol{W^{(0)}}) \neq \mathbf{0}$.

On suppose que $(\forall \boldsymbol{i} \in [\![\mathbf{1}, \boldsymbol{n}]\!]), |\lambda_i| \leq |\lambda_1|$.

On pose deux suites $(\boldsymbol{X_n})_{\boldsymbol{n}\in\mathbb{N}}$, $(\boldsymbol{Y_n})_{\boldsymbol{n}\in\mathbb{N}}$ définis comme suit :

$$\begin{cases} \boldsymbol{X_0} = \boldsymbol{W^{(0)}} \\ \boldsymbol{X_{n+1}} = \boldsymbol{MY_n} \end{cases} \text{ et } \boldsymbol{Y_n} = \frac{\boldsymbol{X_n}}{\|\boldsymbol{X_n}\|_2}$$

Notre but est de chercher un terme général pour la suite $(\boldsymbol{Y_n})_{\boldsymbol{n}\in\mathbb{N}}$ qu'on pourra conjecturer et puis démontrer par récurrence.

D'abord décomposons $\boldsymbol{W^{(0)}}$ dans la base de vecteurs propres **:** $\boldsymbol{W^{(0)}} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i e_i}$

On a $\boldsymbol{Y_0} = \frac{\boldsymbol{X_0}}{\|\boldsymbol{X_0}\|_2} = \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i e_i}\right\|_2}$.

Donc $\boldsymbol{X_1} = \boldsymbol{MY_0} = \boldsymbol{M} \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i e_i}\right\|_2} = \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i M e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i e_i}\right\|_2} = \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i \lambda_i e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i e_i}\right\|_2} = \boldsymbol{\lambda_1} \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i e_i}\right\|_2}$.

Normalisons $\boldsymbol{X_1}$ **:** $\boldsymbol{Y_1} = \frac{\boldsymbol{X_1}}{\|\boldsymbol{X_1}\|_2} = \frac{\boldsymbol{\lambda_1} \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i e_i}\right\|_2}}{\left\|\boldsymbol{\lambda_1} \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i e_i}\right\|_2}\right\|_2} = \frac{\boldsymbol{\lambda_1}}{|\boldsymbol{\lambda_1}|} \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}\right\|_2} = \mathbf{signe}(\boldsymbol{\lambda_1}) \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}\right\|_2}$.

Il est bien clair que $\boldsymbol{X_2} = \boldsymbol{MY_1} = \boldsymbol{M} \times \mathbf{signe}(\boldsymbol{\lambda_1}) \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}\right\|_2} = \mathbf{signe}(\boldsymbol{\lambda_1}) \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{M e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}\right\|_2}$

$$\boldsymbol{Donc}\ \boldsymbol{X_2} = \mathbf{signe}(\boldsymbol{\lambda_1}) \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}^2}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}\right\|_2} = \mathbf{signe}(\boldsymbol{\lambda_1}) \times \boldsymbol{\lambda_1} \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}^2}{\boldsymbol{\lambda_1}^2} \boldsymbol{e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}\right\|_2} = \mathbf{signe}(\boldsymbol{\lambda_1}) \times \boldsymbol{\lambda_1} \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \left(\frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}}\right)^2 \boldsymbol{e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}\right\|_2}.$$

Et donc ensuite, $\boldsymbol{Y_2} = \frac{\boldsymbol{X_2}}{\|\boldsymbol{X_2}\|_2} = \frac{\mathbf{signe}(\boldsymbol{\lambda_1}) \times \boldsymbol{\lambda_1} \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \left(\frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}}\right)^2 \boldsymbol{e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}\right\|_2}}{\left\|\mathbf{signe}(\boldsymbol{\lambda_1}) \times \boldsymbol{\lambda_1} \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \left(\frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}}\right)^2 \boldsymbol{e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}} \boldsymbol{e_i}\right\|}\right\|_2}$

$$\boldsymbol{Ainsi}\ \boldsymbol{Y_2} = \boldsymbol{signe}\left(\mathbf{signe}(\boldsymbol{\lambda_1})\right) \times \mathbf{signe}(\boldsymbol{\lambda_1}) \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \left(\frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}}\right)^2 \boldsymbol{e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \left(\frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}}\right)^2 \boldsymbol{e_i}\right\|_2} = \mathbf{signe}(\boldsymbol{\lambda_1})^2 \times \frac{\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \left(\frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}}\right)^2 \boldsymbol{e_i}}{\left\|\sum_{\boldsymbol{i=1}}^{\boldsymbol{i=m}} \boldsymbol{\omega_i} \left(\frac{\boldsymbol{\lambda_i}}{\boldsymbol{\lambda_1}}\right)^2 \boldsymbol{e_i}\right\|_2}.$$

Et donc on peut conjecturer et démontrer très facilement par simple récurrence que :

$$\boldsymbol{Y_n = \mathbf{signe}(\lambda_1)^n \times \frac{\sum_{i=1}^{i=m} \omega_i (\frac{\lambda_i}{\lambda_1})^n e_i}{\left\| \sum_{i=1}^{i=m} \omega_i (\frac{\lambda_i}{\lambda_1})^n e_i \right\|_2}}$$

Alors, donc comme on a supposé que toutes les valeurs propres soient réelles. On aura $\frac{\lambda_1}{|\lambda_1|} = \frac{\overline{\lambda_1}}{|\lambda_1|}$

Donc $(\frac{\overline{\lambda_1}}{|\lambda_1|})^n \times Y_n = (\frac{\lambda_1}{|\lambda_1|})^n \times Y_n = \mathbf{signe}(\lambda_1)^{2n} \times \frac{\sum_{i=1}^{i=m} \omega_i (\frac{\lambda_i}{\lambda_1})^n e_i}{\left\| \sum_{i=1}^{i=m} \omega_i (\frac{\lambda_i}{\lambda_1})^n e_i \right\|} = \frac{\sum_{i=1}^{i=m} \omega_i (\frac{\lambda_i}{\lambda_1})^n e_i}{\left\| \sum_{i=1}^{i=m} \omega_i (\frac{\lambda_i}{\lambda_1})^n e_i \right\|}$

Or $(\forall\, i \neq 1), \left|\frac{\lambda_i}{\lambda_1}\right| < 1$ **et donc** $(\forall\, i \neq 1), \lim_{n \to +\infty} \left|\frac{\lambda_i}{\lambda_1}\right|^n = 0.$

De ce fait,

$$(\frac{\overline{\lambda_1}}{|\lambda_1|})^n \times Y_n = \frac{\omega_1 e_1 + \sum_{i=2}^{i=m} \omega_i (\frac{\lambda_i}{\lambda_1})^n e_i}{\left\| \omega_1 e_1 + \sum_{i=2}^{i=m} \omega_i (\frac{\lambda_i}{\lambda_1})^n e_i \right\|} \xrightarrow[n \to +\infty]{} \frac{\omega_1 e_1}{\|\omega_1 e_1\|} = signe(\omega_1) e_1$$

En effet, la même démonstration s'applique dans le cas complexe. (Dans ce cas il faudra juste laisser $\frac{\lambda_1}{|\lambda_1|}$ sans la remplacer par signe(λ_1)).

CQFD.

2- La rapidité de convergence

Un autre résultat fourni par la méthode de puissance concerne la vitesse de convergence. Lorsque les multiplicités algébriques et géométriques associées à la valeur propre λ_1 sont égales, le taux de convergence de l'algorithme se comporte en $O\left(\left(\frac{\lambda_2}{\lambda_1}\right)^n\right) = O((\lambda_2)^n)$, où λ_1 et λ_2 sont la plus grande et la seconde plus grande valeurs propres (en valeur absolue). La convergence est bien plus lente dans le cas contraire, et se comporte comme $O\left(\frac{1}{n}\right)$en général.

Ce qui veut dire que la vitesse de convergence de la méthode de la puissance dépend de la vitesse de convergence de la suite $\left(\frac{\lambda_2}{\lambda_1}\right)^n$ et donc de $\frac{\lambda_2}{\lambda_1} = \lambda_2$. Ce qui veut dire que, plus λ_2 est proche de 1, plus la convergence sera de plus en plus lente.

Or, selon le théorème de Perron-Frobenius. Le sous espace propre associé à la valeur propre 1 est de dimension 1. Donc on a égalité des multiplicités algébriques et géométriques. Ainsi, convergence de l'algorithme se comporte en $O\left(\left(\frac{\lambda_2}{\lambda_1}\right)^n\right) = O((\lambda_2)^n)$,

Reste à étudier le comportement de la suite $((\lambda_2)^n)_{n \in \mathbb{N}}$.

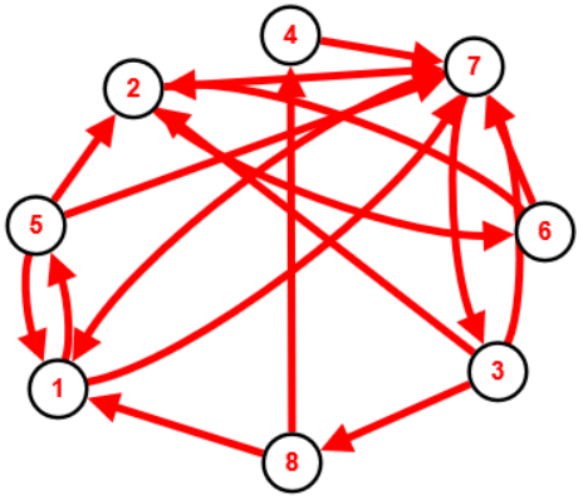
a) La méthode de la déflation

La méthode de déflation est une méthode qui permet, connaissant la valeur propre de plus grand module d'une matrice et un vecteur propre associé, de trouver la seconde valeur propre dont le module est le plus grand. Précisément, on part d'une matrice A d'ordre m dont les valeurs propres vérifient $(\forall \boldsymbol{i} \in [\![\mathbf{1}, \mathbf{m}]\!]), |\boldsymbol{\lambda_i}| \leq |\boldsymbol{\lambda_1}|$. Si on considère $\boldsymbol{\lambda_1}, \boldsymbol{v_1}$ respectivement la valeur propre dominante de la matrice $\mathbf{A}$ et un vecteur propre qui lui est associé, $\boldsymbol{\lambda'_1} = \boldsymbol{\lambda_1}, \mathbf{w_1}$ respectivement la valeur propre dominante de la matrice $\mathbf{A^t}$ et un vecteur propre qui lui est associé.

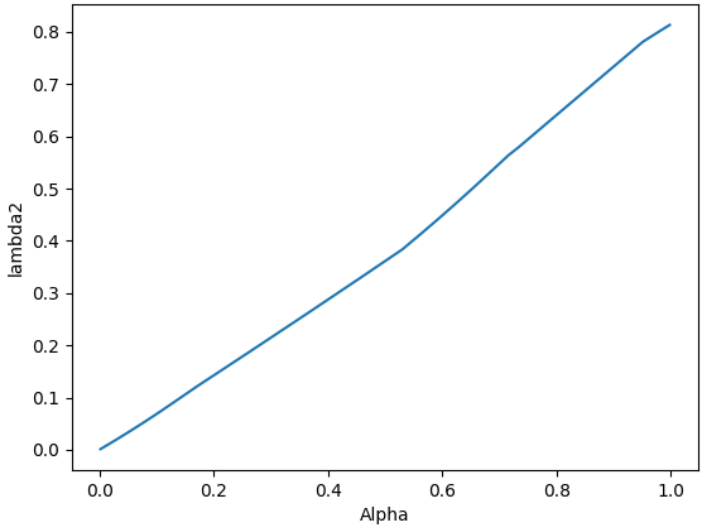
Si on pose $\boldsymbol{B = A - \lambda_1 \times \frac{{v_1}^t . w_1}{w_1 . {v_1}^t}}$. Alors la matrice B possède comme valeur propres $(\boldsymbol{\lambda_2 . \lambda_3 . . . . . \lambda_m . . 0})$! Ainsi en réappliquant la méthode de la puissance à la matrice B on obtiendra la 2ème valeur propre dominante. Plus $\boldsymbol{\lambda_2}$ est proche de 1, plus la convergence sera de plus en plus lente, et vice versa.

b) α : une valeur qui influe beaucoup la vitesse de convergence

On a remarqué que la 2ème valeur propre dominante a une très forte relation avec la probabilité α. En effet, après de nombreux calcul, on a pu conjecturer qu'il existe une certaine linéarité entre $\boldsymbol{\lambda_2}$ **et α.** Pour s'assurer de cela, on a considéré le web suivant :



Ensuite, on a calculé $\boldsymbol{\lambda_2}$ pour des valeurs différentes de α comprises bien sur entre 0 et 1. Le résultat obtenu est le suivant :



Ceci prouve bien la linéarité entre $\boldsymbol{\alpha}$ et $\boldsymbol{\lambda_2}$ du moins pour ce web particulier. Toutefois, après longue réflexion, on a pu conjecturer le coefficient de linéarité entre $\boldsymbol{\alpha}$ **et** $\boldsymbol{\lambda_2}$ **.** En effet, on a conjecturé que $\boldsymbol{\lambda_2 = \alpha\, s_2}$**,** avec $\boldsymbol{s_2}$ la deuxième valeur propre dominante de la matrice S défini par la relation suivante : $\boldsymbol{G_{ij} = \alpha \times S_{ij} + (1-\alpha) \times E_{ij}}$.

3- Les défauts du PageRank

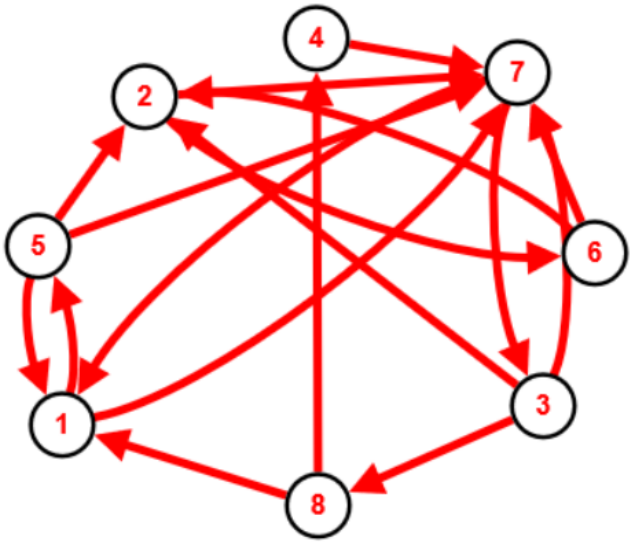
Cette méthode de PageRank de Google n'est pas finalement sans défauts. En effet, le calcul de la pertinence d'une page à travers cette approche néglige plusieurs points. Par exemple, la pertinence d'une page peut être une notion très relative : Une seule et même page pourra être plus pertinente pour une personne qu'une autre, et ce pourra être qu'une question de gout et de préférence de chacun.

A vrai dire, le vrai problème du PageRank de Google, est de considérer que la pertinence des pages du Web ne dépend que de sa structure interne. Quitte à dire que la pertinence des pages est indépendante des gouts de chacun.

IV. Application sur un mini web

Après le codage de la méthode de la puissance en utilisant Python. Ce sera intéressant de calculer le PageRank d'un quelconque web.

Par exemple, si on prend le Web W suivant :



Partons étape par étape :

Donnons la matrice d'adjacence Ω de ce Web. Il est bien clair que $\Omega \in \mathcal{M}_8(\mathbb{R})$.

$$\boldsymbol{\Omega} = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix}$$

Calculons la matrice G de Google, dont on rappelle la définition :

$$\forall \boldsymbol{i,j} \in (\mathbf{1}, \boldsymbol{N}), \boldsymbol{G_{ij}} = \boldsymbol{\alpha} \times \boldsymbol{S_{ij}} + (\mathbf{1} - \boldsymbol{\alpha}) \times \boldsymbol{E_{ij}} \text{ avec } \boldsymbol{E} = \begin{pmatrix} \frac{\mathbf{1}}{\boldsymbol{N}} & \cdots & \frac{\mathbf{1}}{\boldsymbol{N}} \\ \vdots & \ddots & \vdots \\ \frac{\mathbf{1}}{\boldsymbol{N}} & \cdots & \frac{\mathbf{1}}{\boldsymbol{N}} \end{pmatrix}$$

Ceci dit, la matrice G de Google correspondante à ce web W sera :

Avec α = 0.85, on trouve :

$$\boldsymbol{G} = \begin{pmatrix} 0.01875 & 0.01875 & 0.01875 & 0.01875 & 0.44375 & 0.01875 & 0.44375 & 0.01875 \\ 0.01875 & 0.01875 & 0.01875 & 0.01875 & 0.01875 & 0.44375 & 0.44375 & 0.01875 \\ 0.01875 & 0.30208 & 0.01875 & 0.01875 & 0.01875 & 0.01875 & 0.30208 & 0.30208 \\ 0.01875 & 0.01875 & 0.01875 & 0.01875 & 0.01875 & 0.01875 & 0.86874 & 0.01875 \\ 0.30208 & 0.30208 & 0.01875 & 0.01875 & 0.01875 & 0.01875 & 0.30208 & 0.01875 \\ 0.01875 & 0.44375 & 0.01875 & 0.01875 & 0.01875 & 0.01875 & 0.44375 & 0.01875 \\ 0.30208 & 0.01875 & 0.030208 & 0.30208 & 0.01875 & 0.01875 & 0.01875 & 0.01875 \\ 0.44375 & 0.01875 & 0.001875 & 0.44375 & 0.01875 & 0.01875 & 0.01875 & 0.01875 \end{pmatrix}$$

Ainsi après il ne reste qu'à appliquer la méthode de la puissance pour essayer d'extraire le vecteur PageRank μ qui satisfait le système suivant :

$$\begin{cases} \mu = \mu \times \boldsymbol{G}. \\ \|\mu\|_{\mathbf{1}} = \mathbf{1} \\ (\forall \boldsymbol{i} \in [\![\mathbf{1}, \boldsymbol{n}]\!]), \mu_{\boldsymbol{i}} \in [\mathbf{0}, \mathbf{1}] \end{cases}$$

En prenant comme vecteur initial $\boldsymbol{W^{(0)}} = [\mathbf{1} \quad \cdots \quad \mathbf{1}]$ et avec α = 0.85, le vecteur de probabilité μ sera :

$$\mu = [\mathbf{0.15308} \quad \mathbf{0.09905} \quad \mathbf{0.10832} \quad \mathbf{0.12933} \quad \mathbf{0.08381} \quad \mathbf{0.06084} \quad \mathbf{0.31613} \quad \mathbf{0.04944}]$$

Concernant la vitesse de convergence, il faut d'abord déterminer la 2ème valeur propre dominante correspondante à ce cas.

Appliquons la méthode de la déflation :

Construisons d'abord la matrice G. Il est bien clair que $(\boldsymbol{\lambda_1 = 1, et\ v_1 =}\ \mu)$. Reste à calculer $\mathbf{w_1}$, un vecteur propre associé à $\boldsymbol{\lambda'_1} = 1$ la valeur propre dominante de $\mathbf{G^t}$,.

Après avoir effectué le calcul, on trouve que

$$\mathbf{w_1} = [\mathbf{0.125} \quad \mathbf{0.125} \quad \mathbf{0.125} \quad \mathbf{0.125} \quad \mathbf{0.125} \quad \mathbf{0.125} \quad \mathbf{0.125} \quad \mathbf{0.125}]$$

Après calcul de la matrice B, on applique de nouveau la méthode la puissance. Cette dernière nous donne comme valeur pour $\boldsymbol{\lambda_2 = 0.687.}$

Ainsi, d'après le théorème de la méthode de puissance, la convergence se comporte en $\boldsymbol{O\left(\left(\frac{\lambda_2}{\lambda_1}\right)^n\right) = O((\lambda_2)^n) = O((0.687)^n).}$

Ainsi, pour une précision de $\mathbf{10^{-10}}$ près, i.e. $|(\boldsymbol{\lambda_2})^n - \mathbf{0}| \leq \mathbf{10^{-10}}$**.** Donc, il faut au moins $\mathbf{n = E(\frac{\ln 10^{-10}}{\ln \lambda_2})}$**.** Donc il faut au moins presque **n = 62** itérations, pour atteindre une telle précision.

IV. CREATION D'UN MOTEUR DE RECHERCHE

Après avoir codé l'algorithme du PageRank, nous nous sommes intéressés à comment l'appliquer sur un web réel, c'est-à-dire, comment indexer de vrais sites web. Ainsi, nous avons essayé de créer un moteur de recherche.

1- Création de Crawler

Le premier défi rencontré pour la création de ce moteur de recherche a été de récupérer les différents liens des sites web en partant d'une seule url.

Suite à diverses recherches, nous avons pu trouver une méthode de crawling d'url adaptée à nos besoins. Nous y avons ensuite ajouté quelques modifications afin de non pas afficher les divers liens récupérés mais de créer une liste de liste à deux éléments contenant le lien de départ et le lien d'arrivée. Ci-dessous, on peut voir le code écrit à cet effet.

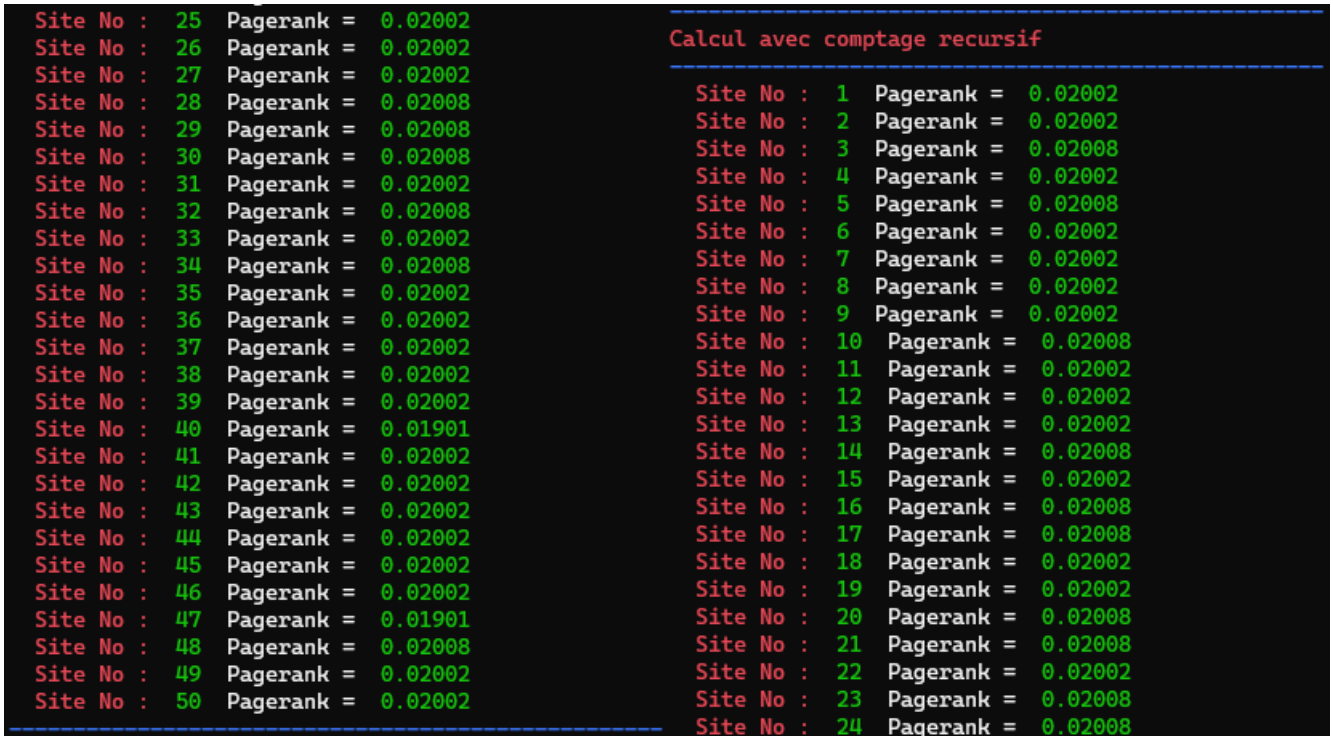
```python
import requests
import re
from urllib.parse import urlparse


class PyCrawler(object):

    #définition de l'objet, le lien de départ et les liens visités
    def __init__(self, starting_url):
        self.starting_url = starting_url
        self.visited = set()

    #fonction qui permet de récupérer le contenu html
    def get_html(self, url):
        try:
            html = requests.get(url)
        except Exception as e:
            print(e)
            return ""
        return html.content.decode('latin-1')

    #fonction qui permet d'extraire les liens du contenu html
    def get_links(self, url):
        html = self.get_html(url)
        parsed = urlparse(url)
        base = f"{parsed.scheme}://{parsed.netloc}"
        links = re.findall('''<a\s+(?:[^>]*?\s+)?href="([^"]*)"''', html)
        for i, link in enumerate(links):
            if not urlparse(link).netloc:
                link_with_base = base + link
                links[i] = link_with_base

        return set(filter(lambda x: 'mailto' not in x, links))


    #fonction qui crawl les pages en ajoutant les liens dans une file
    def crawl(self, url):
        file = []
        file += list(self.get_links(url))
        items = []
        while True:
            if len(items)  == 25:
                break
            link =  file[0]
            if link in self.visited:
                file.pop(0)
                continue
            items += [ [self.starting_url,link] ]
            self.visited.add(link)
            file.pop(0)
            file += list(self.get_links(link))

        return items

    #fonction qui lance le crawling
    def start(self):
        item = []
        item = self.crawl(self.starting_url)
        return item
```

Grâce à ce code, nous avons donc pu extraire les liens des diverses pages.

2- Calcul du PageRank à partir d'une petite partie du Web Internet

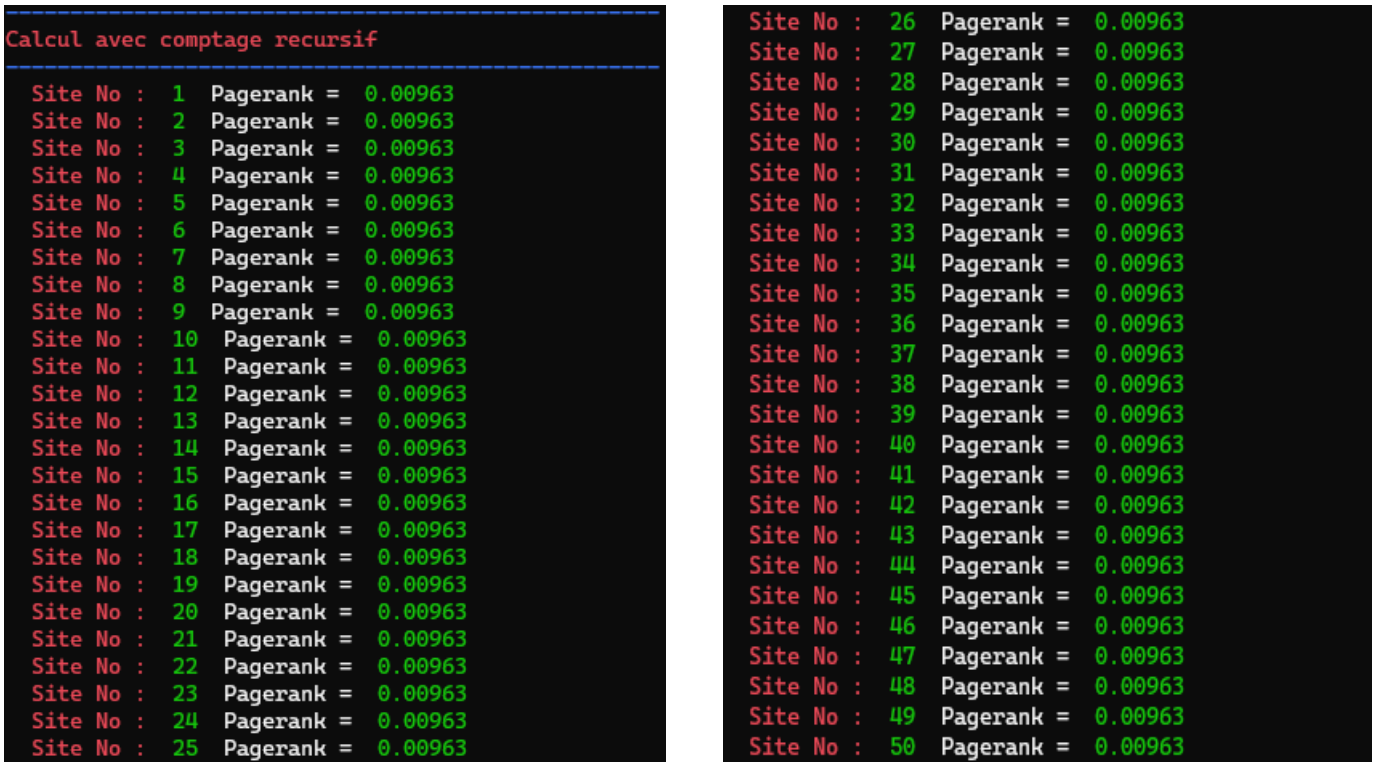
Notre mini moteur de recherche prêt, nous avons donc décider de le tester pour différents sites. Cependant, on a remarqué un problème.

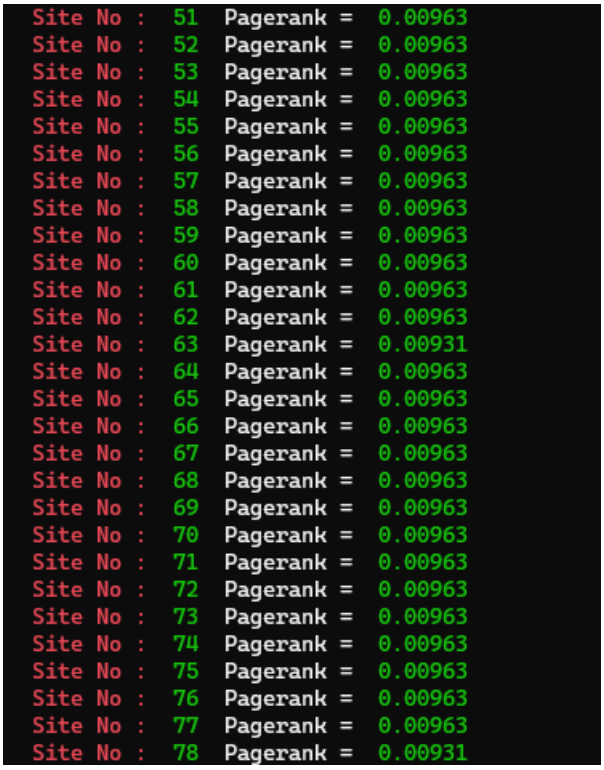
En effet, en partant de trois pages différentes, dans notre cas il s'agit de https://www.solocal.com/, https://www.frandroid.com/ et de https://www.data-transitionnumerique.com/web-scraping-python/ nous avons obtenu pour cinquante pages au total, donc environ 17 pages pour chaque domaine, les résultats suivants :

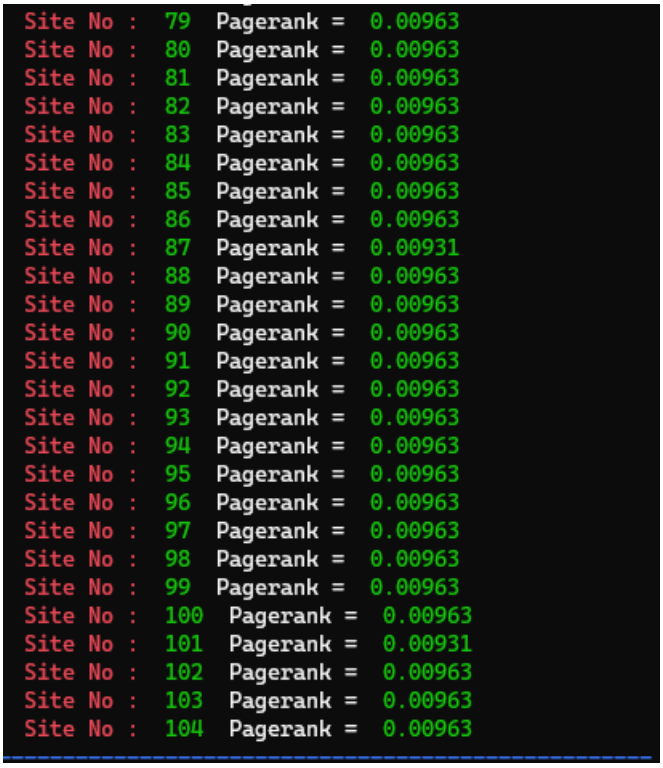


On peut identifier trois différents coefficients principaux dans les valeurs des indices de pertinence du PageRank.

Cependant, en rajoutant le domaine https://www.insa-toulouse.fr/fr/index.html et en augmentant le nombre d'url récupérés à 104, on obtient les résultats suivants :







On remarque alors qu'il ne subsiste que deux valeurs au sein du PageRank tandis qu'on s'attendait plutôt à quatre valeurs.

En augmentant le nombre d'url sans changer le nombre de domaines explorer, on tend à avoir une unique valeur pour le PageRank. Après réflexion, nous avons conclu que le résultat est ainsi parce que nous n'explorons pas le web tout entier et donc on tend à avoir des proportions identiques pour les différents domaines ce qui explique le PageRank obtenu.

Conclusion

Google, un mot que tout monde connait, des plus jeunes aux plus âgés. Aller sur Google est devenu la réponse à la majorité des problèmes qu'on rencontre dans notre vie de tous les jours. Certains considèrent même Google comme ayant réponse à tout. Curieux comme nous sommes, nous avons cherché à comprendre ce qui faisait la popularité de Google. Dans nos recherches, nous avons découvert l'algorithme du PageRank.

L'algorithme du PageRank est l'algorithme qui a permis à Google de se distinguer des autres moteurs de recherche. En effet, utilisé jusqu'en 2016 par Google, il lui a permis d'être le premier moteur de recherche mondial depuis les années 1998.

En partant d'une approche toute simple, nous avons pu aboutir à une méthode plus complète mettant en œuvre la matrice G (matrice Google), la récursivité et d'autres notions assez avancées. Très peu connu, ou souvent connu de nom mais pas de procédé, l'algorithme du PageRank n'est pourtant pas très complexe. Le vecteur PageRank est obtenu après un calcul par la méthode la puissance appliquée à une matrice dont les coefficients représentent la probabilité d'aller d'une page donnée à une autre bien précise.

Ainsi, nous avons pu donner une approche de calcul théorique et numérique du PageRank. Nous avons pu le mettre en œuvre et avons obtenu quelques résultats. Cependant, les résultats numériques restent encore assez vagues. Cela laisse penser que le PageRank connu du grand public et que nous avons essayé d'approcher n'est pas complet. Il resterait donc un ingrédient secret dont seul Google a la recette. Cela explique pourquoi depuis deux décennies, Google est resté indétrônable en tant que moteur de recherche.

L'algorithme du PageRank, recette personnelle de Google reste donc encore assez mystérieuse dans son application. Nous avons pu dévoiler une partie de cette recette, mais une partie reste encore mystérieuse. Pourrions-nous la connaître dans son entièreté un jour ? Peut-être en travaillant chez Google… En attendant, Google reste bien le meilleur moteur de recherche.

INSA Toulouse
135, avenue de Rangueil
31077 Toulouse Cedex 4 - France
www.insa-toulouse.fr

Université Fédérale Toulouse Midi-Pyrénées

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE
DE L'ÉDUCATION NATIONALE,
DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE