Agrégation externe de mathématiques, session 2008
Épreuve de modélisation, option B : Calcul Scientifique

(*public 2008*)

Résumé : On s'intéresse à des questions de valeurs propres qui interviennent de façon cruciale dans le fonctionnement des moteurs de recherche sur Internet

Mots clefs : Valeurs propres et vecteurs propres de matrice, systèmes linéaires

---

► *Il est rappelé que le jury n'exige pas une compréhension exhaustive du texte. Vous êtes laissé(e) libre d'organiser votre discussion comme vous l'entendez. Des suggestions de développement, largement indépendantes les unes des autres, vous sont proposées en fin de texte. Vous n'êtes pas tenu(e) de les suivre. Il vous est conseillé de mettre en lumière vos connaissances à partir du fil conducteur constitué par le texte. Le jury appréciera que la discussion soit accompagnée d'exemples traités sur ordinateur.*

La recherche d'informations pertinentes sur le Web est un des problèmes les plus cruciaux pour l'utilisation de de ce dernier. Des enjeux économiques colossaux sont en jeu, et diverses multinationales se livrent à de grandes manœuvres. Le leader actuel de ce marché, Google, utilise pour déterminer la pertinence des références fournies, un certain nombre d'algorithmes dont certains sont des secrets industriels jalousement gardés, mais d'autres sont publics. On va s'intéresser ici à l'algorithme PageRank, lequel fait intervenir des valeurs propres et vecteurs propres d'une énorme matrice.

1. La matrice de Google

À un moment donné, on peut considérer que le Web est une collection de $N \in \mathbb{N}$ 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 *score de pertinence* à chaque page.

On se donne donc un ordre arbitraire sur l'ensemble des pages que l'on numérote ainsi de $i = 1$ à $i = N$. La structure de connectivité du Web peut alors être représentée par une matrice C de taille $N \times N$ telle que $c_{ij} = 1$ si la page j pointe sur la page i, $c_{ij} = 0$ sinon. 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 $q_{ij} = c_{ij}/N_j$ si $N_j \neq 0$, $q_{ij} = 0$ sinon. 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 $r \in \mathbb{R}^N$ des scores des pages de la forme

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

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 $e = {}^t\begin{pmatrix}1 & 1 & \cdots & 1\end{pmatrix} \in \mathbb{R}^N$ et $d \in \mathbb{R}^N$ tel que $d_j = 1$ si $N_j = 0$, $d_j = 0$ sinon. La matrice

$$P = Q + \frac{1}{N} e\, {}^t d \tag{2}$$

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 ${}^t eP = {}^t e$, 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

$$A = \alpha P + (1-\alpha)\frac{1}{N} e\, {}^t e. \tag{3}$$

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 (il s'agit du théorème de Perron-Frobenius). Finalement, PageRank calcule un tel vecteur propre $r \in \mathbb{R}^N$, normalisé d'une façon ou d'une autre,

$$r = Ar, \tag{4}$$

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.

2. La méthode de la puissance

Étant donnée la taille de la matrice A, la seule méthode numérique envisageable pour calculer le vecteur r n'est autre que la méthode de la puissance. Rappelons-en rapidement le principe.

On part d'un vecteur initial $r^0 \neq 0$, et l'on procède à l'itération

$$q^k = Ar^{k-1}, \quad r^k = \frac{q^k}{\|q^k\|_1}, \tag{5}$$

avec $\|v\|_1 = \sum_{i=1}^N |v_i|$. Si l'on choisit le premier vecteur r^0 à composantes positives, par exemple $r^0 = e$, cette méthode converge vers le vecteur propre recherché r. La vitesse de convergence est de l'ordre de $|\lambda_2|^k$ où λ_2 est la deuxième valeur propre de A, celles-ci étant classées par ordre de module décroissant.

En fait, on peut assez facilement montrer que $|\lambda_2| = \alpha$, ce qui donne un contrôle explicite sur la vitesse de convergence. En effet, soient $\{1, \mu_2, \mu_3, \ldots, \mu_N\}$ les valeurs propres de P. On pose $\hat{e} = \frac{1}{\sqrt{N}} e$ et on choisit une matrice U_1 de taille $N \times (N-1)$ telle que la matrice $U = \begin{pmatrix} \hat{e} & U_1 \end{pmatrix}$ soit orthogonale. Il vient

$$^tUPU = \begin{pmatrix} 1 & 0 \\ w & T \end{pmatrix},$$

avec $T = {}^tU_1 {}^tPU_1$. Manifestement, les valeurs propres de T sont $\{\mu_2, \mu_3, \ldots, \mu_N\}$. Un calcul analogue montre que

$$^tUAU = \begin{pmatrix} 1 & 0 \\ \alpha w & \alpha T \end{pmatrix},$$

ce qui montre que les valeurs propres de A sont $\{1, \alpha\mu_2, \alpha\mu_3, \ldots, \alpha\mu_N\}$. En particulier, comme 1 est la valeur propre de plus grand module de P, on voit que le module de la seconde valeur propre de A est inférieur à α.

3. Aspects algorithmiques

La difficulté de la méthode de la puissance dans le cas de la matrice A vient de sa taille gigantesque. La matrice de départ Q est une matrice très creuse pour laquelle des produits matrice-vecteur sont envisageables en pratique. Par contre, la matrice A est pleine, elle contient de l'ordre de 10^{20} éléments et il est hors de question de l'assembler explicitement et encore moins de l'utiliser pour calculer des produits matrice-vecteur !

Tout d'abord, on remarque que le vecteur q^2 est tel que $\|q^2\|_1 = {}^teq^2 = 1$. À partir de cette étape, la normalisation devient donc inutile. On peut d'ailleurs décider de partir d'un vecteur r^0 tel que $\|r^0\|_1 = 1$.

Par ailleurs, on voit que $y = Az$ peut s'écrire sous la forme $y = \alpha Qz + \frac{\beta}{N} e$. Si $\|z\|_1 = 1$, on a bien sûr $\|y\|_1 = 1$, d'où $\beta = 1 - \|\alpha Qz\|_1$. On a ainsi ramené le calcul du produit matrice pleine-vecteur original à un produit matrice creuse-vecteur et à une évaluation de norme, effectivement calculables à l'échelle du Web (à condition de disposer de ressources informatiques conséquentes, quand même).

On aboutit de la sorte à un algorithme simple

Choisir r

Tant que $s \geq$ tolérance, **faire**

$\hat{r} = \alpha Qr$

$\beta = 1 - \|\hat{r}\|_1$
$\tilde{r} = \hat{r} + \frac{\beta}{N} e$
$s = \|\tilde{r} - r\|_1$
$r = \tilde{r}$

Retourner r

Une valeur typique de α est $\alpha = 0.85$. Pour satisfaire une tolérance de 10^{-4}, suivant l'ordre de grandeur de la constante dans l'estimation de la vitesse de convergence, on peut estimer qu'il faut quelques dizaines itérations au plus.

4. Aspect système linéaire

Transformer un problème de calcul de vecteur propre (4) en une résolution de système linéaire est en règle générale une mauvaise idée. Dans notre cas, comme nous savons par ailleurs que la valeur propre qui nous intéresse vaut 1 et que le vecteur propre r qui nous intéresse a toutes ses composantes positives et est de norme 1, on voit qu'il est solution du système linéaire

$$(I - \alpha P) r = \frac{1-\alpha}{N} e. \tag{6}$$

Or la matrice $I - \alpha P$ est inversible, donc r est l'unique solution du système (6) ! Il y a plus : $I - \alpha P$ est une M-matrice, c'est-à-dire que les coefficients de son inverse sont tous positifs. En fait, notant $|||M|||_1 = \max\{\|Mv\|_1 ; \|v\|_1 = 1\} = \max_j \sum_i |M_{ij}|$ la norme subordonnée d'une matrice M, on montre que

$$|||I - \alpha P|||_1 = 1 + \alpha, \qquad |||(I - \alpha P)^{-1}|||_1 = \frac{1}{1-\alpha}.$$

Par conséquent, le conditionnement de $I - \alpha P$ pour cette norme vaut

$$\mathrm{cond}_1(I - \alpha P) = \frac{1+\alpha}{1-\alpha}.$$

En particulier, il ne dépend pas de la taille ni de la structure de connectivité du Web. On voit aussi que ce conditionnement se dégrade quand $\alpha \to 1$. On en déduit que si r' est le vecteur des scores d'un Web de taille N de matrice P', on a

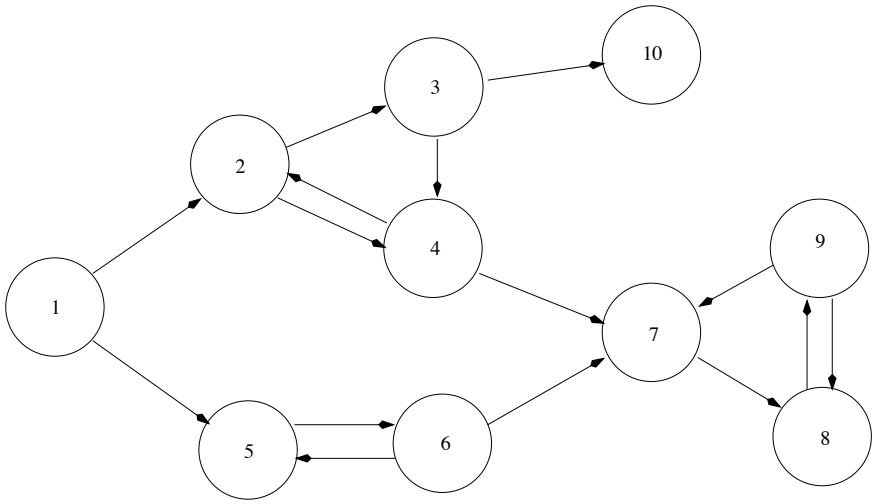
$$\|r - r'\|_1 \leq \frac{\alpha}{1-\alpha} |||P - P'|||_1.$$

En d'autres termes, si on modifie les liens de telle sorte à ne pas trop perturber la matrice P associée, on ne modifie pas trop le vecteur des rangs, et ce qui est plus important, l'ordre de ses composantes quand α n'est pas proche de 1.

5. Simulation numérique à petite échelle

Comme on ne dispose pas de la puissance de calcul de Google — dont il est un secret de polichinelle qu'elle est colossale — on va effectuer des tests de calcul sur un Web-jouet d'une dizaine de pages.

On se donne notre petit Web sous la forme d'un graphe dont les sommets sont les pages et les arêtes les liens hypertextes. Un dessin valant mieux qu'un long discours, les flèches indiquent les liens entre pages :



On peut alors écrire les matrices C, Q, P et A, mettre en œuvre la méthode de la puissance, puis vérifier que l'on a obtenu le bon vecteur propre.

Suggestions pour le développement

► *Soulignons qu'il s'agit d'un menu à la carte et que vous pouvez choisir d'étudier certains points, pas tous, pas nécessairement dans l'ordre, et de façon plus ou moins fouillée. Vous pouvez aussi vous poser d'autres questions que celles indiquées plus bas. Il est très vivement souhaité que vos investigations comportent une partie traitée sur ordinateur et, si possible, des représentations graphiques de vos résultats.*

- Pourquoi la matrice Q contient-elle en général dans le cas de la matrice du Web des colonnes nulles et pourquoi n'admet-elle probablement pas la valeur propre 1 ?
- Pourquoi la matrice P admet-elle en général dans le cas de la matrice du Web 1 comme valeur propre multiple ?
- On interprète souvent la constante $1-\alpha$ comme la probabilité pour un internaute de zapper sur une autre page du Web au hasard. Qu'en pensez-vous ?
- À ce propos, comment interpréteriez-vous une correction de la forme $\alpha P + (1-\alpha)\frac{1}{N} p\, {}^t e$ avec $p_i > 0$, $\|p\|_1 = N$ et $p \neq e$, à la place de (2) ?
- Vous pouvez chercher à justifier les assertions de la section 4.
- L'approche de cette section vous paraît-elle raisonnable du point de vue numérique ?