Maximisation du PageRank

Jean-Philippe Chancelier

May 21, 2017

Contents

1 Une chaîne de Markov

On se donne un graphe qui représente les liens entre des pages web sur un ensemble de pages données. Les noeuds du graphe sont les pages web et la présence d'un arc du noeud i au noeud j indique que la page j est référencée par la page i. Soit n le nombre total de pages, le graphe est donné par sa matrice d'adjacence `Adj` de taille `nxn` où `Adj(i,j)==1` s'il existe un lien de la page i vers la page j et `Adj(i,j)==0` sinon.

Le degré d'un noeud $deg(i)$ est le nombre d'acs partant de ce noeud.

En divisant chaque ligne de la matrice `Adj` par son degré on obtient une matrice sous stochastique, Pss ou la somme en ligne vaut 0 pour les noeuds de degré nul et vaut 1 pour les autres noeuds.

On se donne aussi un vecteur ligne, z de taille n, dont les composantes sont strictement positives et dont la somme des composantes vaut 1.

On appelle P_1 la matrice stochastique obtenue à partir de Pss en remplaçant les lignes nulles par le vecteur ligne z. On appelle P la matrice stochatique $\alpha P_1 + (1-\alpha)ez$ où α est un réel donné $0 < \alpha < 1$ et e est le vecteur colonne de taille n dont toutes les composantes valent 1.

On considère une chaîne de Markov de matrice de transition P.

Question 1 *Décrire le comportement de la chaîne $(X_n)_{n\in\mathbb{N}}$. Pourquoi le vecteur z est appellé vecteur de téléportation.*

2 Calcul du PageRank des états de la chaîne

Le PageRank des états de la chaîne de markov (qui sont les noeuds de notre graphe de pages web) est donné par la mesure invariante π associée à la chaîne.

Question 2 *Que peut-on dire de π ? existence ? unicité ?*

Construction et visualisation de matrices d'adjacences *Adj* aléatoires. Compléter le squelette de programme qui suit:

```
n=10; // Nombre de pages

function show_adj(Adj,diameters)
  [lhs,rhs]=argn(0);
  if rhs < 2 then diameters = 30*ones(1,n);end
  graph = mat_2_graph(sparse(Adj),1,'node-node');
  graph('node_x')=300*cos(2*%pi*(1:n)/(n+1));
  graph('node_y')=300*sin(2*%pi*(1:n)/(n+1));
  graph('node_name')=string([1:n]);
  graph('node_diam')= diameters;
  //graph('node_color')= 1:n;
  //show_graph(graph);
  rep=[1 1 1 1 2 2 2 2 2 2 2 2 2];
  plot_graph(graph,rep);
endfunction

Adj=grand(n,n,'bin',1,0.2);show_adj(Adj);

// Construction de la matrice de transition P
// associée à une matrice d'adjacence.
// Pss: transition d'origine,
// P: matrice de google
// z: vecteur de teleportation
// d: vecteur vaut 1 si le degré vaut zero et 0 sinon

function [P,Pss,Pprim,d,z,alpha]=google(Adj)
  // <A completer>
endfunction

[P,Pss,Pprim,d,z,alpha]=google(Adj);
// verification que P est stochastique

sum(P,'c')
```

Question 3 *La matrice P est stochastique mais présente le défaut d'être une matrice pleine alors que la matrice d'origine Pss était creuse. Montrer que le calcul de $P' * x$ peut se faire en utilisant Pss', d et z en préservant le caractère creux des opérations.*

```
x=rand(n,1)
y1=P'*x;
y2=alpha*Pss'*x + ...
y1-y2
```

Question 4 *Calculer le PageRank des pages du graphes en utilisant la fonction* `spec` *de scicoslab. On redessine le graphe en tenant compte du PageRank pour choisir le diamètre du cercle représentant chaque noeud.*

```
... = spec(...)
pi = ...
xbasc();show_adj(Adj,int(300*pi));
```

Question 5 *Calculer π en utilisant la suite $p_{k+1} = P' * p_k$.*

```
function [pi]=pi_iterative()
  p=ones(n,1);
  while %t
    ....
    if norm(pn-p,%inf) < 10*%eps then break;end
  end
  pi= ...
endfunction

pi=pi_iterative();
clean(pi*P - pi)
```

Question 6 *Calculer π en utilisant la suite précédente mais en conservant Pss', d et z.*

```
function [pi]=pi_iterative_sparse()
  p=ones(n,1);
  ....
  pi=
endfunction

pi=pi_iterative_sparse();
clean(pi*P - pi)
```

3 Maximisation discrète du PageRank

On considère maintetant les m premiers états de la chaîne avec $m = n/2$ et on cherche à maximiser le PageRank des m-premières pages.

$$\sum_{i=0}^{m} \pi_i \tag{1}$$

On suppose qu'on a accès aux p-premières pages ($p \leq m$) et qu'on peut changer leurs liens vers les pages $m+1,\ldots,n$. On peut donc choisir de changer la sous matrice `Adj(1:p,m+1:$)` de la matrice `Adj`

Question 7 *Écrire un programme d'optimisation discret qui résoud le problème de l'optimisation du pagerank (on choisira $p = 2$ et $m = n/2$).*

On cherche maintenant d'autres méthodes pour réaliser cette otpimisation de façon plus efficace.

4 Problèmes ergodiques

On se donne un chaîne de Markov de matrice de transition P et une fonction coût $r(x)$ on cherche à vérifier le théorème ergodique pour une matrice P irréductible.

$$\lim_{T\to+\infty} \frac{1}{T}\mathbb{E}\left[\sum_{t=0}^{T-1} r(X_t)\right] = \sum_x \pi(x)r(x) \tag{2}$$

```
function y=r(x)
  y=x^2
endfunction

n=4;
P=rand(n,n)
pr=sum(P,'c');
P = P ./ (pr*ones(1,n));

// on suppose ici que les etats de la chaine sont 1:n
```

Question 8 *Écrire* `ergodique_markov_T(T,P)` *qui calcule*

$$\frac{1}{T}\mathbb{E}\left[\sum_{t=0}^{T-1} r(X_t)\right] \tag{3}$$

puis `ergodique_markov(P)` *qui calcule* $\sum_x \pi(x)r(x)$ *et comparez.*

```
function cerg=ergodique_markov_T(T,P)
  ...
endfunction

function [cerg,pi]=ergodique_markov(P)
  ...
endfunction

// test
T=100000; CT=ergodique_markov_T(T,P);
[c,pi]=ergodique_markov(P);

c-CT
```

On veut vérifier maintenant que $c = \sum_x \pi(x) r(x)$ s'obtient aussi en résolvant le système linéaire:

$$(P - I) * w + R - c_1 = 0$$
$$(P - I) * c_1 = 0\,.$$

On va vérifier en effet que c_1 est un vecteur constant dont les composantes valent c.

On verifie que $(P - I) * c = 0$ impose à c d'être un vecteur constant.

```
// Le noyau de P-I est engendré par ones(n,1)
[x0,K]=linsolve(P- eye(n,n),zeros(n,1));
```

Question 9 *Suivre les calculs suivants dans scicoslab et conclure.*

```
// le projecteur spectral sur Espace propre associé a 1
Pr = ones(n,1)*pi; // [pi;pi;pi;....]
A = P-eye(n,n);    // A -Id
S = Pr - inv(Pr-A) // Pr-A est inversible
// vérifier que S*Pr et Pr*S sont nuls
clean(S*Pr)
clean(Pr*S)
// A*w  + R - c= 0
// A*c         = 0
R=r((1:n)');
// vérifions que w=-S*R et c=Pr*R sont solution du systeme linaire
w= -S*R;
c= Pr*R;
A*w + R -c
A*c
// Noter que w n'est pas unique, on peut rajouter à w les elts du noyau de A
```

```
// Montrons inversement que c doit être egal à Pr*R
// Pr*A est nul
Pr*A
// on doit donc avoir
// Pr*R - Pr*c = 0 et A*c =0
// en sommant
// Pr*R = (Pr-A)*c
// c = (Pr-A)^-1 *Pr*R
// c = (Pr-S)*Pr*R = Pr*Pr*R -S*Pr*R = Pr*R
// car Pr est un projecteur Pr^2 = Pr et S*Pr = 0
clean(Pr^2-Pr)
clean(S*Pr)
// conclusion c doit valoir Pr*R
// on le vérifie avec linsolve

[x0,K]=linsolve([A,-eye(n,n);zeros(n,n),A],[R;zeros(n,1)]);
// on vérifie bien que e = Pr*R
```

On peut donc résoudre le problème ergodique précédent en cherchant un couple (w, c) solution de

```
 w + c*ones(n,1) = Pw + R,
```

la composante c scalaire donnera la solution du problème ergodique.

On regarde un cas particulier ou la fonction coût pour un état i est donnée par

$$R(i) = \sum_{j=1}^{n} P_{i,j} Rm_{i,j} \tag{4}$$

où P est la matrice de transition de la chaine et Rm est une matrice `nxn` donnée. On suppose aussi que la matrice de transtion à la forme particulière d'une matrice Google.

$$P = \alpha * P_1 + (1 - \alpha) * e * z \tag{5}$$

```
P1=rand(n,n)
pr=sum(P1,'c');
P1 = P1 ./ (pr*ones(1,n));

z=grand(1,n,'unf',0,1);
z=z/sum(z);

alpha = 0.8;

P = alpha*P1 + (1-alpha)*ones(n,1)*z;
```

```
// les couts Rm(i,j)
Rm = grand(n,n,'unf',0,1);
```

Question 10 *Montrer que si w est un points fixe de l'opérateur $w \to \alpha P_1 w + b$ où $b_i = \sum_j P_{i,j} Rm_{i,j}$ alors (w, c) avec $c = (1-\alpha) * z * w$ est solution de $w + c = Pw + R$.*

```
// On le vérifie numeriquement
// trouver la solution de
// w = alpha*P1*w + sum(P.*Rm,'c')

w = ...

// calcul de c
c = (1-alpha)*z*w

// (w,c) solution du pb ergodique ?

w + c - (P*w + sum(P.*R,'c'))

// Maintenant on peut utiliser une méthode itérative
```

Question 11 *Que peut-on dire de l'opérateur $w \to \alpha P_1 w + b$? en déduire une méthode numérique pour calculer son point fixe.*

```
function w=iterative_c(tol)
  ...
endfunction

w=iterative_c(10*%eps);
// calcul de c
c = (1-alpha)*z*w

// (w,c) solution du pb ergodique ?

w + c - (P*w + sum(P.*R,'c'))
```

5 Problèmes ergodiques et maximisation du PageRank

Soit P^ν la matrice de transition obtenue pour un choix ν de redirections de liens au sein de la matrice d'adjacence de graphe. Si on regarde un problème ergodique avec une fonction coût $r(x) = \sum_j P^\nu_{i,j} Rm_{i,j}$ on sait que le coût ergidique devient

$$\sum_{i,j} \pi_i^\nu P_{i,j}^\nu Rm_{i,j}\,. \tag{6}$$

Pour un sous ensemble I de noeuds fixés et le choix $Rm_{i,j} = \xi_I(i)$ (où $\xi_I(i)$ vaut 1 si $i \in I$ et 0 sinon) le coût devient $\sum_{i\in I} \pi_i^\nu$.

Le choix $Rm_{i,j} = \xi_I(i)$ permet d'obtenir un coût ergodique qui est le pagerank de I et on a vu dans la section précédente que le calcul du coût ergodique s'obtient aussi à partir de la solution de

$$w_x = \alpha \sum_j Pprim_{x,j}^\nu w_j + \sum_j P_{x,j}^\nu Rm_{x,j} \tag{7}$$

Si on cherche à maximiser le PageRank on va chercher à résoudre

$$w_x = \sup_\nu \alpha \sum_j Pprim_{x,j}^\nu w_j + \sum_j P_{x,j}^\nu Rm_{x,j} \quad \forall x = 1,\ldots,n \tag{8}$$

On suppose qu'on a accès aux m-premières pages et qu'on peut changer leurs liens vers les pages $m+1$,...,n. On peut donc choisir de changer la sous matrice `Adj(1:p,m+1:$)` de la matrice `Adj`.

On va implémenter un algorithme d'iterations sur les valeurs pour touver une solution à l'équation (9).

- on fixe un w^0 initial
- à l'étape k, on calcule le contrôle, ν^k qui maximise le membre droit de l'équation (9) quand w est fixé à w^k.
- à l'étape k pour le controle fixé ν^k on cherche w^{k+1} solution de l'équation

$$w_x^{k+1} = \alpha \sum_j Pprim_{x,j}^{\nu^k} w_j^{k+1} + \sum_j P_{x,j}^{\nu^k} Rm_{x,j} \quad \forall x = 1,\ldots,n \tag{9}$$

- On s'arrête quand l'écart entre w^{k+1} et w^k devient sous une tolérance fixée.

Question 12 *Implémenter l'algorithme précédent.*