Algèbre linéaire & matricielle

INSA 3BIM
Automne 2014

Samuel Bernard
bernard@math.univ-lyon1.fr
http://math.univ-lyon1.fr/~bernard/numalg.html

Plan de cours

- Rappel
  - espaces vectoriels (base, dim)
  - applications linéaires (rang, det, ker)
- Analyse matricielle
  - valeurs propres
  - espaces caractéristiques
  - polynômes d'endomorphismes (Thm Caley-Hamilton)
- Factorisation et moindres carrés
  - décomposition de Jordan
  - pseudo-inverse
  - QR: Gram-Schmidt and Householder
  - méthodes itératives

Evaluation

- Un devoir à la maison (30%*)
- Un final (2h – dernière séance) (50%*)
- Un quiz (30 min – quand?) (20%*)

- *Pondération susceptitble d'être modifié

Pourquoi l'algèbre linéaire

- *N* équipes
- On connaît le résultats des matches entre équipes *i* et *j*: a_{ij}
- le nombre de matches et les adversaires ne sont pas nécessairement les mêmes pour toutes les équipes: n_i est le nombre de parties jouées par l'équipe *i*.
- Comment classer les équipes?

Exercice

- Etant donné les résultats aij entre équipe *i* et *j*, avec a_{ij} = nombre de victoires de *i* sur *j* + ½ nombre de matches nuls, et le nombre total de matches joués ni par l'équipe *i*, proposer un classement des équipes

Python

```
from numpy import *
import scipy as Sci
import scipy.linalg

N = 20 # nombre d'équipes
GPD = N/2 # nombre de matches par jour
ND = 12   # nombre de jours
team = range(0,N) # numérotation des équipes
A = zeros((N,N))  # initialisation de la matrice
for k in range(0,ND): # boucle de génération des résultats
    p = team
    random.shuffle(p) # permutaion aléatoire – choix des adversaires
    for i in range(0,GPD):
        score = random.random()
        if score>0.5: # équipe A gagne
            A[p[i],p[i+GPD]] = A[p[i],p[i+GPD]] + 1
        else:  # équipe B gagne
            A[p[i+GPD],p[i]] = A[p[i+GPD],p[i]] + 1
```

Python

```
A = A/ND # normalisation des résultats

r0 = ones((N,1)) # classement par défaut
r1score = dot(A,r0) # A*r0 quelle interprétation?
r2score = dot(A,dot(A,r0)) # A*A*r0 quelle interprétation?
r1rank = r1score.argsort(axis=0)
r2rank = r2score.argsort(axis=0)
```

- On définit le score de l'équipe *i*:

$$s_i = \frac{1}{n_i} \sum_{j=1}^{N} a_{ij} r_j$$

- r_j >= 0 est une valeur de classement (force) de l'équipe *j* (inconnue *a priori*)
- a_{ij} positif ou nul
- On suppose que *r* est proportionnel au score *s*

$$Ar = \lambda r$$

- A est la matrice avec entrées a_{ij}/n_i

- On cherche *r* non-négatif (r_i >= pour tout *i*) avec constante de proportionnalité lambda positive
- r est un vecteur propre de A, lambda est une valeur propre
- Est-ce que *r* existe?
- Est-ce qu'il est unique?

Théorème de Perron-Frobenius

Si la **matrice** *A* est non-négative, alors il existe un **vecteur propre** *r* non-négatif associé à une **valeur propre** positive λ. De plus, si la matrice *A* est **irréductible**, le vecteur propre *r* a des entrées strictement positives, est **unique** et **simple**, et la valeur propre associée λ est la plus grande en valeur absolue (λ est égale au **rayon spectral** de *A*).

```
w, v = linalg.eig(A)

idx_lambda = int(asarray((w==max(w)).nonzero()))
w[idx_lambda]
r = abs(v[:,idx_lambda]);
rrank = r.argsort();
rrank = rrank.reshape(-1,1)

concatenate((r1rank,r2rank,rrank),axis=1)
```

- Pour calculer le vecteur propre r non-négatif, on peut utiliser une méthode itérative, la méthode des puissances

$$\lim_{n \to \infty} \frac{A^n r_0}{|A^n r_0|} = r$$