TP 13 informatique

BCPST 1 2019-2020

Révisions

Exercice 1. Écrire une fonction `est_triangulaire_sup(T)` qui prend en argument une matrice carrée T et qui renvoie `True` si T est triangulaire supérieure et `False` sinon.

Exercice 2. Soit A une matrice de $M_{n,p}(\mathbb{R})$. On note $R(A)$ le maximum des minimums de chaque colonne. Par exemple, si $A = \begin{pmatrix} 3 & 1 & 2 \\ 2 & -1 & 4 \\ 5 & 2 & 0 \end{pmatrix}$ on a $R(A) = 2$.

Écrire une fonction `calcul(A)` qui prend en argument une matrice de réels A et qui renvoit la valeur $R(A)$.

Exercice 3. (Extrait Agro 2019 sujet de modélisation, à faire sur feuille puis vérifier les résultats sur PC) Dans la suite, les variables n, p, A, M, i, j et c vérifient les conditions suivantes qui ne seront pas rappelées à chaque question :

- n et p sont des entiers tels que $p \geq n \geq 2$,
- A est une matrice carrée à n lignes inversible,
- M est une matrice à n lignes et à p colonnes telle que la sous-matrice constituée des n premières colonnes est inversible,
- i et j sont des entiers tels que $0 \leq i \leq n-1$ et $0 \leq j \leq p-1$,
- c est un réel non nul.

On note $L_i \leftarrow L_i + cL_j$ l'opération qui ajoute à la ligne i d'une matrice la ligne j multipliée par c.

On donne quelques éléments de syntaxe en Python. On suppose que le module `numpy` a été importée à l'aide de la commande `import numpy as np`.

Python	**Interprétation**
M[i,j]	coefficient (i, j) de la matrice M
np.zeros((n,p))	matrice à n lignes et p colonnes remplie de zéros
T=np.shape(M)	taille de la matrice M
np.shape(M)[0]	nombre de lignes de M
np.shape(M)[1]	nombre de colonnes de M

1. Soit la fonction `initialisation` :

```
def initialisation(A) :
    n = np.shape(A)[0]
    mat = np.zeros((n,2*n))
    for i in range(0,n) :
        for j in range(0,n) :
            mat[i,j] = A[i,j]
    return (mat)
```

Pour chacune des affirmations suivantes, indiquer si elle est vraie ou fausse, en justifiant.

L'appel `initialisation(A)` renvoie :

(a) une matrice rectangulaire à n lignes et $2n$ colonnes remplie de zéros ;

(b) une matrice de même taille A ;

(c) une erreur au niveau d'un `range` ;

(d) une matrice rectangulaire telle que les n premières colonnes correspondent aux n colonnes de A, et les autres colonnes sont nulles.

2. Les fonctions `multip,ajout` et `permut` suivantes ne renvoient rien : elles `modifient` les matrices auxquelles elles s'appliquent.
   (a) Que réalise la fonction `multip` ?

```
def multip(M,i,c) :
    p = np.shape(M)[1]
    for k in range(0,p) :
        M[i,k] = c*M[i,k]
```

   (b) Compléter la fonction `ajout`, afin qu'elle effectue l'opération $L_i \leftarrow L_i + cL_j$.

```
def ajout(M,i,j,c) :
    p = np.shape(M)[1]
    for k in range(0,p) :
        ----ligne(s) à compléter----
```

   (c) Écrire une fonction `permut` prenant en argument M, i, j et qui modifie M en échangeant les valeurs des lignes i et j.

Exercice 4. On veut écrire une fonction `Est_Echelonnee(M)` qui prend en argument une matrice M et qui renvoie `True` si M est échelonnée en lignes et `False` sinon.

1. Écrire une fonction `premier_non_nul(M,i)` prenant en argument une matrice $M \in \mathcal{M}_{n,p}$ et un entier positif $0 \leq i \leq n-1$ et qui renvoit l'indice colonne du premier coefficient non nul de la ligne i. Dans le cas où la ligne est remplie de 0, la valeur de retour est alors égale à -1.
2. En déduire une fonction `liste_premiers_non_nuls(M)` prenant en argument une matrice M ayant n lignes et p colonnes et qui renvoit une liste L où pour tout $i \in \{0, 1, \ldots, n-1\}$ $L[i]$ contient la valeur `premier_non_nul(M,i)`.
3. En remarquant qu'une matrice M de taille $n \times p$ est échelonnée si et seulement si la liste L=`liste_premiers_non_nuls(M)` est de la forme $[a_0, a_1, \ldots, a_k, -1, \ldots, -1]$ avec $0 \leq a_0 < a_1 < a_2 < \ldots < a_k$, écrire une fonction `Est_Echelonnee(M)` qui prend en argument une matrice M et qui renvoie `True` si M est échelonnée et `False` sinon.

Exercice 5. ($\star$ à faire si vous avez fini le reste)

Étant donnés deux individus i et j quelconques, une question naturelle que l'on peut se poser est :
est-ce que i connaît quelqu'un qui connaît quelqu'un qui connaît ... qui connaît j.

On dit que i et j sont des connaissances lointaines si i connaît quelqu'un qui connaît quelqu'un ... qui connaît j. Pour modéliser ce problème, on se limite au cas où on a n individus fixés. On considérera que i connaît j si et seulement si j connaît i. On suppose qu'il nous est donné les relations entre les individus : étant donnés i et j on sait si i et j se connaissent.

Du point de vue informatique et mathématique, cela revient à connaître la matrice A carrée de taille n où $A[i,j] = 1$ si i, j se connaissent et $A[i,j] = 0$ sinon.

On se placera dans le cas où un individu se connaît lui-même.

L'objectif est donc d'utiliser cette matrice A qui encode les relations entre individus pour résoudre le problème posé.

1. On considère le cas particulier de cinq individus : $0, 1, 2, 3, 4$. On étudie plusieurs configurations.
   (a) Dans cette question, on suppose que 0 et 3 se connaissent, 3 et 1 se connaissent, 2 et 4 se connaissent.
      i. Déterminer la matrice A correspondante.
      ii. Calculer A^2. Soit $(i,j) \in \{0,1,2,3,4\}^2$. Montrer que i connaît quelqu'un (éventuellement lui-même) qui connaît j si et seulement si $(A^2)_{i,j} \neq 0$.
      iii. Calculer A^3, A^4. Montrer que $(A^4)_{i,j} \neq 0$ si et seulement si i et j sont des connaissances lointaines.
   (b) Dans cette question, on suppose que 0 et 1 se connaissent, 1 et 2 se connaissent, 1 et 4 se connaissent 3 et 4 se connaissent.
      Répondre aux questions i, ii, iii dans ce cas.
2. On numérote les n individus par $0, 1, \ldots, n-1$.Écrire une fonction `connaissance_lointaine(A)` qui renvoie une matrice M telle que pour tout $(i,j) \in \{0, 1, \ldots, n-1\}^2$, $M[i,j] = 1$ si i, j sont des connaissances lointaines et 0 sinon. On pourra généraliser les exemples précédents en faisant le lien entre les puissances de A et la matrice des connaissances lointaines M.