Chapitre 6 : Calcul matriciel.

I Définir une matrice

Nous avons déjà utilisé des matrices lignes lorsque nous avons vu comment représenter graphiquement une fonction : la commande *x=min : pas : max* , ou *x=linspace(min,max,nombre de points)* produit une matrice ligne (on dira aussi vecteur ligne) de réels régulièrement espacés entre une valeur minimale et une valeur maximale.
En petite dimension, on peut définir une matrice en écrivant tous ses coefficients entre crochets, avec une séparation par des virgules, et par des points-virgules pour les passages à la ligne.

Ex : *-->A=[1,2,0;5,5,-8]*
A =
1. 2. 0.
5. 5. - 8.

La commande *zeros(n,p)* définit la matrice nulle n lignes p colonnes.
La commande *ones(n,p)* définit la matrice n lignes p colonnes dont tous les coefficients valent 1.
La commande *eye(n,n)* définit la matrice identité n lignes n colonnes.
La commande *rand(n,p)* définit une matrice n lignes p colonnes dont les coefficients sont des tirages au hasard d'un nombre réel compris entre 0 et 1 (autrement dit les coefficients sont des réalisations de variables aléatoires indépendantes de loi uniforme sur [0,1]).

Ex : faire quelques essais sur la console : *zeros(3,5)*, *rand(2,2), eye(3,3)* …

Si U est une variable contenant une matrice ligne ou colonne, l'instruction *U(k)* extrait le k-ème coefficient de U. Si A est une variable contenant une matrice, l'instruction A*(i,j)* extrait le coefficient de A situé sur la i-ème ligne et sur la j-ème colonne.

Ex : *->U=2:3:19;U(3)*
ans =
8.

-->A=[1,2,0;5,5,-8];A(2,2)
ans =
5.

II Opérations matricielles

Le tableau suivant précise la syntaxe sur Scilab des opérations matricielles usuelles. Soit A et B deux matrices pour lesquelles les opérations ci-dessous ont un sens, k un entier naturel et α un réel.

opération	A+B	αA	AB	A^k	A^{-1}	${}^tA$
syntaxe	A+B	α*A	A*B	A^k	inv(A)	A'

Ex 1 : Taper successivement les commandes suivantes <u>en essayant de prévoir le résultat</u> :

A=ones(2,2)	A^3	B=[1,1;2,3]	B'	B-A	inv(B)	inv(A)

Attention, certaines opérations matricielles sont possibles en scilab sans l'être en mathématiques.
Ex 2 : <u>Prévoir</u> puis tester pour vérifier votre réponse. : quel est le contenu de la variable A après la commande *A=(1:4)' *(2:5)* ?

Ex 3 : <u>Prévoir</u> puis tester pour vérifier votre réponse : quel est le contenu de la variable A après la commande *A=[1,2;-1,3]^(-1)* ? que va répondre Scilab à la commande *ones(2,2)^(-1) ?*

III Extraction et modification

Nous avons déjà vu comment **extraire** un coefficient.

Pour extraire la i-ème ligne d'une matrice A, on utilise la commande *A(i , :)*
Pour extraire la j-ème colonne d'une matrice A, on utilise la commande *A(: , j)*

Ex : Faire des essais : *B=rand(3,3) ;B(2,:)* puis *C=eyes(3,3) ;C(:,1) etc*

On peut également **modifier** un coefficient, ou une ligne, ou une colonne :

Si U est une variable contenant une matrice ligne ou colonne et a un réel, l'instruction *U(k)=a* modifie le k-ème coefficient de U en le remplaçant par a.

Si A est une variable contenant une matrice de taille n lignes p colonnes :
l'instruction *A(i,j)=a* avec a réel modifie le coefficient en position (i,j) de A en le remplaçant par a,
l'instruction *A(i, :)=B* avec B vecteur ligne de taille p modifie la ligne n°i de A en la remplaçant par B.
l'instruction *A(:, j)=C* avec C vecteur colonne de taille n modifie la colonne n°j de A en la remplaçant par C.

Ex 1 : *-->A=rand(3,4);A(:,3)=zeros(3,1)*
A =
0.6653811 0.6857310 0. 0.1985144
0.6283918 0.8782165 0. 0.5442573
0.8497452 0.0683740 0. 0.2320748

Ex 2 *:* *-->B=ones(2,3);B(2,:)=7*B(2,:);B*
B =
1. 1. 1.
7. 7. 7.

IV Commandes FIND et SIZE

La commande *size(A)* renvoie la taille de la matrice A sous forme d'un vecteur ligne à deux éléments :

Ex : *-->B=[1;4;5];size(B)*
ans =
3. *1.*

La commande *find* permet de trouver les éléments d'une matrice vérifiant une propriété donnée.

Ex : *-->A=rand(2,4)*
A =
0.2312237 0.8833888 0.3076091 0.2146008
0.2164633 0.6525135 0.9329616 0.312642

-->find(A>0.5)
ans =
3. 4. 6.

-->size(find(A>0.5))
ans =
1. 3.

Attention ! Les positions dans la matrice A des coefficients qui rendent le test vrai sont notées comme si A était un vecteur colonne contenant les colonnes de A écrites l'une sous l'autre. Ici, 3 coefficients de A sont strictement supérieurs à 0.5, ce sont les coefficients de numéros 3,4 et 6 selon la logique décrite ci-dessus.

V Opérations élément par élément

Une fonction usuelle f (sin, cos, exp, abs, floor, log, sqrt) appliquée à une matrice A donne la matrice dont les coefficients sont $f(a_{i,j})$.

Ex : -->A=[1,2,-4;3,-3,0];abs(A)
ans =
1. 2. 4.
3. 3. 0.

A et B étant deux matrice de même taille, la commande ***A .* B*** donne la matrice dont les coefficients sont $a_{i,j}\, b_{i,j}$, , la commande ***A . / B*** donne la matrice dont les coefficients sont $a_{i,j} / b_{i,j}$, enfin la commande **A.^k** donne la matrice dont les coefficients sont $a_{i,j}^{\,k}$ (k étant un entier).

Ex 1 : *-->A=[1,2,3];B=[2,-1,6];A.*B*
ans =
2. - 2. 18.

-->A./B
ans =
0.5 - 2. 0.5

Ex 2 : *-->C=ones(2,2),D=C^2,E=C.^2*
C =
1. 1.
1. 1.
D =
2. 2.
2. 2.
E =
1. 1.
1. 1.

TP 6 :

Ex 1 : Taper sur la console la suite d'instructions suivante et comprendre toutes les réponses (on appuie sur « entrée » à la suite de chaque instruction) :

```
a=[0,1,5]
a(2)
a(4)=-2
b=[a,3]
sum(b)
prod(b)
c=[b;2*ones(1,5)]
d=[b,2*ones(1,5)]
size(c)
c( :,4)
```

Ex 2 : Idem

```
a=0.5*ones(2,4)
b=(1:4)'
a*b
a.*b
a^2
a.^2
b.^{-1}
sqrt(b)
find(b<>2)
```

Ex 3 : Déclarer les matrices suivantes sans les entrer coefficient par coefficient. (à faire sur la console)

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \end{pmatrix} \qquad \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 4 & 4 & 4 & 4 \end{pmatrix} \qquad \begin{pmatrix} -1 & -3 & -5 & -7 \\ 1 & 3 & 5 & 7 \\ 2 & 6 & 10 & 14 \end{pmatrix}$$

Ex 4 : Compléter la commande suivante pour qu'elle construise la matrice $A = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$

A=[ones(1,5); --- , --- ; ones(1,5)]

Attention à bien différencier virgule et point-virgule !!

Ex 5 :

1) Ecrire un script qui forme une matrice aléatoire 4 x 4 appelée A puis écrit la matrice obtenue en remplaçant la première ligne par la dernière.
2) Modifier ce script pour qu'on obtienne aussi la matrice réalisée en échangeant les lignes 1 et 4 à partir de A.

Ex 6 : En une seule commande et à l'aide de la matrice ligne (1 2 3 … 10), construire la matrice 10x10 donnant les tables de multiplication.

Ex 7 : Compléter ce script qui demande à l'utilisateur un entier n ainsi qu'une matrice ligne à n éléments, puis qui doit afficher la valeur minimale des n coefficients de cette matrice.

```
n=input('n= ?')
u=zeros(1,n)
for i=1:n, u(i)=input('entrer une valeur')
end
min=u(1)
for i=1:n,        if--- then ---
                  end
end
disp(min)
```

Ex 8 : Faire tourner le programme suivant pour n=5 (plusieurs fois), n=10 (plusieurs fois), n=100. Observer les résultats et expliquer le fonctionnement du programme. A quoi correspond l'affichage en sortie ?

```
n=input('n= ?')
u=rand(1,n)
a=find(u>0.4)
b=size(a)
disp(b(2)/n)
```

Ex 9 : Ecrire un programme qui crée la matrice suivante de taille (n+1)x(n+1) :

$$A = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & 0 & 0 & \dots & 0 \\ . & . & . & \dots & . \\ . & . & . & \dots & . \\ 1 & 0 & 0 & \dots & 0 \end{pmatrix}$$ (s'inspirer de l'exercice 4)

Ex 9 (suite) : puis qui la modifie de manière à obtenir le triangle de Pascal et qui l'affiche. Commencer par rappeler la relation de récurrence qui sert à faire ce travail (formule de Pascal).

Ex 10 : On considère les deux systèmes linéaires suivants :

$$\left\{\begin{array}{c} 2x+y+z=3 \\ x+2y+z=-1 \\ x+y+2z=6 \end{array}\right\} \quad \text{et} \quad \left\{\begin{array}{c} x+y+z=0 \\ x+2y-2z=-1 \\ 2x+3y-z=-1 \end{array}\right\}$$

1) Les résoudre « à la main ». Voir que le premier est de Cramer, mais pas le second.
2) Utiliser les opérations du calcul matriciel pour résoudre le premier système sur la console de Scilab.
3) Tester la même commande appliquée au second système.

Ex 11 :

1) Créer la matrice ligne de taille 10 qui comporte les 10 premiers entiers non nuls au carré.
2) En déduire à l'aide de la fonction *sum* la valeur de $1+2^2+...+10^2$.

Ex 12 : En s'inspirant de l'exercice précédent, écrire :

1) Une fonction Scilab qui prend en entrée un entier positif n et qui renvoie la valeur de n !.
2) Une fonction Scilab qui prend en entrée un entier positif n et qui renvoie la somme des n premiers impairs (par exemple, si on entre 4, la valeur de sortie doit être 1+3+5+7=16).

Ex 13 :Tracé de la fonction $f(x)=x^2+\frac{1}{x}+3$ sur [1,5]. Ecrire un script qui réalise les actions suivantes :

- créer la matrice ligne x des abscisses,
- créer la matrice ligne des ordonnées grace aux opérations pointées,
- dessiner la courbe

Comment faisait-on sans les opérations pointées ?

Ex 14 : Soit la suite définie par $u_0=2$ et $u_{n+1}=\sqrt{2+u_n}$.

1) Ecrire une fonction qui prend un entier positif n en entrée et qui renvoie u_n .
2) Modifier la fonction précédente afin qu'elle donne en sortie la matrice ligne $(u_0 u_1 ... u_n)$
3) Représenter graphiquement les 15 premiers termes de cette suite (utiliser *plot(u(n),'xr')* pour que les points de coordonnées (n, u_n) soient matérialisés par une croix rouge)
4) Que peut-on conjecturer si $u_0=1$? Et si $u_0=3$?

Ex 15 : Soit la suite définie par $u_1=1$ et $u_{n+1}=1-e^{-u_n}$. Compléter le programme suivant afin d'obtenir la représentation graphique des 100 premiers termes de cette suite.

```
u=zeros(1,100)
----
for ---
---
end
plot(u,'+')
```

Ex 16 : Compléter le programme suivant pour qu'il prenne en entrée une matrice A et donne en sortie la matrice transposée.

```
function B=transpo(A)
        B=[]
        [n,p] =.....
        for k=1 : …....
                for j=1 :.....
                        B(i,j)=.....
                end
        end
endfunction
```

Ex 17 : Écrire une fonction scilab qui prend en entrée une matrice 2×2 , et donne en sortie une phrase caractérisant l'inversibilité de la matrice et si possible, donnant l'inverse de celle ci ?

Ex 18 : Résoudre avec scilab les 2 systèmes suivants et constater :

$$\begin{cases} x+9y+10z=-50 \\ 9x+5y+z=180 \\ 5x+10y+9z=40 \end{cases} \quad \text{et} \quad \begin{cases} x+9y+10z=-50 \\ 9x+5y+z=180 \\ 5x+10y+9z=41 \end{cases}$$