Université René Descartes Paris 5
UFR de mathématiques et informatique

Licence de mathématiques
Licence MASS

Méthodes Numériques

Examen du 12 mai 2004

Durée 3h
Documents autorisés : documents distribués et notes personnelles de cours
Calculettes et téléphones portables interdits

I

SCILAB exécute les instructions suivantes. Donner l'affichage obtenu à l'écran (uniquement l'affichage obtenu à l'écran par SCILAB, tout affichage supplémentaire sera pénalisé).

```
u=1:5
A=u;
for i=1:4,A=[A;u];end,A
D=diag(u)
A=D*A
for i=1:5, A(i,:)=A(i,:)-A(2,:);end,A
for i=5:-1:1, A(i,:)=A(i,:)-A(4,:);end,A
A=A-ones(5,1)*A(3,:)

A=[3 6 5 3;2 1 2 1;8 1 2 8; 2 2 3 1]
u= A([2,4],2)'
v=[-u(2),u(1)]
A([2,4],:)=[u;v]*A([2,4],:)

deff('y=somme(t)','y=0,for i=1:size(t,''c''),y=y+t(i),end')
s=somme(u)

deff('t=div(t1,t2)','n=size(t1,''c''),for i=1:n,t(i)=t1(i)/t2(i),end')
uv=div(u,v)

deff('y=f(x)','y=x^2')
y=f(u)
```

II

Les 4 versions suivantes d'instructions SCILAB donnent dans `s` le même résultat.

```
(1) s=0,for i=1:n,u=1/i,s=s+u,end
```
```
(2) s=0;for i=1:n,u(i)=1/i;s=s+u(i);end
```
```
(3) for i=1:n,u(i)=1/i;end,s=sum(u)
```
```
(4) u=ones(1,n)./[1:n],s=sum(u)
```
```
(5) deff('y=somme(t)','y=0,for i=1:size(t,''c''),y=y+t(i),end')
          for i=1:n,u(i)=1/i;end,s=somme(u)
```
```
(6) deff('y=somme(t)','y=0,for i=1:size(t,''c''),y=y+t(i),end')
    deff('t=div(t1,t2)','n=size(t1,''c''),for i=1:n,t(i)=t1(i)/t2(i),end')
    u=div(ones(1,n),[1:n])',s=somme(u)
```

1. Quel est ce résultat ?
2. Certaines versions utilisent plus de mémoire que d'autres. Ecrire les numéros des versions ordonnées en commençant par la version qui nécessite le plus de mémoire, en terminant par celle qui en nécessite le moins et en indiquant les ex æquo, de la façon suivante, par exemple, si la version (1) nécessite le plus de mémoire, la version (6) le moins et les autres la même taille entre les deux, on écrira 1 > 2 = 3 = 4 = 5 > 6
3. Certaines versions sont plus rapides que d'autres. Ecrire les numéros des versions ordonnées en commençant par la version qui est la moins rapide, en terminant par celle qui est la plus rapide et en indiquant les ex æquo, de la façon suivante, par exemple, si la version (1) est la moins rapide, la version (6) la plus rapide et les autres prennent le même temps intermédiaire, on écrira 1 < 2 = 3 = 4 = 5 < 6

III

On considère l'ensemble de courbes paramétrées définies par $y=f_k(x)$ avec $f_k(x)=\sin(x+k)$.
1. Ecrire un programme SCILAB qui trace le graphe de la fonction $y=f_k(x)$, pour un k fixé, pour x compris entre x1 et x2.
2. Ecrire un programme SCILAB qui trace sur le même dessin les graphes de ces courbes pour toutes les valeurs de k comprises entre k1 et k2 par pas de p,

a) en utilisant une boucle `for` explicite et un appel à `plot2d` pour chaque courbe;
b) en utilisant un seul appel à `plot2d` dessinant l'ensemble de ces courbes.

IV

Soit $A = \begin{bmatrix} -1 & 2 & 1 \\ 2 & 4 & 3 \\ 1 & 3 & 2 \end{bmatrix}$

1. Calculer l'inverse de A par la méthode de Gauss-Jordan avec pivot maximal en appliquant l'algorithme vu en cours utilisant une seule matrice 3×3.
2. Calculer le conditionnement de A pour les normes 1, ∞ et 2, sachant que les valeurs propres de cette matrice sont -1.70, -0.087 et 6.79.
3. On résout un système linéaire de la forme A X = B où A est la matrice précédente. B est connu avec une précision relative de 1 ‰. A quelle précision est connue la solution X ?

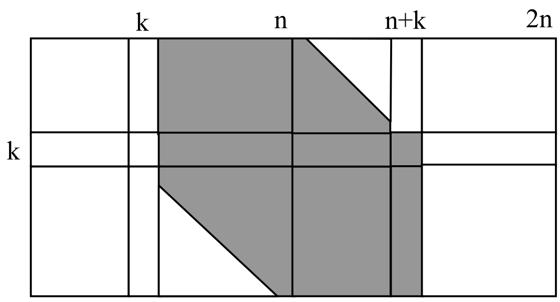
V

Matrices symétriques
1. Montrer que l'inverse d'une matrice symétrique est une matrice symétrique.

2. Montrer que dans l'algorithme de Gauss-Jordan, à l'étape k, la sous-matrice A(k+1:n, k+1:n) est symétrique.

3. On considère l'algorithme de calcul d'inverse du cours utilisant deux matrices n×n, c'est-à-dire une matrice n×2n

a) Montrer que, dans le cas des matrices symétriques, il suffit de calculer à l'étape k les parties des matrices grisées dans le dessin ci-dessous.



b) Modifier l'algorithme en conséquence. (Il pourra être nécessaire de sauvegarder des valeurs dans les parties non utilisées pour le calcul). On répondra à cette question en indiquant les modifications sur le programme de la feuille de réponse jointe.

c) Application au calcul de l'inverse de la matrice $\begin{bmatrix} -1 & 2 & 1 \\ 2 & 4 & 3 \\ 1 & 3 & 2 \end{bmatrix}$.

On donnera les valeurs de la matrice 3×6 uniquement à la fin de chaque étape (au sens du cours) en <u>encadrant</u> les valeurs calculées à cette étape et en <u>soulignant</u> les valeurs recopiées pour être sauvegardées à cette étape. (Les valeurs non modifiées à cette étape seront écrites simplement sur la copie).

d) Calculer la complexité de cet algorithme.

4. On considère maintenant l'algorithme du cours utilisant une seule matrice n×n.

a) Définir les parties de la matrice qu'il suffit de calculer. On pourra faire un dessin en s'inspirant de celui de la question 3 a).

b) et c) Questions identiques aux b) et c) de la question 3.

Feuille de réponse pour les questions V 3.b et V.4.b

question V.3.b

```
function gjinv = gaussjordaninv1(A)

n=size(A,'c') // calcul de la taille de A et vérif carrée
if ~ size(A,'r')==n  then printf('matrice non carree'), end
A(:,n+1:2*n)=diag(ones(1,n)),

for k = 1 : n ,   // etape k
   if A(k,k)==0 then ...

   end,

   p=A(k,k)  // pivot


   A(k,[k:n+k]) = A(k,[k:n+k])/p, // nouvelle ligne k

   for i = [1:k-1,k+1:n] ,   // nouvelles ligne i



       A(i,[k:n+k]) = A(i,[k:n+k]) - A(i,k)*A(k,[k:n+k]),

   end,
end,


gjinv=A(:,n+1:2*n), // c'est l'inverse
```

question V.4.b

```
function gjinv = gaussjordaninv2(A)

n=size(A,'c') // calcul de la taille de A et vérif carrée
if ~ size(A,'r')==n  then printf('matrice non carree'), end

for k = 1 : n ,   // etape k
   if A(k,k)==0 then ...

   end,

   p=A(k,k)  // pivot


   A(k,[1:k-1,k+1:n]) = A(k,[1:k-1,k+1:n])/p, // nouvelle ligne k

   for i = [1:k-1,k+1:n] ,   // nouvelles ligne i



            A(i,[1:k-1,k+1:n]) = A(i,[1:k-1,k+1:n]) - A(i,k)*A(k,[1:k-1,k+1:n]),



   end,

   A([1:k-1,k+1:n],k) = -A([1:k-1,k+1:n],k)/p,

   A(k,k) = 1/p,

end,



gjinv=A, // c'est l'inverse
```