

TP d'introduction à Scilab

Nicolas KIELBASIEWICZ

Exercice 1 : Premiers pas avec Scilab

1. Définir des vecteurs lignes :

Soit $\vec{u} = (1 \ 2 \ 3)$

- (a) Méthode directe : Taper les commandes suivantes :

```
u=[1 2 3]
u=[1,2,3]
u=[1 2 3];
```

Soulignez le rôle du point-virgule et donc l'importance de l'utiliser en fin de chaque ligne de commande. Pour afficher une variable, on utilisera alors la commande **disp**.

- (b) Fonctions plus avancées :

- i. Taper les commandes suivantes :

```
u=linspace(1,3,3)
u=1:1:3
```

- ii. Ouvrir, avec votre éditeur favori, un fichier `exo1.sce` et définir ce vecteur \vec{u} . Pour exécuter ce fichier dans la fenêtre scilab, taper **exec('exo1.sce')**; A partir de maintenant, on travaillera dans le fichier `exo1.sce`.

2. Définir des vecteurs colonnes

- (a) Taper la commande suivante :

```
v=[-5 ;2 ;1];
```

- (b) Définir de la même manière le vecteur $\vec{w} = \begin{pmatrix} -1 \\ -3 \\ 7 \end{pmatrix}$

3. Définir des matrices : vers la meilleure méthode

Soit la matrice $A = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix}$

- (a) Méthode directe :

Taper la commande suivante :

```
A=[2 -1 0 ; -1 2 -1 ; 0 -1 2];
```

Extraire le coefficient de la deuxième ligne et troisième colonne.

- (b) Utiliser des boucles :

Taper et compléter les instructions suivantes :

```

A=zeros ( 3 , 3 );
i =1;
while ( ... )
A(i , i)=2;
...
end
for i =1:2
A(i , i+1)=-1;
A(i+1 , i)=-1;
end
disp(A)

```

(c) Utiliser l'insertion/extraction :

Pour extraire une sous-matrice, on utilise l'opérateur **colon**, à savoir « : »

Extraire les éléments de A se trouvant entre les lignes 1 et 2 et les colonnes 2 et 3.

Taper et compléter la série d'instructions suivantes pour obtenir la matrice voulue à l'aide de l'opérateur **colon** :

```

A=2*eye ( 3 , 3 );
B=-1*eye ( 2 , 2 );
...
...

```

4. Opérations matricielles :

(a) Calculer $\vec{u}' + 3\vec{v} - \vec{w}/5$.

(b) Définir dans un fichier lib.sci la fonction produit scalaire de deux vecteurs colonnes. Pour cela, taper et compléter les lignes suivantes :

```

function y=sp ( u , v )
...
endfunction

```

On ajoutera en début de fichier exo1.sce la commande **getf('lib.sci')** ;, pour charger la librairie. Tester cette fonction en calculant le produit scalaire entre \vec{v} et \vec{w} .

(c) Dans le fichier exo1.sce, à l'aide de la commande **deff**, définir la fonction *sq* qui calcule le carré de toutes les coordonnées d'un vecteur.

(d) Calculer $A\vec{v}$.

5. Utilisation de l'aide

A l'aide de la commande **help**, trouver les commandes Scilab permettant de :

(a) calculer $\|u\|_1$, $\|v\|_2$ et $\|w\|_\infty$;

(b) déterminer les dimensions de la matrice A, en extraire le nombre de colonnes ;

(c) calculer le déterminant et l'inverse de A ;

(d) résoudre le problème $A\vec{x} = \vec{v}$ (il y en a deux). On écrira pour cela deux fonctions séparées dans un fichier mylinsys.sci .

6. Entrée/sortie et instruction de contrôle

L'objectif est d'écrire un script qui permet de choisir laquelle des deux fonctions précédentes exécuter. Ecrire dans un fichier main.sce les instructions suivantes en remplaçant les pointillés par l'instruction adéquate :

```

getf( 'mylinsys.sci' );
printf( "Choix_de_la_fonction_\n" );
printf( "1->_nom_fct1_\n" );
printf( "2->_nom_fct2_\n" );
x=input( "Votre_choix:" );
if (x==1) then
...
else
...
end

```

Exercice 2 : Autour du laplacien

On va éditer un fichier `laplacien1D.sce`

1. Assemblage de matrices

On considère le problème suivant :

$$-\frac{d^2u}{dx^2} = 4\pi^2 \sin 2x \text{ pour tout } x \in]0, 1[$$

On utilisera pour cela le schéma centré d'ordre 2.

Assembler la matrice du laplacien. On notera que sa structure étant tridiagonale, on aura tout intérêt à utiliser judicieusement les commandes **spzeros**, **speye** et **colon** pour éviter d'écrire la moindre boucle.

2. Résolution d'un système linéaire - Affichage 2D

- Résoudre le problème à l'aide de la factorisation de Cholesky. On utilisera les commandes **chfact** et **chsolve**.
- Afficher la solution à l'aide de la commande **plot2d**.
- Mêmes questions avec le problème suivant (on éditera le fichier `laplacien2D.sce`) :

$$-\frac{d^2u}{dx^2} - \frac{d^2u}{dy^2} = 13\pi^2 \sin 2x \cos 3y \text{ pour tout } (x, y) \in]0, 1[\times]0, 1[$$

On utilisera également le schéma centré d'ordre 2. Même remarque concernant l'assemblage de la matrice qui cette fois-ci sera pentadiagonale. On utilisera cette fois-ci les commandes **matrix** et **plot3d** pour l'affichage.

Problème : Autour des transformations géométriques dans l'espace

1. Les entrées/sorties : génération de l'objet source

L'objet de cette partie du programme est de générer un vase 3D à symétrie cylindrique à partir des données (r_i, z_i) issues d'un fichier.

- Récupérer dans le fichier `generatrices.txt` les données dans une matrice à deux colonnes a . On utilisera à cet effet les commandes **mopen**, **mfscanf** et **mclose**. On définira la matrice gen comme étant la transposée de a .
- Ecrire une fonction `vase` qui génère les positions cartésiennes des points de l'objet à partir de ses positions cylindriques, c'est à dire qu'elle prendra comme argument un vecteur d'angles $angle$, et la matrice gen .

2. Structure de programme : les trois transformations élémentaires

On considère un objet (en l'occurrence notre vase), représenté par une matrice xyz $3 \times n$, la i -ème colonne de cette matrice représentant les coordonnées du i -ème point de l'objet. On va éditer un fichier `transformations.sci` qui contiendra plusieurs fonctions.

- (a) Ecrire une fonction translation. Elle prendra comme arguments $vect$, le vecteur directeur ligne de la translation, et la matrice xyz des points de l'objet. On pourra utiliser judicieusement la commande **ones**.
- (b) Ecrire une fonction rotation calculant l'image d'un objet xyz par rotation d'angle le vecteur ligne $angle$. On notera que toute rotation de l'espace peut être déterminée par la composée des rotations autour de chacun des axes.

Rappel : La matrice de rotation d'angle θ autour de l'axe des abscisses s'écrit :

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & \sin \theta \\ 0 & -\sin \theta & \cos \theta \end{bmatrix}$$

- (c) Ecrire une fonction calculant l'image d'un objet xyz par homothétie de centre le vecteur ligne $centre$ et de rapport le vecteur ligne f .

Rappel et astuce :

M' est l'image de M par l'homothétie de centre Ω et de rapport k ssi $\overrightarrow{\Omega M''} = k \overrightarrow{\Omega M}$ On peut réécrire cette relation en utilisant la relation de Chasles avec l'origine O de notre repère spatial.

On obtient donc $\overrightarrow{OM'} = \overrightarrow{O\Omega} + k(\overrightarrow{OM} - \overrightarrow{O\Omega})$

On peut donc décomposer une homothétie de la façon suivante :

- une translation de vecteur $-\overrightarrow{O\Omega}$
- une mise à l'échelle des coordonnées de facteur k
- une translation de vecteur $\overrightarrow{O\Omega}$. On peut donc utiliser judicieusement la fonction translation définie précédemment, ainsi que la commande Scilab **diag**.

3. Affichage 3D

- (a) Utiliser la commande **eval3dp** pour générer les facettes quadrangulaires de notre objet.
- (b) Utiliser la commande **plot3d** pour afficher notre vase.

4. Utilisation des transformations

- (a) Calculer l'image de notre vase par la composée de la translation de vecteur $[1 \ 3 \ -3]$ et de l'homothétie de centre $[1 \ 3 \ -3]$ et de rapport 1.5 dans les trois directions d'espace.
- (b) Reprendre la question précédente en remplaçant l'homothétie par la rotation d'angle $[40 \ 0 \ 90]$. Pour l'affichage, on utilisera les commandes **matrix** et **plot3d**.