INITIATION A SCILAB

Scilab est un logiciel de calcul de numérique conçu par l'INRIA (France). Il est doté d'un environnement et d'un langage de programmation propres. Scilab permet en particulier d'effectuer un grand nombre de calculs sur les matrices et sur les fonctions, de résoudre des équations ou des systèmes d'équations, de résoudre des équations différentielles et de visualiser graphiquement plusieurs types de données. C'est un logiciel libre qui peut être téléchargé sur le site http://www.scilab.org.

Scilab comporte un grand nombre de fonctions prédéfinies. Il est en conséquence difficile de le décrire en entier. L'étudiant est invité à travers cette séance de travaux pratiques à s'habituer progressivement à son utilisation, en maîtrisant les commandes courantes. Il peut aussi s'appuyer sur l'aide en ligne (voir l'icône correspondante) ou utiliser la commande `help`, ou se procurer des rétérences à ce sujet comme celles indiquées ci-dessous. Il est à noter qu'il y a deux manières d'exécuter des instructions sous Scilab : soit en ligne de commande directement, soit en les regroupant dans des fichiers de commande (appelés scripts). Les scripts ont traditionnellement l'extension `.sce` quand ils contiennent des instructions, ou l'extension `.sci` quand ils contiennent une ou plusieurs fonctions. La commande `exec nomScript` sert à exécuter un script (on peut aussi accéder à cette commande dans le menu de l'éditeur). Il est conseillé d'utiliser l'éditeur de texte propre à Scilab.

Exercice 1 - Taper les commandes suivantes dans l'ordre indiqué et observer puis commenter l'effet de chacune

```
X = [4, 5, 7]
X = [4 5 7]
X X = [4;5;7]
X = [4 5 7]'
X = [4 5 7] ;
X = [4:7]
Y = [sqrt(3) 12+2*%i %pi -1]
X*Y
X'*Y
X.*Y
C = [5 : 7; 1, 2, 3; 7 : 9]
A = [4 : 7; 1, 2, 3; 7 : 9]
B(3, 2) = 13
size(B)
C = zeros(3, 2)
X = [4:0.5:7]
Z = X.*X-10*X; plot(X, Z, 'o');
A = [1 2 3; 2 4 3; 1 1 -1]
A'
det(A)
inv(A)
A*A*A
A^3
b = [-1 ; -4; 17]
inv(A)*b
A\b
C = [-1:1;0:2;-1:1]
A*C
```

```
A.*C
tic()
toc()
disp('J''aime l''UVSQ');
deff('y = uvsq(x)', 'y = x*cos(x*%pi) + log(x)');
uvsq(1), uvsq(2), disp('genial')
x=[1:0.1:5]; fplot2d(x, uvsq)
clf() ; pwd
ls
mkdir RepEssaiTp1
ls
chdir RepEssaiTp1/
pwd
chdir ../
pwd
help plot
```

Annexe.

Voici la synthaxe générale de quelques instructions :

a) Instruction de contrôle `if` :

`if` condition `then`
  instructions
`else` (éventuellement)
  instructions
`end`

b) Instruction `while` :

`while` condition
  instructions
`end`

c) Boucle `for` :

`for p = d:p:n`
  instructions
`end`
où `p` est l'indice des itérations, `d` l'indice de départ, `n` l'indice final et `p` le pas (optionnel).

d) Fonctions :

`function [ps1, ps2, ···] = nom_fonction(pe1,pe2,···)`
  ⋮
`endfunction`

où `ps1, ps2, ...` sont les paramètres de sortie tandis que `pe1,pe2,···` sont les paramètres d'entrée.

Voici quelques exemples

— Calcul des 100 premiers termes de la suite définie par :
$u_1 = 1; u_n = \sin(u_{n-1})$ pour $n \geq 2$.

```
u(1) = 1;
for n = 2:100
   u(n) = sin(u(n-1));
end
```

— Calcul de la somme des entiers $n \geq 1$ vérifiant $n + \ln(n^3 + 1) \leq 400$:

```
som = 0;
n = 1;
while (n + log (n^3+1) <= 400)
   som = som + n;
   n = n + 1;
end
```

— Fonction f définie sur $\mathbb{R}$ par

$$f(x) = \begin{cases} 1 + \ln(x+1) \text{ si } x > 0, \\ e^x \text{ sinon.} \end{cases}$$

```
function [y] = mafonction(x)
  if (x > 0) then
    y = 1 + log(x+1);
  else
    y = exp(x);
  end;
endfunction
```