

Formulaire 1 : Introduction au langage Scilab*version du 25 février 2016*

- Le langage informatique **Scilab** est un langage interprété de calcul scientifique en *open source* (c'est à dire gratuit). L'environnement **Scilab** est très comparable à l'environnement **Matlab** qui permet aussi une utilisation souple et interactive. On introduit dans la suite quelques instructions permettant un traitement graphique de base ou le calcul sur des tableaux à valeurs numériques (matrices, vecteurs ...). De nombreuses fonctions mathématiques existent dans le langage comme les fonctions sinus, cosinus, exponentielle
- D'autres fonctions plus spécifiques peuvent être programmées par l'utilisateur : c'est la notion de **function** qui sera étudiée dans les chapitres suivants.
- On trouvera dans ce cours une introduction élémentaire au logiciel **Scilab** (interface graphique et langage de programmation) par la pratique et l'exemple. De nombreux manuels **Scilab** sont disponibles. Les notions de base sont les mêmes quelque soit le langage.

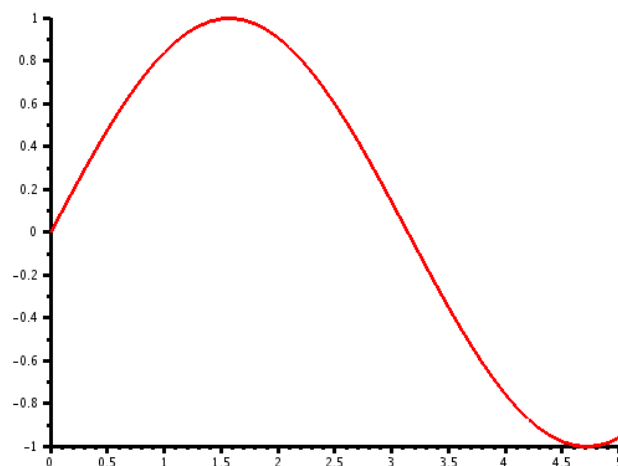
1) L'environnement graphique : la commande plot2d

L'instruction **plot2d** permet le tracé de graphe (dans le plan). Dans la commande **plot2d(x,y,icolor)**, **x** et **y** sont des tableaux et **icolor** est un indice de couleur (**icolor=1** correspond au noir, **icolor=2** au bleu etc.).

1.1) Exemple 1 : le tracé de graphe. Les instructions suivantes :

```
clear
x=0:0.1:5;y=sin(x);
plot2d(x,y,5);
```

conduisent au résultat suivant :



La commande **x=0:0.1:5;** génère un tableau ligne **x** de nombre de 0 à 5 par pas de 0.1 :

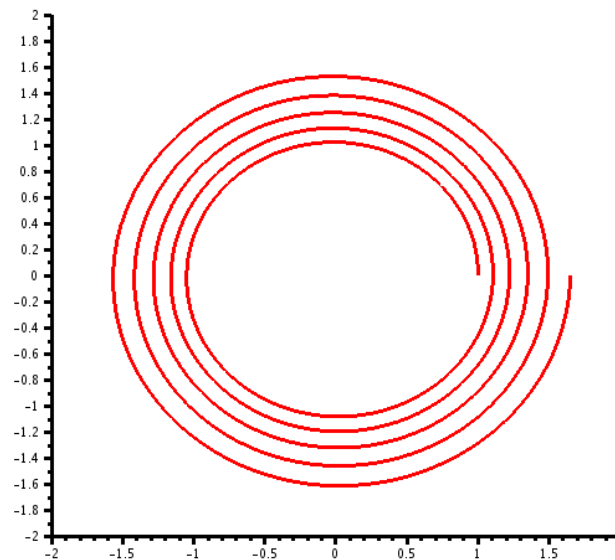
x = [0 , 0.1 , 0.2 , 0.3 , . . . , 4.9 , 5]

et la commande `y=sin(x)`; génère un tableau ligne `y` tel que `y(i)=sin(x(i))`.

1.2) Exemple 2 : le tracé de courbe paramétrée. Les instructions suivantes :

```
clear
t=0:0.01:5;N=length(t);a=0.1;b=2*%pi;
for i=1:N
x(i)=exp(a*t(i))*cos(b*t(i));y(i)=exp(a*t(i))*sin(b*t(i));
end
plot2d(x,y,5);
```

conduisent au résultat suivant :



La fonction `length(t)` donne la dimension du tableau `t`.

2) La notion de tableau

On a déjà rencontré la notion de tableau ligne. La matrice A (3×3) suivante :

$$A = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

est générée par l'instruction `A=[2,1,0;1,2,1;0,1,2]`; et le résultat dans la fenêtre de commande (Scinotes) est le suivant :

```
-->A
A  =
    2.    1.    0.
    1.    2.    1.
    0.    1.    2.
-->
```

2.1) Le produit de matrice. Si A et B sont des tableaux dont les dimensions sont compatibles le produit est effectué par la commande $C=A*B$. Les instructions suivantes :

$A=[2,1,0;1,2,1;0,1,2]; B=[1,1;1,1;1,1]; C=A*B;$

gènèrent le résultat suivant :

```
-->C
C  =
    3.    3.
    4.    4.
    3.    3.
-->
```

Le produit d'une matrice par un vecteur n'est qu'un cas particulier puisque les instructions

$A=[2,1,0;1,2,1;0,1,2]; X=[1;1;1]; Y=A*X;$

conduisent à :

```
-->Y
Y  =
    3.
    4.
    3.
-->
```

2.2) La transposition. Si A est un tableau $B=A'$ est le tableau transposé. Les instructions suivantes :

$A=[3,1,0,0;1,2,1,5;0,1,2,3]; B=A';$

gènèrent le résultat suivant :

```
-->B
B  =
    3.    1.    0.
    1.    2.    1.
    0.    1.    2.
    0.    5.    3.
-->
```

On calcule donc facilement la norme euclidienne d'un vecteur X par les instructions :

$X=[3;2;5]; norme=sqrt(X'*X);$

avec le résultat suivant :

```
-->norme
norme  =
    6.164414
-->
```

La commande `norm(X)` permet de calculer directement cette norme.

3) Les formats d'impression

On utilise la commande `printf` pour les sorties numériques. Le programme suivant :

```
a=exp(1);N=10;eps=1e-6;
printf(" a= %f ; N=%d ; eps=%e \n",a,N,eps);
```

donne le résultat suivant :

```
a= 2.718282 ; N=10 ; eps=1.000000e-06
```

4) Les commandes `while` et `if`

4.1) La commande `if`. L'exécution conditionnelle se fait de manière classique avec la commande `if condition ..... end`. Le programme suivant permet de générer la matrice A de l'exercice 4 :

```
clear
N=8;A=2*eye(N,N);
for i=1:N for j=1:N
if(abs(i-j)==1) A(i,j)=-1; end
end end
valpr=spec(A);X=rand(N,1,"uniform");
// Tracé du graphe du polynôme caractéristique entre t=0 et t=4
t=0:0.01:4;M=length(t);
for k=1:M
y(k)=det(t(k)*eye(N,N)-A);
end
plot2d(t,zeros(1,M),1);
xset("thickness",2);plot2d(t,y,5);
for l=1:N
plot2d(valpr(l),0,-3);
end
// Calcul de la plus grande valeur propre par la méthode de la puissance itérée
test=1;eps=1e-8;i=0;
while test>eps
Y=A*X;lambda=Y'*X;Y=Y/norm(Y);test=norm(Y-X);
X=Y;i=i+1;
end
printf("nombre d itération %d, lambda=%f, test=%e \n",i,lambda,test);
```

Il donne le résultat suivant :

```
-->A
A =
    2.   - 1.    0.    0.    0.    0.    0.    0.
   - 1.    2.   - 1.    0.    0.    0.    0.    0.
    0.   - 1.    2.   - 1.    0.    0.    0.    0.
    0.    0.   - 1.    2.   - 1.    0.    0.    0.
```

```

0.    0.    0.  - 1.    2.  - 1.    0.    0.
0.    0.    0.    0.  - 1.    2.  - 1.    0.
0.    0.    0.    0.    0.  - 1.    2.  - 1.
0.    0.    0.    0.    0.    0.  - 1.    2.

```

-->

4.2) La commande while. Elle permet d'exécuter une série d'instructions tant qu'un test logique est satisfait. Le programme suivant permet de calculer les suites X_n et λ_n de l'exercice 2 jusqu'à la convergence (à un ε près) :

```

clear
A=[2,-1,0;-1,2,-1;0,-1,2];X=[1;1;1];
test=1;eps=1e-5;i=0;
while test>eps
Y=A*X;lambda=Y'*X;Y=Y/norm(Y);test=norm(Y-X);
X=Y;i=i+1;
printf("Itération %d X=[%f,%f,%f] lambda=%f \n",i,X(1),X(2),X(3),lambda);
end

```

Il donne le résultat suivant :

```

-->exec('/home/demay/Ens:15-16/L2/A-Introduction/Exo_2.sci', -1)
Itération 1 X=[0.707107,0.000000,0.707107] lambda=2.000000
Itération 2 X=[0.577350,-0.577350,0.577350] lambda=2.000000
Itération 3 X=[0.514496,-0.685994,0.514496] lambda=3.333333
Itération 4 X=[0.502519,-0.703526,0.502519] lambda=3.411765
Itération 5 X=[0.500433,-0.706494,0.500433] lambda=3.414141
Itération 6 X=[0.500074,-0.707002,0.500074] lambda=3.414211
Itération 7 X=[0.500013,-0.707089,0.500013] lambda=3.414213
Itération 8 X=[0.500002,-0.707104,0.500002] lambda=3.414214
Itération 9 X=[0.500000,-0.707106,0.500000] lambda=3.414214
-->

```

Exercices

- Exercice 1 : Tracer le graphe entre $x = 0$ et $x = 2$ de la fonction :

$$y(x) = \max(\sin(\pi x), x(1-x))$$

- Exercice 2 : On considère la matrice A et le vecteur X_1 suivants :

$$A = \begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} \quad ; \quad X_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

- 2.1) Ecrire le programme calculant la puissance 5 de la matrice A .
- 2.2) Calculer les valeurs propres et les vecteurs propres de A . On vérifiera ce résultat en utilisant les commandes `spec(A)` et `[R,diagevals]=spec(A)`.
- 2.3) On construit les suites récurrentes suivantes ($\|Y\|$ désigne la norme euclidienne) :

$$X_{n+1} = \frac{AX_n}{\|AX_n\|} \quad ; \quad \lambda_n = \langle AX_n, X_n \rangle \quad (1)$$

Ecrire le programme calculant les N premiers termes de ces suites ($N = 10$ par exemple). On utilisera une boucle itérative `for`. Que constatez vous ?

- 2.4) Utiliser ensuite une boucle itérative conditionnée `while`. Vérifier que λ_n converge vers la plus grande valeur propre de A et que X_n converge vers le vecteur propre associé à cette plus grande valeur propre.
- 2.5) Est-ce que cela reste vrai pour d'autres valeurs de X_1 ?

- Exercice 3 : On considère maintenant la matrice A et le vecteur X_1 suivants :

$$A = \begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix} \quad ; \quad X_1 = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

- 3.1) Calculer les valeurs propres et les vecteurs propres de A en utilisant les commandes `spec(A)` et `[R,diagevals]=spec(A)`.
- 3.2) Tracer le graphe du polynôme caractéristique de A défini par $P(t) = \det(t \text{Id} - A)$ pour $0 \leq t \leq 4$. On utilisera les commandes `spec(eye(4,4))` et `spec(det)`.
- 3.3) Procéder comme en 2.4) et vérifier que λ_n converge vers la plus grande valeur propre de A et que X_n converge vers le vecteur propre associé à cette plus grande valeur propre. Est-ce que cela reste vrai pour d'autres valeurs de X_1 ?
- 3.4) Utiliser une valeur initiale de X_1 aléatoire en utilisant la commande `X1=rand(4,1,"uniform")`.

- Exercice 4 : Ecrire le programme `Scilab` générant la matrice A définie de dimension $N \times N$ par :

$$\begin{aligned} A(i,j) &= 2 && \text{pour } i = j \\ A(i,j) &= -1 && \text{pour } |i - j| = 1 \\ A(i,j) &= 0 && \text{pour } |i - j| > 1 \end{aligned}$$

On testera d'abord la solutions proposée en 4.1) puis on cherchera une solution utilisant les instructions `ones(N,N)` et `ones(N-1,N-1)`. Procéder ensuite comme dans l'exercice 3.

Solutions

• Exercice 1 :

```
x=0:0.01:2;N=length(x);
for i=1:N
y1(i)=sin(%pi*x(i));
y2(i)=x(i)*(1-x(i));
end
y=max(y1,y2)
plot2d(x,y1,5);plot2d(x,y2,2);
xset("thickness",3);plot2d(x,y,1);
```

• Exercice 2 :

```
clear
A=[2,-1,0;-1,2,-1;0,-1,2];X=[1;1;1];
test=1;eps=1e-5;i=0;
while test>eps
Y=A*X;lambda=Y'*X;Y=Y/norm(Y);test=norm(Y-X);
X=Y;i=i+1;
printf("Itération %d X=[%f,%f,%f] lambda=%f \n",i,X(1),X(2),X(3),lambda);
end
```

• Exercice 3 :

```
clear
A=[2,-1,0,0;-1,2,-1,0;0,-1,2,-1;0,0,-1,2];valpr=spec(A);X=rand(4,1,"uniform");
// Tracé du graphe du polynôme caractéristique entre t=0 et t=4
t=0:0.1:4;
for k=1:length(t)
y(k)=det(t(k)*eye(4,4)-A);
end
plot2d(t,zeros(1,length(t)),1);
xset("thickness",3);plot2d(t,y,5);
for l=1:4
plot2d(valpr(l),0,-3);
end
// Calcul de la plus grande valeur propre par la méthode de la puissance itérée
test=1;eps=1e-12;i=0;
while test>eps
Y=A*X;lambda=Y'*X;Y=Y/norm(Y);test=norm(Y-X);
X=Y;i=i+1;
printf("Itération %d X=[%f,%f,%f,%f] lambda=%f \n",i,X(1),X(2),X(3),X(4),lambda);
end
```

• Exercice 4 :

```
clear
N=8;A=2*eye(N,N);
for i=1:N for j=1:N
if(abs(i-j)==1) A(i,j)=-1; end
end end
valpr=spec(A);X=rand(N,1,"uniform");
// Tracé du graphe du polynôme caractéristique entre t=0 et t=4
t=0:0.01:4;M=length(t);
for k=1:M
y(k)=det(t(k)*eye(N,N)-A);
end
plot2d(t,zeros(1,M),1);
xset("thickness",2);plot2d(t,y,5);
for l=1:N
plot2d(valpr(l),0,-3);
end
// Calcul de la plus grande valeur propre par la méthode de la puissance itérée
test=1;eps=1e-8;i=0;
while test>eps
Y=A*X;lambda=Y'*X;Y=Y/norm(Y);test=norm(Y-X);
X=Y;i=i+1;
end
printf("nombre d itération %d, lambda=%f, test=%e \n",i,lambda,test);
```

