

TP 1

INFORMATIQUE 3

1- Objectif : Ce TP a comme but de vous familiariser avec l'usage de logiciel MATLAB de la compagnie Mathworks et à la programmation dans cet environnement. L'idée est de vous exposer les bases de cet outil de travail. On note bien que le nom « MATLAB » vient de la contraction MATrix LABoratory, ce qui signifie que toutes les variables sont considérées comme des matrices.

2- Présentation général

2.1. Vérifier qu'une variable scalaire est vue par MATLAB comme une matrice 1x1 (une ligne, une colonne). Comme le montre l'exemple suivant dans lequel on affecte à la variable x la valeur 3 et on demande ensuite ses dimensions par la commande size :



```
Command Window
>> x=3

x =

     3

>> size(x)

ans =

     1     1

fx >>
```

2.2. Réaliser l'exemple suivant permettant d'éviter l'affichage d'une variable scalaire, il suffit de suivre la commande par un point-virgule, ce qui permettra éventuellement des programmes plus rapides (pas de perte de temps due à l'affichage).



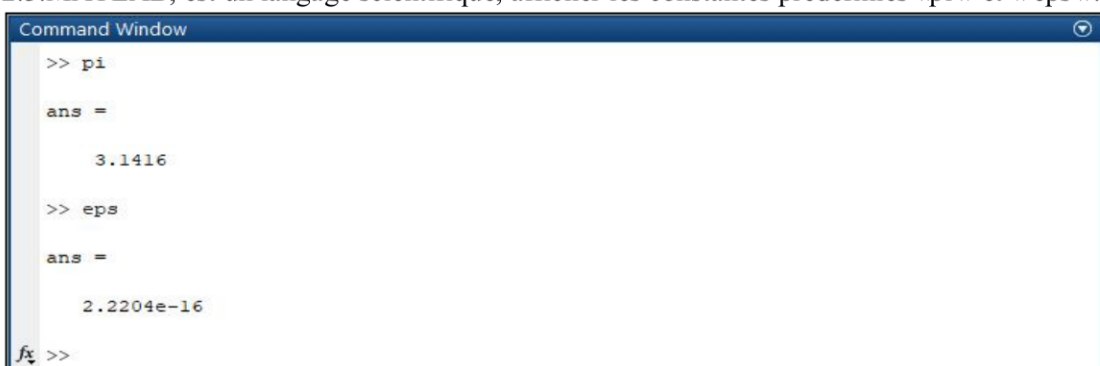
```
Command Window
>> x=5;
>> size(x)

ans =

     1     1

fx >> |
```

2.3. MATLAB, est un langage scientifique, afficher les constantes prédéfinies «pi » et « eps ».



```
Command Window
>> pi

ans =

     3.1416

>> eps

ans =

    2.2204e-16

fx >>
```

2.4. Afficher les variables i, j représentant le nombre imaginaire, comme :

```
Command Window

>> i

ans =

    0.0000 + 1.0000i

>> j

ans =

    0.0000 + 1.0000i

fx >>
```

2.5. Réaliser l'opération suivante (+ - / *..) :

```
>> x=3;
>> y=2;
>> x+y
ans =
     6
>> ans + 5
ans =
    11
```

Lorsque l'utilisateur ne fixe pas de variable de sortie, MATLAB place le résultat d'une opération dans « ans ». Cette variable temporaire peut bien sûr être utilisée pour un calcul suivant comme le montre l'exemple précédent. Il est toujours possible de connaître les variables utilisées et leur type à l'aide de la fonction whos.

La solution de x+y a donc été perdue. Il est donc préférable de toujours donner des noms aux variables de sortie :

```
>> a= x + y
a =
     6
```

2.6. La fonction clear permet d'effacer des variables. Réaliser l'exemple suivant :

```
>> clc      % On efface le contenu de la fenêtre de commande
>> clear x  % On efface x de la mémoire
>> x
Undefined function or variable 'x'.
```

Note. Le signe de pourcentage (%) permet de mettre ce qui suit sur une ligne en **commentaire** (MATLAB n'en tiendras pas compte à l'exécution).

3- Opérations mathématiques avec MATLAB : Scalaires, vecteurs, matrices

L'élément de base de MATLAB est la **matrice**. C'est-à-dire qu'un scalaire est une matrice de dimension 1x1. Un vecteur colonne de dimension n est une matrice nx1. Une vectrice ligne de dimension n est une matrice 1 x n.

3.1.Les scalaires se déclarent directement. Réaliser l'exemple suivant :

```
>> a = 3 ;  
>> a  
a =  
    3
```

3.2.Les vecteurs lignes se déclarent de la manière suivante. Réaliser l'exemple suivant :

```
>> VL = [ 2 1 5 ]  
VL =  
     2     1     5  
>> VL = [ 2, 1, 5]  
VL =  
     2     1     5  
>> size ( VL )  
ans =  
     1     3
```

Note. Dans ce cas on sépare les éléments par des espaces.

Comme chaque élément de MATLAB, le vecteur est une matrice. Ici c'est une matrice 1x3.

1 : Nombre de ligne.

3 : Nombre de colonne.

3.3.Les vecteurs colonnes se déclarent de la manière ci-dessous. Réaliser l'exemple suivant :

```
>> VC = [ 2 ; 0 ; 3 ]  
VC =  
  
     2  
     0  
     3  
>> size ( VC )  
ans =  
     3     1
```

Note. Dans ce cas on sépare les éléments par des points-virgules ou on utilise le retour chariot. Ici le vecteur est une matrice 3x1

3.4.La plus grande dimension d'un vecteur constitue sa longueur (length).

```
>> length ( VC )  
ans =  
     3
```

Mesurer la longueur du vecteur VC comme présent l'exemple ci-dessus.

3.5.Déterminer le transposer du vecteur VC à l'aide de la fonction « transpose » ou avec l'apostrophe (').

```
>> V = transpose (VC)  
V =  
     2     0     3
```

Ou,

```
>> V = VC '  
V =  
    2    0    3
```

3.6. Le double point (:) est l'opérateur d'incrémentation dans MATLAB. Ainsi, pour créer un vecteur ligne de **valeurs de 0 à 10 par incrément de 2**, réaliser l'exemple suivant :

```
>> x = [ 0: 2: 10]  
x =  
    0    2    4    6    8   10
```

3.7. Réaliser l'exemple ci-dessous permettant d'éviter de mettre les crochets si les composants d'un vecteur varient d'un pas constant :

```
>> x = 0: 2: 10  
x =  
    0    2    4    6    8   10
```

3.8. Réaliser l'exemple ci-dessous. Si l'incrément est de 1, le pas n'est pas noté. On met le (:) uniquement entre le premier et le dernier élément

```
>> y = 0: 10  
y =  
    0    1    2    3    4    5    6    7    8    9   10
```

3.9. Réaliser l'exemple ci-dessous permettant accéder à un élément d'un vecteur et même modifier celui-ci directement (ex : le troisième élément du vecteur ligne y et le remplacer par l'élément 7) :

```
>> y (3)  
ans =  
     2  
>> y (3) = 7  
y =  
    1    2    7    4    5    6    7    8    9   10
```

3.10. Réaliser l'exemple ci-dessous permettant la suppression d'un élément d'un vecteur, par exemple :

```
>> x = [ 1: 2: 10 ]  
x =  
    1    3    5    7    9  
>> x ( 4 ) = [ ]           % suppression de l'élément d'indice 4  
x =  
    1    3    5    9
```

3.11. Réaliser l'exemple ci-dessous permettant d'ajouter une valeur à l'élément d'indice i du vecteur, par exemple :

```
>> x = [ 1 3 5 9 ]
>> x(2)=x(2)+10      % on ajoute la valeur 10 à l'élément d'indice 2 du vecteur
x =
    1    13     5     9
```

3.12. Calculer la valeur moyenne d'un vecteur par la fonction « mean »:

```
>> x = [ 1 3 5 9 ]
x =
    1     3     5     9
>> m = mean(x)        % la valeur moyenne du vecteur x
m =
    4.5000
```

Les opérations usuelles d'addition, de soustraction et de multiplication par scalaire sur les vecteurs sont définies dans MATLAB :

```
>> x1 = [2 3];
>> x2 = [1 4];
>> x1+x2      % addition de vecteurs
ans =
     3     7
>> x2 - x1     % soustraction de vecteurs
ans =
    -1     1
>> x3 = 3 * x1 % multiplication par un scalaire
x3 =
     6     9
```

Dans le cas de la multiplication et de la division, il faut faire attention aux dimensions des vecteurs en cause. Pour la multiplication et la division élément par élément, on ajoute un point devant l'opérateur (.* et ./).

Exemple :

```
>> x1.*x2      % multiplication élément par élément
ans =
     2    12
>> x1./x2      % division élément par élément
ans =
    2.0000    0.7500
```

Cependant, MATLAB lance une erreur lorsque les dimensions ne concordent pas (remarquez les messages d'erreur, ils sont parfois utiles pour corriger vos programmes) :

```
>> x4=[1 3 4];
>> x5=x1.*x4
Error using .*
Matrix dimensions must agree.
```

La multiplication de deux vecteurs est donnée par (*). Ici, l'ordre a de l'importance (et la taille aussi):

```
>> v1 = [2 3];           % vecteur 1x2
>> v2 = [1 4];
>> v3 = v2';             % vecteur 2x1
>> v = v1 * v3           % (1x2) * (2x1) =
v =                       (1x1)
    14
>> v = v3 * v1           % (2x1) * (1x2) =
v =
     2     3
     8    12
```

3.13. Réaliser l'exemple ci-dessous permettant de concaténer des vecteurs,

```
>> v1 = [2 3];
>> v2 = [1 4];
>> v = [v1 v2]
v =
     2     3     1     4
```