I. Prise en main

Table of Contents

Objectifs....1
Entrer des commandes....1
Créer des variables....1
Donner un nom à une variable....2
Type de variable....3
Créer des vecteurs....4
Indexer des vecteurs....5
Matrices particulières....7
Opérations sur les vecteurs et les matrices....8
Help....8
Fonctions pré-définies....9

Objectifs

- Découvrir l'interface de MATLAB
- Donner quelques opérations mathématiques
- Manipuler des vecteurs et des matrices

Entrer des commandes

- Entrer une instruction dans la fenêtre de `Commande`, après le prompt MATLAB (`>>`)
- Taper Entrée pour exécuter l'instruction.

Exemple

```
3*5
```

```
ans = 15
```

MATLAB sauvegarde le résultat dans la variable MATLAB `ans`.

Créer des variables

Exemple

```
x = 3*5
```

```
x = 15
```

```
x = x + 1
```

```
x = 16
```

- Toutes les variables sont listées dans le `Workspace`.
- Pas de déclaration préalable des variables

Ajouter un point virgule à la fin de l'instruction.

```
y = x/2;
```

L'instruction est exécutée. La variable est créée dans le `Workspace`, mais le résultat ne s'affiche pas dans la fenêtre de `Commande`.

- Entrer le nom de la variable dans la fenêtre de `Commande`.

```
y
```

```
y = 8
```

MATLAB retourne la valeur courante de la variable.

Donner un nom à une variable

Vous pouvez donner le nom de votre choix à une variable. Quelques règles sont à respecter :

- le nom doit commencer par une lettre
- le nom contient uniquement des lettres, des chiffres ou un "underscore" (_).
-  **Pas d'espace, pas de caractère spécial !**
- MATLAB est sensible à la casse. La variable *A* est différente de la variable *a*.
- MATLAB contient des constantes et des fonctions pré-définies

Exemple

```
pi
```

```
ans = 3.1416
```

```
i
```

```
ans = 0.0000 + 1.0000i
```

```
j
```

```
ans = 0.0000 + 1.0000i
```

 **Eviter de donner le nom d'une constante ou d'une fonction pré-définie à vos variables.**

Exemple

```
pi = 120
```

```
pi = 120
```

Eviter les noms suivants :

- i, j, pi;
- length, char, size, plot, break, cos, sin, exp, image, log, sqrt;
- ...

Type de variable

La commande `whos` permet de lister toutes les variables du `Workspace` ainsi que leur type.

Exemple

```
mois = 'septembre'
```

```
mois =
'septembre'
```

```
whos
```

```
  Name         Size            Bytes  Class     Attributes

  V0           1x1                 8  double
  Vref         1x1                 8  double
  ans          1x1                16  double    complex
  codebin      1x4                 8  char
  mois         1x9                18  char
  pi           1x1                 8  double
  prompt       1x19               38  char
  x            1x1                 8  double
  y            1x1                 8  double
```

Type numérique

Nom	Description	Intervalle
double	64 bit floating point	-1.79769313486232E308 to -4.94065645841247E-324 4.94065645841247E-324 to 1.79769313486232E308
single	32 bit floating point	-3.402823E38 to -1.401298E-45 1.401298E-45 to 3.402823E38
uint8	8 bit unsigned integer	Integers from 0 to 255
int8	8 bit signed integer	Integers from -128 to 127
uint16	16 bit unsigned integer	Integers from 0 to 65535
int16	16 bit signed integer	Integers from -32768 to 32767
uint32	32 bit unsigned integer	Integers from 0 to 4294967295
int32	32 bit signed integer	Integers from -2147483648 to 2147483647

Conversion de type

Exemple

La commande `double` permet de convertir une variable de type `char` en type `double`.

```
moisdouble = double(mois)
```

```
moisdouble = 1×9
   115   101   112   116   101   109    98   114   101
```

Supprimer des variables

La commande `clear` permet de supprimer des variables

```
clear mois
clear all
```

La commande `clc` permet d'effacer toutes les sorties de la fenêtre de `Commande`.

Créer des vecteurs

- Toutes les variables MATLAB sont des *vecteurs.*
- Un scalaire est un vecteur de dimension 1 x 1 (1 ligne, 1 colonne).
- Les crochets ([]) permettent de créer des vecteurs possédant plusieurs lignes et/ou plusieurs colonnes.

Vecteur ligne de dimension 1 x n (1 ligne et `n` colonnes)

```
vect1 = [1 2 3]
```

```
vect1 = 1×3
     1     2     3
```

L'espace permet de séparer les différentes colonnes.

Pour créer des vecteurs contenant une suite de nombre, l'opérateur `:` est utilisé :

```
début : pas : fin
```

Exemple

```
x = 1:2:100
```

```
x = 1×50
     1     3     5     7     9    11    13    15    17    19    21    23    25 ···
```

Vecteur colonne de dimension n x 1 (n lignes et 1 colonne)

```
vect2 = [1;2;3]
```

```
vect2 = 3×1
     1
     2
     3
```

Le point-virgule permet de séparer les différentes lignes.

Matrice de dimension (n lignes et m colonnes)

Exemple

```
vect3 = [1 2 3;4 5 6;7 8 9;10 11 12]
```

```
vect3 = 4×3
     1     2     3
     4     5     6
     7     8     9
    10    11    12
```

L'instruction `size` permet de connaître la dimension d'un vecteur.

Exemple

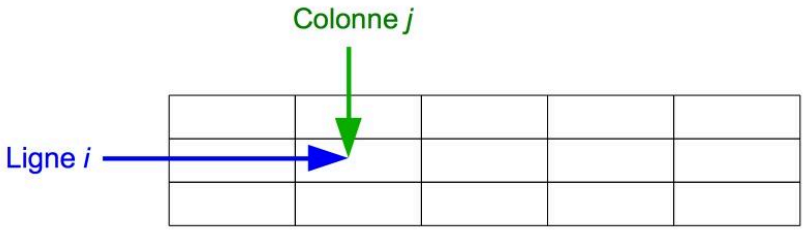
```
[nbligne nbcolonne] = size(vect3)
```

```
nbligne = 4
nbcolonne = 3
```

Indexer des vecteurs

Un vecteur est un tableau à `n` lignes et `m` colonnes.

Pour accéder à une valeur dans le vecteur, il faut préciser la ligne et la colonne correspondantes.



Extraire des valeurs d'un vecteur

Pour extraire des valeurs d'un vecteur, il suffit d'utiliser les numéros de ligne et de colonne à extraire.

Exemple

```
vect4 = vect3(2,3)
```

```
vect4 = 6
```

Exemple

```
vect5 = vect3(1:end,2)
```

```
vect5 = 4×1
     2
     5
     8
    11
```

L'instruction `end` remplace le nombre total de lignes ou le nombre total de colonnes. Ici, `end` = 4.

Exemple

```
vect6 = vect3(3,:)
```

```
vect6 = 1×3
     7     8     9
```

L'opérateur `:` permet de remplacer l'instruction : `1:end`. Ici, 1:3

Chercher des valeurs particulières dans un vecteur

L'instruction `find` permet de chercher les indices correspondant à des valeurs d'intérêt dans un vecteur.

Exemple

```
tempmoyenne = [-1.3;5.5;8.9;9.1;15.7;20.2;20.9;19.7;13.9;11.8;5.9;4];
moisannee = [1;2;3;4;5;6;7;8;9;10;11;12];
moischaud = moisannee(find(tempmoyenne==max(tempmoyenne)))
```

```
moischaud = 7
```

Concaténer des valeurs d'un vecteur

Pour ajouter une valeur sur une colonne, la concaténation se fait par un espace. Les crochets (`[]`) sont obligatoires.

Exemple

```
x = [1 2 30]
```

```
x = 1×3
     1     2    30
```

```
y = [10 x]
```

```
y = 1×4
    10     1     2    30
```

Pour ajouter une valeur sur une ligne, la concaténation se fait par un `;`

Les crochets (`[]`) sont obligatoires.

```
z = [x;4 5 6]
```

```
z = 2×3
     1     2    30
     4     5     6
```

Exemple

```
w = [x;45 5]
```

```
Error using vertcat
Dimensions of arrays being concatenated are not consistent.
```

 Les dimensions doivent être compatibles !!!

Transposer une matrice

Pour transformer les lignes en colonnes et vice versa, l'opérateur transposée est `'` :

```
mat1 = [1 2;3 4;5 6]
mat1transp = mat1'
[nbl1 nbc1] = size(mat1)
[nbl2 nbc2] = size(mat1transp)
```

Matrices particulières

- `ones(n,m)` `:` Matrice de dimension `n x m`, contenant que des 1

```
mat1 = ones(3,4)
```

- `zeros(n,m)` : Matrice de dimension `n x m`, contenant que des 0

```
mat0 = zeros(2,3)
```

- `eye(n)` : Matrice de dimension `n x n`, contenant des 1 sur la diagonale et des 0 ailleurs

```
matident = eyes(3)
```

Opérations sur les vecteurs et les matrices

Ajouter un scalaire à un vecteur

```
v1 = [2.1;3.4;5.4;6.2]
v2 = v1 + 2
```

Ajouter des vecteurs de même dimension

```
vs = v1 + v2
```

Multiplier ou diviser un vecteur par un scalaire

```
vs = 3*vs
vs = vs/3
```

 L'opérateur * réalise une multiplication matricielle. Les dimensions des matrices doivent être compatibles !!!

```
x = v1*v2
```

L'opérateur `.*` réalise une multiplication élément par élément.

```
a = [1;2;3]
b = [2;2;4]
c = a.*b
```

Opérateur puissance `^`

```
d = c.^3
```

Help

MATLAB propose une aide très bien documentée. A user et abuser sans modération !!!

Fonctions pré-définies

MATLAB contient des fonctions pré-définies (noms réservés !). La liste proposée est loin d'être exhaustive.

- **Fonctions trigonométriques**

cos, acos, sin, asin, tan, atan, exp

- **Fonctions statistiques**

min, max, mean (moyenne)**, std** (écart-type)**, var** (variance)**, mode, sum, quantile**

- **Autres fonctions**

log (logarithme népérien)**, sqrt** (racine carrée)