Introduction à Matlab (Octave)

Noria Foukia

Rapport de TP à rendre en version papier le 16 janvier 2018
(4 points pour l'implémentation des exercices, 2 points pour le rapport)

1. Introduction

L'application Matlab vous offre plusieurs fenêtres dont une fenêtre principale contenant la fenêtre de commandes avec le prompt : ». C'est dans cette fenêtre que vous entrerez toutes les commandes. Il est à noter que toutes les commandes sont en minuscules et en anglais. Lorsque l'on entre une commande, Matlab affiche systématiquement[1] le résultat de cette commande dans cette même fenêtre.
Matlab est un langage de programmation de haut niveau pour le calcul numérique. Il est particulièrement performant pour le calcul matriciel, car sa structure de données est basée sur les matrices, et il dispose de possibilités d'affichage très riches.

2. Lancer Matlab

Pour lancer Matlab, vous pouvez cliquer sur l'icône :

On va taper quelques commandes dans la fenêtre principale mais on peut aussi créer un fichier et mettre les commandes à l'intérieur. Vous pouvez par exemple, appeler ce fichier *tp1.m*

3. Utiliser l'aide

Le nombre de fonctions de Matlab étant énorme, vous devrez utiliser l'aide quasiment en permanence. Deux méthodes sont possibles pour cela, soit en mode texte, soit via l'interface graphique (par le menu). En mode texte, la commande **help** vous donne un aperçu des commandes disponibles :

Pour obtenir de l'aide, tapez :

>> help "nom de la commande"

[1] Pour supprimer l'affichage du résultat, terminez la commande avec un ; (point virgule).

Par exemples :
>> help help

ou encore :
>> help whos

ou encore
>> help sin

Pour obtenir les informations concernant une section particulières, entrez ***help section*** :

Par exemple pour Elementary math functions :
>> help elfun

Essayez toutes les commandes qui vous sont expliquées dans ce qui suit en choisissant d'autres valeurs.

4. Utilisation de base

- Pour créer et/ou attribuer une valeur à une variable, tapez *variable = valeur*.

Par exemple :
>> a = 3

- Pour supprimer l'affichage du résultat, terminez la commande avec un ; (point virgule), par exemple:

>> b=a+1;

- Pour afficher la valeur attribuée à une variable, tapez le nom de celle-ci :

>> b

b = 4

- Pour écrire plusieurs commandes, séparez-les par un , ou un ;

*>> c = 0.5 ; d = c*a , c*

d = 1.5000

c = 0.5000

- Matlab fait la distinction entre majuscules et minuscules.
- Si aucune variable n'est associée à une expression, le résultat est stocké d'office dans une variable nommée ans :

>> c+5

ans = 5.5000

- Certaines variables sont prédéfinies:

Par exemple :
>> pi

ans = 3.1416

- L'état interne de votre **session** Matlab est donné par un ensemble de variables que vous créez et modifiez au cours de la **session**. Vous pouvez afficher cet ensemble à l'aide des commandes ***who, whos*** ou par le bouton correspondant dans le menu.

- Pour effacer une variable, utilisez la commande ***clear variable*** ou ***clear all***.

5. Scalaires

- Le type de scalaire manipulé est transparent pour l'utilisateur. Ce type peut être **entier, réel ou complexe** :

Exemples :
>> a = 1
a = 1

>> b = 1.02
b = 1.0200

>> x = 1.45e4
x = 14500

>> c = 1+2.4i
c = 1.0000 + 2.4000i

La constante ***i*** est le nombre imaginaire pré-déclaré, de même que certaines constantes (***e, pi, etc.***).

6. Vecteurs

- La structure de données principale de Matlab est **le tableau** à une ou plusieurs dimensions. Pour créer un vecteur ligne, on utilise **les crochets**, et on sépare les composantes par des espaces ou des virgules :

Exemples :
>> A = [1 2 3 4 5]
A = 1 2 3 4 5

>> B = [6,7,8]

B = 6 7 8

- Une commande de la forme ***C = [a:h:b]*** crée un vecteur ligne dont les composantes sont données par par la liste :

a a + h a + 2h … a + nh où a + nh inférieur à b mais a + (n+1)h strictement supérieur à b

Exemples :
>> C = [0:1:10]

C = 0 1 2 3 4 5 6 7 8 9 10

>> D = [1:0.3:2]

D = 1.0000 1.3000 1.6000 1.9000

- On peut calculer avec les vecteurs comme avec les scalaires:

Exemples :
*>> 2*A*

ans = 2 4 6 8 10

>> A+2

ans = 3 4 5 6 7

>> D = [-1 3 4 -2 0]

D = -1 3 4 -2 0

*>> A-2*D*

ans = 3 -4 -5 8 5

- Notez le comportement particulier de la somme d'un scalaire et un vecteur ; cela revient à ajouter le scalaire à chaque composante du vecteur. Par contre, on n'a pas le droit d'ajouter des vecteurs de dimensions différentes :

Exemples :
>>A+B
??? Error using ==> +
Matrix dimensions must agree.

- Pour accéder à un ou plusieurs éléments d'un vecteur, utilisez **les parenthèses** :

Exemples :
>> B(1), A(1:3)

ans = 6

ans = 1 2 3

>> C

C = 0 1 2 3 4 5 6 7 8 9 10

>> C(2[2:4])*

ans = 3 5 7

Notez que dans la dernière commande [2:4] équivaut à 2 3 4 et 2*[2:4] équivaut à 4 6 8. Notez aussi que les indices d'un tableau commencent toujours avec un 1.

- Pour modifier un sous-ensemble d'éléments d'un vecteur on utilise:

>> A(1) = 0

A = 0 2 3 4 5

>> A(2:5) = [4 3 2 1]

A = 0 4 3 2 1

>> A([1 3 5]) = 2

A = 2 4 2 2 2

Exercice 1
Créez le vecteur [9 7 5 3 1]

Exercice 2
Créez le vecteur :
10.0000
9.5000
9.0000
8.5000
8.0000

7. Matrices

- Une matrice peut être créée en donnant la liste de ses lignes, séparées par ***;*** Dans chaque ligne, on peut séparer les coefficients par des espaces ou par des virgules.

Exemples :

```
>> F1 = [1 2 3;4 5 6]

F1 =   1   2   3
       4   5   6

>> F2 = F1+2

F2 =   3   4   5
       6   7   8

>> F3 = [1,2;3,4]

F3 =   1   2
       3   4

>> 3*F3

ans =  3   6
       9  12
```

- Notez ici aussi que d'ajouter un scalaire à une matrice revient à ajouter le scalaire à chaque coefficient de la matrice. On peut multiplier les matrices, à condition que le produit soit défini (nombre des colonnes de la première égal au nombre de lignes de la deuxième):

Exemples :
*>> F3*F1*

ans = 9 12 15
19 26 33

*>> F1*F2*
*??? Error using ==> **
Inner matrix dimensions must agree.

- L'apostrophe dénote la transposée d'une matrice

Exemples :
>> F2'

ans = 3 6
4 7
5 8

*>> F1*F2'*

ans = 26 44
62 107

- On peut aussi faire le produit de matrices de même taille élément par élément par l'opération dénotée par .* (**produit de Hadamard**).

Exemples :
>> F1

F1 = 1 2 3
4 5 6

>> F2

F2 = 3 4 5
6 7 8

*>> F1.*F2*

ans = 3 8 15
24 35 48

- On peut ajouter des colonnes ou des lignes à une matrice.

Exemples :
>> [F1;7 8 9]

ans = 1 2 3
4 5 6
7 8 9

>> [F1,[7;8]]

ans = 1 2 3 7
4 5 6 8

Exercice 3
Créez la matrice :
1 2 3 4
5 6 7 8
9 10 11 12

- On accède aux éléments d'une matrice de la manière suivante :

>> F1(1,2)

ans = 2

>> F1(2:4)

ans = 4 2 5

- Pour voir la 3-ème colonne :

>> F1(:,3)

ans = 3
6

- Pour voir la deuxième ligne :

>> F1(2,:)

ans = 4 5 6 8

- *F(i)* est le i-ème coefficient de la matrice F, où les coefficients sont numérotés colonne par colonne, de haut en bas et de gauche à droite.

Exemples :

>> F4 = [1 2 3 4 ; 5 6 7 8; 9 10 11 12]

```
F4 =   1    2    3    4
       5    6    7    8
       9   10   11   12
```

>> F4(5)

ans =   6

>> F4(2[1:6])*

ans =   5   2   10   7   4   12

Ci-dessus on a demandé les coefficients numérotés 2[1:6], c'est-à-dire :*
2 4 6 8 10 12

- Les 3 premières colonnes :

>> F4(:,1:3)

```
ans =  1    2    3
       5    6    7
       9   10   11
```

- Les 2 premières lignes :

>> F4(1:2,:)

```
ans =  1    2    3    4
       5    6    7    8
```

- Il existe des **fonctions** pour créer des matrices particulières :

Exemples :
>> eye(3)

```
ans =  1    0    0
       0    1    0
       0    0    1
```

>> ones(3,2)

ans =

1 1
1 1
1 1

>> zeros(2,3)

ans =

0 0 0
0 0 0

- Pour modifier les éléments d'une matrice, on utilise :

>>F4(2,3)=0

F4 =

1 2 3 4
5 6 0 8
9 10 11 12

>> F4(3)=0

F4 =

1 2 3 4
5 6 0 8
0 10 11 12

>> F4(2[1:6])=0*

F4 =

1 0 3 0
0 6 0 8
0 0 11 0

Exercice 4

Créez une matrice de taille 9x9 représentant un quadrillage de 0 (pour noir) et 1 (pour blanc). Commencez par créer une matrice n'ayant que des 1. Modifiez ensuite tous les éléments de numéro impair par des 0 (par une seule commande si possible).

- Les fonctions usuelles comme *sin, cos, tan, exp, tanh*, etc., peuvent être directement appliquées à tous les éléments d'un tableau; le résultat est un tableau contenant les valeurs prises par ces fonctions sur le coefficient du tableau de départ.

Exemples :
>> F3

F3 =

```
    1    2
    3    0
```

>> exp(F3)

ans =

```
    2.7183    7.3891
   20.0855         1
```

Exercice 5

Calculez la liste des valeurs prises par la fonction (sin(x)-x)/exp(x) en donnant à x toutes les valeurs de -2 à 2 espacées de 0.1.

8. Chaînes de caractères

Les chaînes de caractères se manipulent comme des vecteurs. Elles sont déclarées avec des guillemets simples '

Exemples :
>> s='Hello'
s = Hello
>> s(2)
ans = e

9. Opérations mathématiques élémentaires

Les opérations standards sur les scalaires sont **: *addition +, soustraction -, multiplication *, division /, puissance ^*. La racine carrée** s'obtient par la fonction **sqrt**. On dispose de toutes

les fonctions usuelles sur les scalaires : faire **help elfun** pour de plus amples détails. Attention, les fonctions peuvent renvoyer des complexes.

Exemples :

```
>> sqrt(- 1)
ans =
0 + 1.0000i
```

- En ce qui concerne les vecteurs et matrices ces opérateurs se prolongent au sens du calcul vectoriel et matriciel. En particulier, il faut veiller à la compatibilité des tailles des objets entre eux (déjà vu plus haut).

Exemples :

```
>> u=1:3
u =
1 2 3

>> v = [1  0  - 1]
v =
1         0        - 1


>>  u+v
ans =
2         2         2

>>  v'
ans =
  1
  0
- 1

>>  u*v'
ans =
- 2

>> v'*u
ans =
  1         2         3
  0         0         0
- 1        - 2       - 3
```

Il est également possible de multiplier une matrice par un scalaire (déjà vu plus haut)

Exercice 6
Créez la matrice :

0	4	4	4
4	0	4	4
4	4	0	4
4	4	4	0

Exercice 7

Créez la matrice suivante par concaténation de matrices (en utilisant ce qui a été vu avant) :

1.0000	3.4000	0	0	0	0	5.0000
1.0000	0	3.4000	0	0	0	6.0000
1.0000	0	0	3.4000	0	0	7.0000
1.0000	0	0	0	3.4000	0	8.0000
1.0000	0	0	0	0	3.4000	9.0000

Exercice 8

Créez la matrice :

1	0	7	0
0	1	7	0
0	0	7	0
0	0	7	1

Exercice 9
Inversez les deux colonnes centrales dans la matrice précédente pour obtenir :

1	7	0	0
0	7	1	0
0	7	0	0
0	7	0	1

10. Graphiques

Une nouvelle figure est créée par la commande ***figure***

Si A et B sont des vecteurs de même dimension donnée par n, la commande ***plot(A,B)*** dessine les segments de droite joignant les points de coordonnées (A(i),B(i)), pour i = 1,..., n.

Par exemple :
>> A=[1 2 3 4]

A =

1 2 3 4

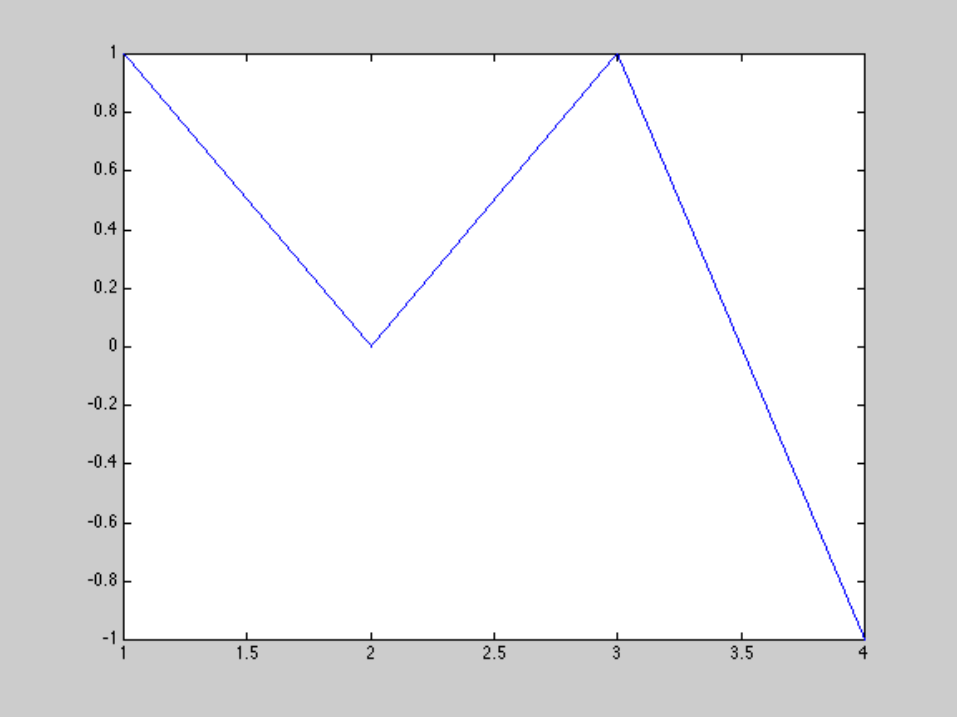
>> B=[1 0 1 -1]

B =

1 0 1 -1

>> plot(A,B)

donne comme résultat :



- On peut spécifier le style d'affichage

Exemple :

>> plot(A,B,'g')*

dessine des **** verts (g=green)*** aux points (A(i),B(i)), sans les joindre par des segments. Tapez ***help plot*** pour voir d'autres styles.

- Pour avoir plusieurs graphiques sur une même figure on fait :

>> hold on
>> plot(C,D)
>> hold off

- Pour effacer une figure

>> clf

- Pour la fermer figure on utilise la commande ***close***

- Pour tracer le graphe de la fonction sin(x), on peut par exemple écrire :

*>>A=[0:0.01:2*pi];plot(A,sin(A))*

Exercice 10

Afficher dans un graphique e quadrillage de 0 et de 1 fait précédemment en noir et blanc.

- **NOTE : pour sauvegarder les variables de votre session Matlab tapez dans la fenêtre de commandes :**

>> save tp1

Attention suivant l'endroit où vous avez créé tp1.m, il faudra peut-être indiquer le chemin dans la commande (voir l'aide)

- Pour les retrouver pendant une autre session, tapez :

>> load tp1 (avec le chemin)

Références

David Filliat, Introduction à Matlab, ENSTA (France)
Patrick Ciarlet et Eric Lunéville : http://www.ensta.fr/~ciarlet/Doc-Matlab/Doc-Matlab-Couleur.pdf
Bastien Chopard , Felice Ronga, Cours de mathématiques pour Informaticiens, Introduction a Matlab, CUI, Uni Dufour, Genève