HUB: Introduction à Matlab

Un plateforme scienfique

Cours
HARAKALY Robert

1 Introduction

Matlab pour «MATrix LABoratory», est un logiciel conçu pour fournir un environnement de calcul numérique de haut niveau. Il est particulièrement performant pour le calcul matriciel car sa structure de données interne est basée sur les matrices. Il dispose également de grandes capacités graphiques pour, par exemple, la visualisation d'objets mathématiques complexes. Son fonctionnement repose sur un langage de programmation interprété qui permet un développement très rapide. Pour des applications nécessitant un temps de calcul plus élevé, un langage compié comme le $C++$, le fortran ou autre, est mieux adapté.

2 Premiers pas dans Matlab

2.1 Commandes et calculs de base

Matlab est un interpréteur. Cela veut dire que l'utilisateur rentre des commandes et Matlab les exécute ligne par ligne.
Exemple:

» $2+2$
ans =
4

Le symbole [»] indique à l'utilisateur où il faut rentrer la commande. Matlab fournit le résultat de la commande sous la ligne contenant ans =.
L'exécution d'une ligne provoque automatiquement l'affichage des résultats sous forme d'une liste de donnés numériques. Cette fonctionnalité peut être bloquée en mettant un ";" à la fin de chaque ligne de programme.
Toute suite de caractères suivant le symbole [%] est ignorée par Matlab pour commenter.

» $2+2$ %Une première commande
ans =
4

» $3 * 5$ %Une deuxième commande
ans =
15

Matlab possède de nombreuses constantes mathématiques et fonctions prédéfinies utiles en mathématiques que nous allons étudier au cours de ces travaux pratiques.

» pi %Le rapport de la circonférence d'un cercle à son diamètre.
ans =
3.1416
» $sin(pi/6)$ %Un exemple d'évaluation d'une fonction
ans =
0.5000
» $log(1.5)$ %Un second exemple
ans =
0.4055
» $i\hat{}2$ %i est l'unité imaginaire
ans =
-1

2.2 Variables

Il peut parfois être utile de stocker une valeur dans une variable pour l'utiliser plus tard. Les variables s'utilisent sans déclaration préalable de type, Matlab adoptant en interne l'objet matrice pour toutes les variables (scalaires ou non, réelle ou imaginaire, chaîne de caractères...) . Le type (dimensionnement de la matrice) est donc géré automatiquement d'après l'usage (affectation) de la variable.

» A=23 % On assigne la valeur 23 à la variable A.
A =
23

Les variables peuvent contenir une chaîne de caractères (inclue entre des guillement simples [']).

» A='coucou' % On enregistre une chaîne de caractères dans la variable A.
A =
'coucou'

3 Travail avec les m-files

Pour l'élaboration de programmes complexes, il n'est pas pratique de taper les commandes une à une dans la Command Window. Nous allons voir comment regrouper des

commandes dans un fichier appelé un m-file. Le nom m-file vient de l'extension de ces fichiers (.m).

3.1 Scripts

En utilisant l'onglet Current Folder, placez-vous dans votre répertoire personnel. Vous pouvez ensuite créez ensuite un répertoire pour vos travaux Matlab, par exemple hub (dans l'onglet Current Folder, bouton droit de la souris et New -> Folder).
Créez ensuite un script (onglet Editor, New -> Script). Apparaît alors une nouvelle fenêtre ressemblant à un éditeur de texte, c'est l'Editor.
Comme d'habitude, le symbole [%] permet d'ajouter des commentaires.

```
% My first m-file
A=ones(4) % Une Matrice 4x4 remplie de 1
v=[1 2 3 4]' % Un vecteur colonne
w=A*v % Transformation du vecteur par la matrice.
```

Quand vous avez fini décrire le script, enregistrez le fichier dans le répertoire, par exemple, hub.m. Tout programme enregistré sous l'extension .m peut être lancé depuis la Commande Windows de Matlab, en frappant son nom.

```
» hub
A =
1   1   1   1
1   1   1   1
1   1   1   1
1   1   1   1
v=
1
2
3
4
w=
10
10
10
10
```

3.2 Fonctions

Dans un m-file, on peut définir des fonctions après la dernière commande du script, en utilisant le mot-clef function.
La définition de la fonction doit se terminer par le mot-clef end.
Le script ci-dessous utilise une fonction, moyenne, qui prend un entier comme argument

et retourne ensuite la valeur moyenne des éléments de matrice aléatoire de taille n x n.

n = 4; % On choisit une valeur pour n.
moyenne(n) % On utilise la fonction moyenne() définie ci-dessous.

function m = moyenne(n)
A = rand(n); % Une matrice aléatoire de taille n x n
imagesc(A); % Affiche les ?el?ements de matrice sous forme de couleurs
m = mean(mean(A)); % Calcule les moyennes des colonnes, puis la moyenne du vecteur ainsi obtenu. Ceci est équivalent à calculer la moyenne de tous les éléments de matrice.
end % Termine la définition de la fonction

La forme générale de la déclaration d'une fonction est

function 'nom variable retour' = 'nom fonction' ('noms paramètres')
Donc dans cette exemple, m = nom variable retour et moyenne = nom fonction

A noter aussi que la fonction se Termine avec la commande "end"!

3.3 Expressions booléennes et Conditions - if . . . else . . . end

En Matlab le test d'égalité se fait à l'aide de [==] et celui d'inégalité à l'aide de [~=]. Ces tests retournent une valeur booléenne suivant la véracité de l'expression. On peut combiner des variables et valeurs booléennes au moyen des opérateurs logiques "et" [&] et "ou" [|].

Pour l'utilisation des commandes 'if', il faut aussi penser à terminer avec la command end, voici un exemple.

if (test)
(commandes)
elseif (test)
(commandes)
...
else (test)
(commandes)
end

3.4 Boucles - while - for . . . end

Pour les boucles "while" ou "for" c'est le même principe que la commande "if", il Faut penser a terminer avec le end, de plus il n'y a pas besoin de mettre l'ittération comme "i++", cela se fait tout seul. Exemple:

for k = liste

[commande]
end

while test
[commande]
end

4 Vecteurs et matrices

La structure de données de base de Matlab est la matrice; même un nombre est considéré comme une matrice 1 x 1. Toutes les fonctions et opérations relatives aux matrices sont très optimisées et sont à utiliser aussi souvent que possible.

4.1 Création

Une matrice est délimité par des crochets. On sépare les colonnes par des espaces et les lignes par des points-virgules.

```
» A = [1 1 1 ; 2 2 2] % Une matrice de taille 2 x 3
A=
1    1    1
2    2    2
» B=[1 ; 2 ; 3] % Un vecteur
B=
1
2
3
» C=[1 2 3] % Une matrice de taille 1 x 3
C=
1    2    3
```

Matlab propose des commandes pour créeer certaines matrices particulières simplement.

Commande	Description
ones(n, m)	Matrice de taille n x m ne contenant que des 1.
zeros(n, m)	Matrice de taille n x m ne contenant que des 0.
eye(n, m)	Matrice de taille n x m contenant des 1 sur la première diagonale et des 0 ailleurs.
diag(v)	Matrice diagonale où les éléments de la diagonale sont les composantes du vecteur v, et dont les éléments hors diagonale sont nuls.
rand(n, m)	Matrice de taille n x m contenant des nombres aléatoires uniformément répartis entre 0 et 1.

Matlab dispose également de moyen très simple pour créer des listes. La commande [a:h:b] crée une liste. Le cas particulier [a:b] est un raccourci pour [a:1:b].

X = [1:2:10] % Un vecteur dont les éléments vont de 1 à 9, par incréments de 2.
X=
1 3 5 7 9

4.2 Accès et modifications

On présente dans cette section diverses méthodes pour accéder et modifier les éléments d'une matrice. Dans la table qui suit, A désigne une matrice de taille n x m, $1 \leq k \leq n$ et $1 \leq l \leq m$ sont des nombres entiers et M une matrice d'entiers positifs inférieurs ou égaux à nm.

Commande	Description
A(k, l)	Renvoie l'élément se trouvant à la $k^{ème}$ ligne et la $l^{ème}$ colonne.
A(k)	Renvoie le $k^{ème}$ élément d'une matrice. En Matlab, les éléments d'une matrice de taille n x m sont indexés de 1 à nm de haut en bas et de gauche à droite.
A(M)	Renvoie une matrice de même taille que M dont chaque élément A(k) a pour valeur A(M(k)). (Le $k^{ème}$ élément de M est donc un indice, déterminant un élément de la matrice A 'a placer en position k dans A(M).)
A(k,:)	Renvoie la $k^{ème}$ ligne de la matrice.
A(:,l)	Renvoie la $l^{ème}$ colonne de la matrice.
A(k:l,p:q)	Renvoie sous-matrice de A composée de l'intersection des lignes k à l (inclue) et des colonnes p à q (inclue).

Pour modifier les éléments d'une matrice, on utilise les mêmes commandes que ci-dessus. On ajoute à la commande le signe [=] et la nouvelle valeur.

```
» A % Notre Matrice
A =
1    2    3    4
12   13   14   15
» A(2, 2) = 999 % On assigne la valeur 999 à l'élément de matrice en position (2, 2)
A =
1    2    3    4
12   999  14   15
```

4.3 Opérations avec les matrices

Opérations de bases. Dans ce qui suit, A et B sont des matrices et c est un nombre. On note les éléments de matrices de A et de B par (a_{ij}) et (b_{ij}).

Commande	Description
A + B	Matrice dont l'élément (i, j) est $a_{ij} + b_{ij}$. A et B doivent avoir les mêmes dimensions.
A + c = c + A	Matrice dont l'élément (i, j) est $a_{ij} + c$
A - B	Matrice dont l'élément (i, j) est $a_{ij} - b_{ij}$. A et B doivent avoir les mêmes dimensions.
A - c	Matrice dont l'élément (i, j) est $a_{ij} - c$.
c - A	Matrice dont l'élément (i, j) est $c - a_{ij}$.
A * B	Matrice dont l'élément (i, j) est $\sum_j a_{ik} b_{kj}$. C'est le produit matriciel standard. Le nombre de colonnes de A doit être le même que le nombre de lignes de B.
A * c (ou c*A)	Matrice dont l'élément (i, j) est ca_{ij}. C'est la multiplication d'une matrice par un scalaire.
A.*B	Matrice dont l'élément (i, j) est $a_{ij} b_{ij}$. C'est la multiplication élément par élément, peu utilisée en algèbre linéaire, mais souvent pratique. A et B doivent avoir les mêmes dimensions.
Aˆn ($n \in \mathbb{Z}_+$)	A * A * . . . * A (n fois); A doit être carrée.
Aˆ(-1)	Matrice inverse, telle que A * Aˆ(-1) = Aˆ(-1) * A est la matrice identité. A doit être inversible.
Aˆ(-n)	Aˆ(-1) * Aˆ(-1) * . . . * Aˆ(-1) (n fois); A doit être inversible.
A.ˆB	Matrice dont l'élément (i, j) est $a_{ij}^{b_{ij}}$. A et B doivent avoir les mêmes dimensions.
A'	Transposition et conjugaison complexe.
A.'	Transposition. A.' = A' dans le cas o u A est réelle.
B/A	Matrice X telle que XA = B. Si A est inversible, alors $X = BA^{-1}$. Le nombre de colonnes de A doit être le même que le nombre de colonnes de B.
A\B	Matrice X telle que AX = B. Si A est inversible, alors $X = A^{-1}B$. Le nombre de lignes de A doit être le même que le nombre de lignes de B.
A./B	Matrice dont l'élément (i, j) est a_{ij}/b_{ij}. A et B doivent avoir les mêmes dimensions.
A.\B	Matrice dont l'élément (i, j) est b_{ij}/a_{ij}. A et B doivent avoir les mêmes dimensions.
A/c	Matrice dont l'élément (i, j) est a_{ij}/c.

Important. Pour la résolution de systèmes d'équations AX = B ou XA = B, utilisez toujours les commandes A\B ou B/A. Il n'est pas nécessaire d'inverser la matrice A pour r'esoudre le système. L'inversion de A est coûteuse en temps de calcul et moins précise que la résolution directe du système.

Fonctions sur les matrices. Nous présentons ici quelques fonctions définies dans Matlab prenant comme paramètres des matrices. Pour plus d'information, tapez help suivi du nom de la fonction. Dans le tableau qui suit, A est une matrice et v est un vecteur.

Commande	Description
det(A)	Déterminant de A; la matrice A doit être carrée.
trace(A)	Trace de A pour une matrice A carrée.
rank(A)	Rang de A (la dimension de l'image de l'application associée à A).
null(A)	Base du noyau de A. L'argument supplémentaire 'r' donne une meilleure base (voir help null).
diag(A)	Diagonale principale de A.
norm(v)	Norme euclidienne de v. Il est aussi possible de calculer d'autres normes, voir help norm.
mean(A)	Liste (vecteur ligne) contenant la moyenne des éléments de chaque colonne.
sum(A)	Liste contenant la somme des éléments de chaque colonne.
prod(A)	Liste contenant le produit des éléments de chaque colonne.
max(A)	Liste contenant la valeur maximale de chaque colonne.
min(A)	Liste contenant la valeur minimale de chaque colonne.
length(A)	Nombre d'éléments du vecteur v. Appliqué à une matrice A, length(A) retourne le maximum entre le nombre de lignes et de colonnes.
eig(A)	Liste des valeurs propres de A.
[M, D] = eig(A)	Enregistre sous la forme de variables une matrice diagonale D contenant les valeurs propres de A et une matrice M contenant comme colonnes les vecteurs propres de A.

Toutes les fonctions mathématiques classiques (cos, sin, log, exp, etc...) peuvent également être appliquées à une matrice A. Le résultat est la matrice obtenue de A en appliquant la fonction séparément à chaqun des éléments de matrice de A. Ceci est pratique pour calculer rapidement une fonction sur un ensemble de valeurs.

5 Graphisme

5.1 Courbes dans le plan

5.1.1 Courbe

Etant donnés deux vecteurs de même taille, x et y, la fonction plot(x,y) dessine les points de coordonnée (x(k),y(k)) pour $1 \leq k \leq$ length(x).. Par défaut, Matlab

relie les points par des segments de droite.

» x = [-1 -.5 -1 0 1 .5 1 0 -1]; % Un vecteur de coordonnées horizontales.
» y = [-1 0 1 .5 1 0 -1 -.5 -1]; % Un vecteur de coordonnées verticales.
» plot(x, y) % On dessine les neuf points du plan de coordonnées (x,y), reliés par des segments de droite.

La figure dessinée apparaît dans une nouvelle fenêtre.

5.1.2 Graphe d'une fonction

En prenant un grand nombre de points régulièrements espacés dans le vecteur x et en définissant ensuite y = f(x) pour une fonction f, la fonction plot(x,y) dessine le graphe de la fonction f.

» x=[0:0.01:4*pi]; % Un vecteur dont les éléments vont de 0 à 4*pi, par pas de 0.01.
» y = cos(x); % Un vecteur dont les éléments sont les cosinus des éléments de x.
» plot(x, y)

5.1.3 Plusieurs courbes

» z = sin(x); % Un vecteur dont les éléments sont les sinus des éléments de x.
» plot(x, y)
» plot(x, z)

La suite de commandes ne trace pas deux graphes, mais un seul. En fait, le deuxième plot(x,z) vient effacer et remplacer le premier plot(x,y). Pour remédier à cela, Matlab propose plusieurs méthodes suivant si l'on désire que les courbes apparaissent dans une ou plusieurs fenêtres.
Pour voir les graphiques sur deux fenêtres, il suffit de dire à Matlab de construire une nouvelle fenêtre avec la commande figure.

» plot(x, y)
» figure % Demande à Matlab de dessiner sur une nouvelle fenêtre.
» plot(x, z) % Dessine les points de coordonnées (x,z) dans la nouvelle fenêtre

Pour avoir les deux courbes dans la même fenêtre, il existe deux méthodes équivalentes:
soit avec les commandes hold on et hold off,

» hold on,plot(x,y),plot(x,z),hold off % Après hold on, Matlab dessine dans la même fenêtre, sans effacer les dessins précédents. La commande hold off annule la

commande hold on pour les prochains dessins.

soit en donnant plus de paramètres à la commande plot.

» plot(x,y,x,z)

5.2 Affichage de surfaces

Pour la visualisation de surfaces en 3 dimension données par des équations du type z = f(x, y), Matlab met a disposition deux fonctions: mesh et surf. La seule différence entre les deux vient du rendu graphique: mesh affiche la surface en fil-de-fer et surf en surface remplie.

```
» [x, y]=meshgrid(-3:0.1:3,0:0.2:5); % Crée une grille
» z = x.^2 - y.^2;
» surf(x,y,z) % Dessine la surface correspondante.
```

La fonction meshgrid(-3:0.1:3,0:0.2:5) crée deux matrices x et y de taille 26 x 61, que l'on peut voir comme les coordonnées (x, y) de points d?une grille de 26 x 61 points. Les coordonnées x vont de -3 à 3 par pas de 0.1 et les coordonnées y vont de 0 à 5 par pas de 0.2.
On calcule alors la fonction z sur chaque point de la grille à partir de ses coordonnées, puis on dessine la surface au moyen de surf.

5.3 Affichage de matrices

Étant donnée une matrice A de taille n x m, Matlab propose une méthode, imagesc, pour visualiser le contenu de A. Matlab dessine un rectangle partagé en n x m petits rectangles où la couleur du rectangle (i, j) d'epend de la valeur de l'élément a_{ij} de la matrice.

```
» a=ones(11,11) % Crée une matrice 11 x 11 remplie de 1.
» a([1:2:121])=0 % Change les éléments de numéro pair à 0.
» imagesc(a), axis equal % Visualise a.
```