TP 01: Logiciel MATLAB

1 Introduction

Le logiciel MATLAB, développé par la société The MathWorks (`http://www.mathworks.com`), est devenu un langage de référence pour l'analyse et la résolution de problème scientifique. Il intègre à la fois des solutions de calcul, de visualisation et un environnement de développement. Son nom provient de MATrix LABoratory,

MATLAB a de nombreux avantages par rapport aux langages de programmation traditionnels (tel que le C/C++). Il permet le developpement interactif des scripts à l'aide d'une invite de commandes et l'utilisation des fichiers code-sources (i.e;, programmmes et fonctions), MATLAB est un langage interprété. La structure de données de base est le tableau ne necessitant pas de dimensionnement. Il fournit de nombreuses fonctions et procedures regroupées en boîtes à outils (toolbox) pour de nombreux domaines (par ex : signal processing , statistics, control theory, optimization, ...)

De plus, MATLAB dispose d'une excellente documentation.

2 Démarrer MATLAB

Lorsque vous lancez MATLAB pour la première fois, l'écran ressemble à celui de la Figure 1. Le 'bureau' MATLAB est une fenêtre contenant d'autres sous-fenêtres. Les principaux outils disponibles depuis ce bureau sont :

- COMMAND WINDOW : invite de commande permettant de taper des instructions, d'appeler des scripts, d'exécuter des fonctions MATLAB.
- COMMAND HISTORY : historique des commandes lancées depuis l'invite de commande.
- WORKSPACE : il liste les variables en mémoire, il permet également de parcourir graphiquement le contenu des variables
- CURRENT DIRECTORY : un navigateur de fichier intégré à MATLAB pour visualiser le répertoire de travail courant et y effetuer les opérations classiques tel que renommer ou supprimer un fichier.
- le HELP BROWSER : un navigateur permettant de parcourir l'aide de MATLAB. L'aide est un outil précieux pour trouver les fonctions et apprendre leur fonctionnement (notamment le format des données à fournir en entrée ainsi que les valeurs renvoyées par la fonction).

Par la suite, il est conseillé de tester toutes les instructions précédées de `>>` dans la *command window*. Lorsque vous optenez une erreur, essayez d'en comprendre la signification. Avec un peu de pratique, vous verrez que les messages d'erreur sont en général explicites.

3 MATLAB comme calculette

Comme tout langage de programmation, MATLAB dispose de fonctions de calculs mathématiques. Nous en voyons ici quelques exemples d'utilisation.

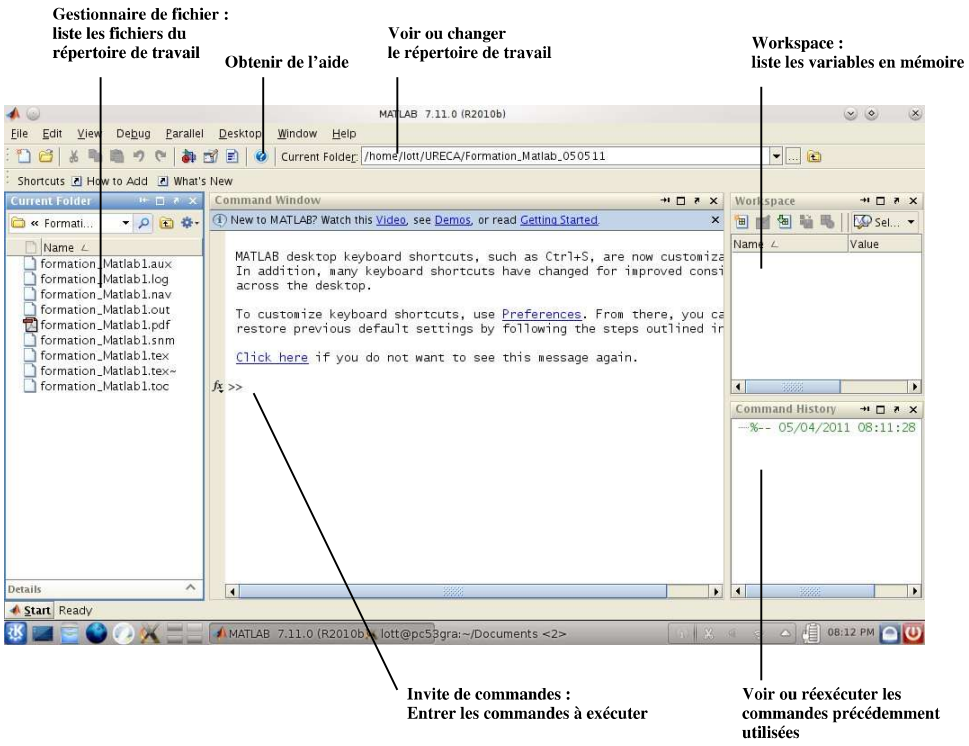

FIGURE 1 – L'interface graphique de l'environnement MATLAB .

+ (addition)	⇒	>> 3+2
- (soustraction)	⇒	>> 3-2
* (multiplication)	⇒	>> 3*2
/ (division)	⇒	>> 3/2
^ (puissance)	⇒	>> 3^2

Si l'on regarde ce que donne l'exemple d'addition :

```
>> 3+2
ans =
    5
```

On peut remarquer que lorsqu'aucune variable d'affectation n'est spécifiée pour stocker le résultat de l'opération, MATLAB stocke le résultat dans une variable appelée `ans` (diminutif pour `answer`). Cette variable sera écrasée à chaque nouvelle opération. Pour éviter cela, on peut spécifier le nom d'une variable pour stocker le résultat. On pourra alors réutiliser le résultat de l'opération plus tard. Par ex :

```
>>x = 3+2
x    =
    5
```

x prend alors la valeur 5 . Cette variable peut alors être réutilisée dans un calcul suivant :

```
>>x/2
ans =
    2.5000
```

3.1 Priorité des opérateurs

Les opérations sont évaluées en donnant la priorité aux opérateurs selon l'ordre suivant :

1. ()
2. ^
3. * /
4. + -

Exemple 1 :

```
>> 3 + 2 * 4 ^ 2
           -----
             16
        ---------
            32
   -------------
      ans = 35
```

1) Evalutation de `4^2` car `^` a la priorité sur les autres opérateurs `* et +`

2) Evalutation de `2*16` car `*` a la priorité sur `+`

3) Pur finir, évalutation de `3+32`

Exemple 2 :

```
>> ( ( 3 + 2 ) * 4 ) ^ 2
     ---------
         5
   ----------------
           20
   --------------------
        ans = 400
```

1) Evalutation de `3+2` car `()` a la priorité sur `*`

2) Evalutation de `5*4` car `()` a la priorité sur `^`

3) Pour finir, évalutation de `20^2`

A titre d'exercice, évaluer les expressions suivantes :

1) $\frac{123*456}{123+456}$, vous devriez obtenir .....

2) $2\frac{\frac{1}{3}+\frac{1}{5}+\frac{1}{6}}{\frac{2}{3}+\frac{4}{5}+\frac{5}{6}}$, vous devriez obtenir .....

3) $\frac{1}{12-6^2}+\frac{2}{3}$, vous devriez obtenir .....

4) $\frac{1+(\frac{2}{3})^{\frac{1}{2}}}{12-6^2}$, vous devriez obtenir .....

4 Manipuler des variables

Une variable est un emplacement en mémoire permettant de stocker provisoirement une donnée. On réfère à l'emplacement en mémoire par le nom que l'on donne à la variable. On pourra utiliser le contenu de la variable (dans un calcul par exemple si la donnée stockée est une valeur numérique) ou modifier la donnée stockée à l'emplacement mémoire.

On distingue plusieurs types de variable selon les données qu'elles servent à stocker (nombre, caractère alphanumérique, tableau, matrice, structure). Contrairement à d'autres langages de programmation, sous MATLAB le type des variable n'a pas besoin d'être spécifié, MATLAB infère le type d'une variable en fonction de la donnée que l'on y stocke.

Sous MATALB, les noms de variable doivent commencer par une lettre, sont sensibles à la casse (différenciation des caractères majuscule/minuscule) et ne peuvent contenir aucun caractère spécial excepté le tiret bas (_ , *underscore*). De même, il faut éviter d'utiliser comme nom de variable des noms déjà employés comme nom de fonctions (par ex : min, max, exp ...). MATLAB générera également une erreur si un des mots-clés réservés

suivant est utilisé : `for`, `end`, `if`, `while`, `function`, `return`, `elseif`, `case`, `otherwise`, `switch`, `continue`, `else`, `try`, `catch`, `global`, `persistent`, `break`. Ces mots-clés font partie de la syntaxe du langage MATLAB, nous verrons dans la suite des exemples d'utilisation de certains de ces mots-clefs.

4.1 Utilisation d'une variable

Les variables sont créées lors de la première affectation (opération qui permet d'attribuer une valeur à une variable).

A $\leftarrow$ 0 *on affecte la valeur 0 à la variable A, si A n'existait pas auparavant, elle est créée.*
B $\leftarrow$ 11 *on affecte la valeur 11 à la variable B*

On peut alors utiliser une variable dans un calcul ou pour affecter son contenu à une autre variable.

A $\leftarrow$ B *on affecte la valeur contenue ds B à la variable A, A vaut à présent 11, B reste inchangée*
A $\leftarrow$ A + 1 *on affecte la valeur contenue ds A incrémentée de 1 à la variable A, A vaut à présent 12*

Sous MATLAB, l'affectation se fait à l'aide de l'opérateur `=`

```
>> A = 0
>> B = 11
>> A = B
>> A =  A + 1
```

4.2 Quelques types de variables : nombres, vecteurs (tableau 1D), matrices (tableau 2D)

- Nombres :

```
>>a = 0.66
>>deuxiemeNombre = 2/3
>>mon_nom_de_variable = -4
```

- Tableau 1D en ligne :

```
>>rowvect1 = [1,2,3]
```

On sépare les éléments du vecteur par une virgule pour obtenir un agencement en ligne

```
>>rowvect2 = [a , mon_nom_de_variable , 12]
```

Nous avons ici utilisé le contenu des variables `a` et `mon_nom_de_variable` pour affecter les 2 premiers éléments de la matrice `rowvect2`. Le résultat sera `rowvect2 = [0.66 , -4 , 12]`

- Tableau 1D en colonne :

`>>colvect = [1;2;3]` (on sépare les éléments par un point virgule pour obtenir un agencement en colonne)

- Matrice :

```
>>maPremiereMatrice = [1,2,3;4,5,6;7,8,9] %creation d'une matrice de dimension (3x3)
```

(les élements d'une même ligne sont séparés par des virgules, les lignes de la matrice sont séparées par un point virgule)

Le caractère `%` permet de spécifier un commentaire dans le code : ce qui suit sur une ligne le caractère `%` ne sera pas interprété par MATLAB. Les commentaires dans un code source d'un programme informatique sont en général destinés à un lecteur humain pour aider à la compréhension du programme. Vous n'avez pas besoin de recopier le commentaire dans les exemples sur une ligne de ce tutoriel.