Nom : **Date :**
Prénom :

Faculté des sciences et ingénierie (Toulouse III) Année universitaire
Département de mathématiques – L3 MMESI 2012-2013
Analyse numérique I

TP n° 2 – Initiation à l'algorithmique

---

1 Découverte de MATLAB (2) : les scripts

Pour comprendre l'intérêt des scripts, effectuer les calculs suivants en ligne de commande :

- créer la matrice $A = \begin{pmatrix} 1 & 2 \\ i & 4 \end{pmatrix}$;
- créer $B = A^{\mathrm{T}}$;
- créer $S = A + B$;
- tester si la matrice S est symétrique $(S - S^{\mathrm{T}} = 0)$;

Refaire ces calculs pour $\widetilde{A} = \begin{pmatrix} 1 & 2 \\ 3i & 4 \end{pmatrix}$.

Cela devient vite pénible de devoir compiler successivement tous les calculs en ligne de commande, même lorsqu'une petite modification est à faire. De plus, ces calculs sont liés les uns aux autres : il serait préférable de pouvoir écrire toutes ces lignes puis, ensuite, de *tout compiler d'un coup*. Les scripts vont répondre à ce besoin.

Effectuer la procédure suivante afin de créer votre premier script MATLAB.

1. Créer un dossier `L3_analyseNumerique` *(en cliquant sur l'icône "Open file" puis "Create New Folder"* [1] *)*.
2. Créer un sous-dossier `TP02`.
3. Dans ce sous-dossier, créer un fichier `Intro.m` *(en cliquant sur l'icône "New M-file")* et y reproduire les calculs ci-dessus.

4. Exécuter le fichier `Intro.m` ; indiquer ce qu'il faut dire à MATLAB.

5. Que suffit-il de faire pour réexécuter les calculs mais, cette fois-ci, sans afficher le résultat de la création de B ?

1. S'il n'est pas possible de renommer le dossier créé à partir du MATLAB de l'université, en créer un second… qu'il sera posssible de renommer (!).

2 Structures itératives et conditionnelles (for, while, if)

2.1 Structures itératives (ou répétitives)

Utilité : faire se répéter une (ou plusieurs) instruction(s).

2.1.1 Boucle for

Elle permet de faire se répéter un certain nombre de fois une (ou plusieurs) instruction(s) ; le "nombre de fois" est explicitement connu de l'utilisateur.

Créer un script `exemples_for.m` pour y tester les exemples suivants.

Exemple 1.

```
Pour i=0 à i=5, faire
  afficher i
Fin
```

```
for i=0:5
  i
end
```

Exemple 2. Compléter la case vide (cadre grisé : code MATLAB attendu).

```
Pour i=0 à i=10, faire
  afficher i
Fin
```

Exemple 3.

```
Pour j=0 à j=10 par pas de 2, faire
  afficher j
Fin
```

```
for j=0:2:10
  j
end
```

Exemple 4. Compléter la case vide (cadre blanc : algorithme en français attendu).

```
for j=5:-1:0
  j
end
```

Qu'est-ce qui est renvoyé par MATLAB ?

Exercice 1. Écrire un script calculant les valeurs de $\cos(0), \cos\left(\frac{\pi}{3}\right), \cos\left(\frac{2\pi}{3}\right), \ldots, \cos(2\pi)$. (Dans la case blanche est attendu un algorithme en français ; dans la case grisée, le code MATLAB correspondant.)

2.1.2 Boucle while

Elle permet de faire se répéter un certain nombre de fois une (ou plusieurs) instruction(s) ; le "nombre de fois" n'est pas explicitement connu de l'utilisateur : il dépend d'une condition. Autrement dit : *tant que* la condition est vérifiée, on répète les instructions ; dès qu'elle ne l'est plus, on s'arrête.

Créer un script `exemples_while.m` pour y tester les exemples suivants.

Exemple 5.

```
i=0
Tant que i<5, faire
  i=i+1
Fin
```

```
i=0
while (i<5)
  i=i+1
end
```

Exemple 6. Compléter la case vide.

```
i=5
Tant que i>0, faire
  i=i-1
Fin
```

Exemple 7. Écrire l'algorithme correspondant au code MATLAB suivant.

```
i=5;
while (i<=0)
  i=i-1
end
```

Qu'est-ce qui est renvoyé par MATLAB ? pourquoi ?

Exemple 8. Expliquer le résultat obtenu sur cet exemple, ainsi que la syntaxe « == ».

```
i=5
Tant que i est égal à 5, faire
  i=i-1
Fin
```

```
i=5
while (i==5)
  i=i-1
end
```

Exercice 2. Que se passerait-il si l'on demandait *(ne pas tester !)* :

```
i=5
Tant que i>0, faire
  i=i+1
Fin
```

Exercice 3.

1. Soit $s = 0.5$. Partir de $x = 100$ et remplacer x par sa racine carrée jusqu'à obtenir un nombre strictement plus petit que s. Que se passe-t-il ? pourquoi ? comment s'en sortir ? (ctrl + C)

2. Réessayer avec $s = 1.2$. Combien d'itérations ont été nécessaires pour atteindre le résultat ? (Faire calculer à MATLAB ce nombre d'itérations ; le script ne devra afficher que les différentes valeurs prises par x ainsi que le nombre final d'itérations.)

2.2 Structure conditionnelle : tests if

Utilité : faire ou ne pas faire d'instruction(s). Autrement dit : dans tel cas faire ceci, sinon faire cela.

Créer un script `exemples_if.m` pour y tester les exemples suivants.

Exemple 9. Tester ce script, pour $i = 2$, $i = 5$ et $i = 10$.

```
Si i=10, alors
  calculer 2×π
  écrire ''car i=10''
Fin
```

```
if (i==10)
  2*pi
  disp('car i=10')
end
```

Exemple 10. Tester ce script, pour différentes valeurs de x.

```
Si x modulo 2 vaut 0, alors
  écrire ''x est pair''
Fin
```

Exemple 11. Tester ce script, pour différentes valeurs de x.

```
Si x modulo 2 vaut 0, alors
  écrire ‘‘x est pair’’
sinon
  écrire ‘‘x est impair’’
Fin
```

```
if mod(x,2)==0
  disp('x est pair')
else
  disp('x est impair')
end
```

Que se passe-t-il si x n'est pas entier ? pourquoi ?

Exemple 12. Tester ce script, pour différentes valeurs de i.

```
Si i>0, alors
  écrire ‘‘i positif’’
sinon, si i<0 alors
  écrire ‘‘i négatif’’
sinon
  écrire ‘‘i nul’’
Fin
```

```
if i>0
  disp('i positif')
elseif i<0
  disp('i négatif')
else
  disp('i nul')
end
```

Exercice 4. Étant donné un réel x, écrire un script renvoyant $|x|$ (sans utiliser **abs**).

Exercice 5. Étant donné un entier n, renvoyer "trop grand" si $n \geq 10$ et sinon : renvoyer $n^{n!}$ si n est pair et $n!^n$ si n est impair. Que renvoie-t-il pour $n = 2$? $n = 7$? $n = 6$?

3 Application à des algorithmes connus

Exercice 6 (Systèmes triangulaires)**.**

1. Programmer les deux algorithmes de résolution d'un système triangulaire. *Rappel :* il s'agit de résoudre $Ax = b$, où $A \in \mathrm{GL}_n(\mathbb{C})$ est triangulaire supérieure (ou inférieure) et $b \in \mathbb{C}^n$.

Algorithme de descente

```
n=taille de A
Initialiser x
x_1=b_1/a_11
Pour i allant de 2 à n
  x_i=b_i
  Pour k=1 à i-1
    x_i=x_i − a_ik x_k
  Fin
  x_i=x_i/a_ii
Fin
Afficher x
```

Algorithme de remontée

```
n=taille de A
Initialiser x
x_n = b_n/a_nn
Pour i allant de n-1 à 1
  x_i = b_i
  Pour k=i+1 à n
    x_i = x_i − a_ik x_k
  Fin
  x_i = x_i/a_ii
Fin
Afficher x
```

2. Le(s)quel(s) de ces algorithmes permet(tent) de résoudre le système linéaire suivant, vu en cours ?

$$\begin{cases} 3x_1 + 2x_2 + x_3 = 1 \\ \frac{7}{3}x_2 + \frac{5}{3}x_3 = \frac{5}{3} \\ \frac{24}{7}x_3 = \frac{3}{7} \end{cases} \tag{1}$$

3. En interprétant le système (1) sous forme $Ax = b$, demander à MATLAB de le résoudre, avec l'algorithme choisi à la question 2 : quelle solution trouve-t-il ?

4. Que retourne la commande `A\b` ? En déduire à quoi sert la commande `\` dans la syntaxe `M\v` où M est une matrice et v un vecteur.

```
A\b
```

5. Expliciter pas à pas ("à la main") toutes les étapes effectuées par l'algorithme pour résoudre (1), *i.e.* se prendre pour l'ordinateur et faire les calculs de l'algorithme. Quelle est la solution de $Ax = b$?

6. Compléter le premier algorithme de la question 1 pour rajouter le test suivant : si un des a_{ii} est nul, alors écrire "non inversible" ; sinon, faire l'algorithme de descente.

Exercice 7 (Nombres premiers)**.** Donner la liste des nombres premiers entre 1 et N ; prendre pour exemple d'application $N = 100$.

Méthode pour lister : commencer par créer un vecteur P vide ; ajouter ensuite les éléments désirés au vecteur P. On pourra s'inspirer des commandes suivantes :

```
P=[]
P=[P,2]
```

```
isprime(37)
```

N'oubliez pas de rendre le TP,
avec vos nom et prénom.