Initiation à Matlab

Généraltés

- Matlab est un logiciel de calcul scientifique :
  - fonctions mathématiques usuelles
  - calcul matriciel
  - racines d'un polynôme
  - équations différentielles
  - intégration numérique
  - graphiques 2D & 3D
- **Matlab peut être utilisé des deux manières suivantes:**
  - en entrant des instructions à la suite du prompteur >>
  - en créant des **scripts** (ou **m-files**) ou des fonctions (extension **.m)**

Aide

La commande help peut être utilisée en ligne à fin de fournir l’aide en ligne des problèmes ou questions syntaxiques.

Commandes numériques en ligne

- a=1
- 2+a
- b=3*ans
- c=b^4
- d=sqrt(c)
- d+c*a^b/d

Matrices

Manipulations:

- A=[1 2;3 4] ou A=[1,2 3,4] et B=[1 1]
- C=[A;B]: concaténation
- C=0:10 un vecteur allant de 0 à 10
- C=-3:.3:3 un vecteur allant de -3 à 3 par pas de 3
- D=zeros(2) matrice de taille 2x2 contenant des 0
- D=ones(3,4) matrice de taille 3x4 contenant que des 1
- I=eye(3) matrice identité de taille 3x3

Matrices

Accès aux éléments de la matrice:

- Soit C=[ones(4,5);5 4 6 2 3] et b=[1 2 3]
- C(5,4) nous donne le chiffre positionné dan la 5 ème ligne 4 ème colonne.
- b(2)=4 affecte à la deuxième valeur de b une nouvelle valeur
- C(1,:) affiche la première ligne de la matrice C
- C(:,end) affiche la dernière colonne de C
- C(:) vecteur colonne composé de tous les éléments de C
- find(C==2) affiche les positions des éléments égaux à la valeur 2.

Fonctions portant sur les matrices

- length(A) retourne la plus grande des valeurs entre le nombre de lignes et de colonnes de A.
- size(A) retourne le nombre de lignes et de colonnes de la matrice A (L'expression [n,m]=size(A) affecte à n et m le nombre de lignes et de colonnes de A respectivement)
- rank(A) retourne le rang de la matrice A
- trace(A) calcule la valeur de la trace de A(somme des éléments diagonaux)
- norm(A) calcule la norme 2 de la matrice A

- det(A) calcule le déterminant de la matrice A
- diag(A) retourne les éléments de la diagonale de A
- inv(A) calcule l'inverse de A (s'il existe)
- Pour extraire la diagonale de A : diag(A)
- Pour construire la matrice diagonale de A : diag(diag(A))

Opérations sur les matrices

- **Soit A=[1 2 3;5 6 4;2 1 2] et B=[2 4 1; 5 7 1; 2 5 4]**
- A’ : Matrice conjuguée de A
- A^-1: La matrice inverse de A (si elle existe)
- A + B :Addition des matrices A et B
- A − B: Soustraction des matrice A et B
- A\B: Résout le système linéaire Ax=B, elle effectue donc l’opération: B*A^-1
- A/B: Effectue l’opération A^-1*B
- A.* B: Multiplication des éléments de A par les éléments de B un par un
- A. /B :Division des éléments de A par les éléments de B un par un
- A.^ x :Chaque élément de A sera élevé à la puissance x
- A + x: Additionne x à chacun des éléments de A
- A * x :Multiplie chacun des éléments de A par x

Fonctions portant sur les matrices

- sum(A) fait la somme des éléments de chaque colonne de A
- prod(A) fait le produit des éléments de chaque colonne de A
- max(A) renvoie les valeurs maximales de chaque colonne
- min(A) renvoie les valeurs minimales de chaque colonne
- mean(A) renvoie la moyenne de chaque colonne
- sort(C) fait le tri d'une colonne

Polynôme sous Matlab

Pour évaluer la valeur d'un polynôme en plusieurs points, on utilise les commandes suivantes :

- polyval(p,a) : évalue la valeur du polynôme p en chaque coefficient de a.
- polyvalm(p,A) : calcule pour la matrice carrée A la valeur du polynôme matricielle.

Polynôme sous Matlab

Exemples:

```
>> p=[-1 2 -4 3]
p =
    -1     2    -4
3
>> polyval(p,0)
ans =
         3
>> polyval(p,1)
ans =
         0
```

```
>> A=[1 2 3;-1 0 5;2 -2 4]
A =
     1     2     3
    -1     0     5
     2    -2     4
 >> polyvalm(p,A)
ans =
   -50    24   -57
   -33    -5   -21
   -24     0   -53
```

Opérations sur les polynômes

Etant donné deux polynômes p et q, on a :

- conv(p,q) : produit des polynômes p et q.
- deconv(p,q) : le polynôme obtenu de la division de p par q.
- [h,r]=deconv(p,q) : h est le polynôme obtenu de la division de p par q et r est le reste de la division.
- polyder(p) : calcule les coefficients du polynôme dérivé p’ de p

Polynôme sous Matlab

Exemples:

```
>> p=[-1 2 -4 3]
p =
    -1     2    -4     3

>> q=[1 0 2]
q =
     1     0     2

>> conv(p,q)
ans =
    -1     2    -6     7    -8
6
```

```
>> deconv(p,q)
ans =
    -1     2
>> [h,r]=deconv(p,q)
h =
    -1     2
r =
     0     0    -2    -1
 >> polyder(p)
ans =
    -3     4    -4
```

Programmation sous Matlab

- ***Comme pour la plus part des logiciels de programmation, Matlab utilise une syntaxe dont nous allons donner les éléments les plus utilisés.***

Opérateurs logiques

- **Nous donnons ci-dessous les principaux opérateurs logiques utiles pour la programmation.**
- **- == : égal à,**
- **- ~= : différent de,**
- **- < : strictement inférieur à,**
- **- <= : inférieur ou égal à,**
- **- > : strictement supérieur à,**
- **- >= : supérieur ou égal à,**
- **- & : et logique,**
- **- | : ou logique,**
- **- ~ : non logique,**
- **- xor : ou exclusif logique.**

Syntaxe de branchement

- **Instruction conditionnelle if : permet d'exécuter un groupe d'instructions si la condition après l'instruction if est vraie. La syntaxe est la suivante :**

```
If (condition logique1)
 Instructions 1
elseif (condition logique 2)
 Instructions 2
…
else
 Instructions p
end
```

On peut omettre les parenthèses autour des conditions logiques.

Syntaxe de branchement

- **Exemples:** Etant donnée l'équation ax2 + bx + c = 0, le nombre de racines réelles dépend du signe de delta.

```
>> a=1;b=1;c=-2;
>> delta=b^2-4*a*c;
>> if (delta < 0)
        ('pas de solutions
réelles')
    elseif (delta==0)
        ('une solution réelle
double')
    else
        ('deux solutions
réelles distinctes')
    end
```

```
ans =
        deux solutions
distinctes
>> a=1;b=1;c=2;
>> delta=b^2-4*a*c;
>> ans =
            pas de
solutions réelles
```

Boucles

- **La boucle for :** permet de répéter un certain nombre de fois fixé à l'avance une instruction structurée. Sa syntaxe est la suivante :

```
for variable = valeur de début : pas : valeur de fin
                instructions ;
end
```

Boucles

- **Exemples:** Calculer la tangente des entiers entre 1 et 5.

```
>> for i=1:5
       x(i)=tan(i)
           end
      x =
1.5574   -2.1850   -0.1425    1.1578   -3.3805
```

Boucles

- **La boucle répétitive while :** est utilisée lorsqu'on répète une instruction structurée sans savoir à l 'avance quand on doit s'arrêter . La syntaxe est la suivante :

```
while (condition logique)
  instructions ;
end
```

l'instruction break : permet d'arrêter l'exécution à l'intérieur des boucles for et while.

Boucles

- **Exemples:** Etant donné les trente premiers entiers placés d'une manière aléatoire dans un vecteur v, on chercher la position du nombre 10 dans ce vecteur.

```
>> x=randperm(30)   % génère une permutation aléatoire des 30 premiers
entiers
      x =    5    30    24    14    18     3    26    28     1    22    11
20     2    25     6
             4    17    12    29     7    21    10     9    13    27    19
16    15    23     8
>> i=length(x);     while x(i)~=10          % i est  la taille du
vecteur x
          i=i-1;
          end;   i
```

Exercice1:

- Créer les matrices (il y a plusieurs façons) :

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix} \text{ et } \begin{bmatrix} 0 & 4 & 4 & 4 \\ 4 & 0 & 4 & 4 \\ 4 & 4 & 0 & 4 \\ 4 & 4 & 4 & 0 \end{bmatrix}.$$

- Extraire la deuxième colonne de la première matrice, sa deuxième ligne et sa sous-matrice 2 x 2 centrale.
- Inverser les 2 colonnes centrales de la première matrice et les deux premières lignes de la seconde matrice.

Exercice2:

- 5) Ecrire une fonction Matlab prenant 2 vecteurs ligne U et V en entrée et produisant en sortie le produit scalaire.

Mode Programmation

En mode programmation, toute instruction non terminée par un `;` génère l'affichage d'un résultat lors de l'exécution du programme dans la fenêtre d'exécution. L'affichage d'un résultat ne doit se faire que pour les résultats finaux ou pour détecter des erreurs. Ainsi, il faut prendre garde de taper le `;` à la fin de chaque instruction.

Scripts et fonctions

function [$vars_1$, ..., $vars_m$**] =** nom_fonction **(**$vare\_1$, ..., $vare_n$**)**

séquence d'instructions.

end

Enregistrement du fichier contenant la fonction par le nom_fonction.m

Réferences

- http://nte.mines-albi.fr/MATLAB/co/uc_Generalites.html
- Introduction to MATLAB ES 156 Signals and Systems 2007 Harvard SEAS
- INITIATION A MATLAB FILIERE SMA QUATRIEME SEMESTRE ANNEE UNIVERSITAIRE 2009-2010, A. MAZROUI