Communications numériques

Formulaire pour l'utilisation de MATLAB

Université Paris 13, Institut Galilée, Ecole d'ingénieurs Sup Galilée
Parcours Informatique et Réseaux Apprentissage - 2ème année

2018-2019

Scripts et fonctions

- Un script est un fichier `.m` pouvant être exécuté. On écrira en haut de chaque script le nom et le prénom de l'auteur, la date, ainsi que les commandes suivantes qui permettent de nettoyer tout l'espace de travail à chaque fois que le script est lancé.

```
% NOM Prenom
% Date
clear all % Supprime toutes les variables de l'espace de travail
close all % Ferme toutes les figures courantes
clc    % Nettoie l'historique des commandes
```

- Une fonction est un fichier `.m` ne pouvant pas être exécuté tel quel. Il faut créer en parallèle un script où l'on définit les entrées, où l'on appelle la fonction et où l'on récupère les sorties de la fonction. Le nom de la fonction doit être exactement le nom du fichier `.m` associé. L'entête du fichier doit être le suivant :

```
function [out1,out2] = nomFonction(in1,in2,in3)
% nomFonction : nom de la fonction : le script doit s'appeler nomFonction.m
% in1, in2, in3 : arguments d'entree de la fonction
% out1, out2 : arguments de sortie de la fonction
```

Vecteurs

Création

- Création manuelle d'un vecteur

```
x=[];          % vecteur vide
x=[3;7;2;1;5]; % vecteur colonne
x=[3 7 2 1 5]; % vecteur ligne
```

- Vecteur de taille `N` composé uniquement de 0

```
x=zeros(N,1); % vecteur colonne
x=zeros(1,N); % vecteur ligne
```

- Vecteur de taille `N` composé uniquement de 1

```
x=ones(N,1); % vecteur colonne
x=ones(1,N); % vecteur ligne
```

- Vecteur contenant toutes les valeurs entre `x_min` et `x_max` avec un pas de `x_step`

```
x=x_min:x_step:x_max % vecteur ligne
ex : t=0:0.1:0.5   -> t=[0 0.1 0.2 0.3 0.4 0.5]
```

Manipulation

- Connaître la taille `N` d'un vecteur `x`

```
N=length(x);
```

- Concaténer deux vecteurs `x` et `y`

```
z=[x; y]; % x et y vecteurs colonne
z=[x y]; % x et y vecteurs ligne
```

- Transposer un vecteur

```
x=x.'; % Transposition (sans prendre le conjugué)
x=x';  % Tranposition (en prenant le conjugué)
```

- Accéder à l'élément `i` du vecteur `x` (attention, `i` commence à 1 pour le premier élément)

```
x(i);
```

- Accéder à tous les éléments de `x` dont l'indice est compris entre `n1` et `n2` (inclus)

```
x(n1:n2); % renvoie un vecteur de taille n2-n1+1
```

- Accéder à tous les éléments de `x` dont l'indice est écrit dans le vecteur `ind`

```
x(ind); % renvoie un vecteur de taille length(ind)
ex : x=[2 7 9 8 0];  ind=[3 5 1];    -> x(ind)=[9 0 2];
```

Opérations usuelles

- Réaliser une opération de base entre un vecteur `x` et un scalaire `a`

```
x=x+a; % Addition
x=x-a; % Soustraction
x=x*a; % Multiplication
x=x/a; % Division
```

- Réaliser une opération de base terme à terme entre deux vecteurs `x` et `y` de même taille

```
z=x+y;  % Addition
z=x-y;  % Soustraction
z=x.*y; % Multiplication
z=x./y; % Division
```

- Mettre tous les éléments d'un vecteur `x` à la puissance `p`

```
x=x.^p; % Puissance p quelconque
```

- Prendre la valeur absolue de tous les éléments d'un vecteur `x`

```
x=abs(x); % Valeur absolue
```

- Arrondir tous les éléments d'un vecteur `x`

```
x=round(x); % Arrondit a l'entier le plus proche
x=floor(x); % Arrondit a l'entier inferieur (partie entiere)
x=ceil(x);  % Arrondit a l'entier superieur
```

- Réaliser une opération de base sur tous les éléments d'un vecteur `x` pour obtenir un scalaire `a`

```
a=sum(x);  % Somme
a=prod(x); % Produit
a=mean(x); % Moyenne
a=var(x);  % Variance
```

Recherche d'éléments

- Rechercher la valeur minimum/maximum dans un vecteur `x`, et récupérer l'indice où cette valeur se situe dans le vecteur

```
[x_min,i_min]=min(x); % x_min : valeur du minimum, i_min : indice où la valeur se situe
[x_max,i_max]=max(x); % x_max : valeur du maximum, i_max : indice où la valeur se situe
```

- Rechercher les indices `ind` pour lesquels les éléments d'un vecteur `x` valent `a`

```
ind=find(x==a); % Indices pour lequels la condition est vraie
% Renvoie un vecteur vide si la condition fausse partout
ind=find(abs(x-a)<10^(-10)); % Plus prudent si la valeur a est réelle
```

- Rechercher tous les éléments d'un vecteur `x` vérifiant une certaine condition, et modifier leurs valeurs

```
x(x<0)=0;       % Met a 0 toutes les valeurs strictement negatives de x
x(y<3 & y>0)=1; % Met a 1 toutes les valeurs du vecteur x
% correspondant aux indices où y est compris entre 0 et 3
```

Boucles et conditions

- Instruction `for`

```
for i=1:N
  x(i)=i;
end
```

- Instruction `while`

```
while (i<=N)
  x(i)=i;
end
```

- Instruction `if`

```
if (i==3)
  i=i+1;
end
```

- Instruction `switch`

```
switch k
    case 1
     x=1;
    case 2
     x=2;
    otherwise
     x=0;
end
```

Figures

- Tracer un vecteur `x` en fonction d'un vecteur `t`

```
figure     % Ouvre une nouvelle figure
plot(t,x)  % Affiche le vecteur x en fonction du vecteur t
title('Ma Figure') % Donne un titre a la figure
xlabel('NomX') % Donne un nom a l'axe des abscisses
ylabel('NomY') % Donne un nom a l'axe des ordonnees
xlim([0 1])  % Restreint la figure pour les abscisses comprises entre 0 et 1
ylim([-1 1]) % Restreint la figure pour les ordonnees comprises entre -1 et 1
```

- Tracer plusieurs courbes sur le même graphique

```
figure     % Ouvre une nouvelle figure
plot(t,x)  % Affiche le vecteur x en fonction du vecteur t
hold on    % Indique que la prochaine instruction sera sur le meme graphique
plot(t,y,'-r') % Affiche le vecteur y en fonction du vecteur t (en rouge)
```