Traitement des Signaux Aléatoires

Université Paris Diderot
Physique PMA M1 et M1 EIDD
2011-2012

TP 1
Introduction

L'objectif de cette séance est de prendre en main le logiciel MATLAB. Les notions traitées sont la manipulation de variables et de vecteurs, l'utilisation de fonctions, la représentation graphique, la génération de nombres aléatoires, le calcul d'écart-type, l'écriture et la lecture de fichiers et éventuellement la définition de fonction.

1 Manipulation des variables et utilisation de fonctions

Déclaration et initialisation :
- `a = 0.` **: nombre réel,**
- `a = [1., 2., 3., 4., 5.]` **: série (ou tableau une dimension) avec des valeurs initiales quelconques,**
- `a = 0:0.1.:5.` **: série avec des valeurs initiales régulières (valeur de 0 à 5 avec un pas de 0.01),**
- `a = [1., 2., 3., 4., 5.;11.,16.,8.,5.,23.]` **: tableau (2 dimensions) avec des valeurs initiales quelconques,**
- `a=[0.:0.1:5.;0.:0.2:10.]` **: tableau avec des valeurs initiales régulières,**

Accès (lecture/écriture) à un tableau :
- `a(l,c)` **: accès à l'élément de la ligne** `l` **et de la colonne** `c` **du tableau 2D** `a`
- `a(l,:)` **: accès à tous les éléments de la ligne** `l` **du tableau 2D** `a`,

Pour appliquer une fonction a chacune des valeurs d'une série, il suffit d'utiliser une syntaxe du type : `result = sin(a)`
Note : Pour éviter l'affichage du résultat, il faut terminer la ligne par un point virgule.

Pour chacune des questions, écrivez les commandes MATLAB correspondantes sur votre compte-rendu.

1. Inversez les valeurs de deux variables A et B ?

Correction:
```
A = 13.
B = 25.
C = A
A = B
B = C
A,B
```

2. Créez une série contenant des valeurs temporelles sur une durée de 10 ms à une fréquence d'échantillonnage de 10 kHz.

Correction:
```
t = 0.:1./10000.:10.e-3
```

3. Créez une série contenant l'amplitude du signal de la note 'la' (fréquence de 440 Hz) correspondant aux valeurs temporelles de la question 2.

Correction:
```
signal = sin(2.*pi*440*t)
```

4. Créez un tableau dont la première ligne contient les valeurs temporelles de la question 2 et dans la deuxième celles du signal demandé à la question 3.

Correction:
```
sig = [0.:1./10000.:0.01;0.:1./10000.:0.01]
sig(2,:) = sin(2.*pi*440*sig(1,:))
```

2 Représentation des données

Représentation graphique de la liste `x` en fonction de la liste `y` :
`plot(x,y,`*style*`,` *option1*`,` *valeur de l'option1*`,` *option2*`,` *valeur de l'option2*`, ...)`
style **correspondant au style de ligne (`'-'`, `'--'`, `'-.'` ou `':'`) et/ou de marqueur pour les points (`.`, `'+'`,`'s'`, `'o'`, etc). Exemple : `'-+'` correspond à une ligne en tiret avec des points en forme de plus (voir `doc LineSpec` pour plus de détails). Quelques options utiles sont :**

- **`'Color'` : couleur de ligne. (note : la couleur peut également spécifié dans le style). Exemples : `'red'`, `blue`, ...**
- **`'LineWidth'` : epaisseur de la ligne (en point)**
- **`Marker` : forme du marqueur : `'o'`, `'.'`, `'s'` `'^'`**
- **`'MarkerSize'` : taille des points (en point)**
- **`'MarkerEdgeColor'` : couleur des points**

Le titre des axes est spécifié par les commandes `xlabel('`*titre de l'axe des y'*`)` et `xlabel('`*titre de l'axe des y'*`)`
La légende est spécifié par al commandes `legend('`*légende courbe 1'*`,'`*légende courbe 2'*`,...)`
Pour superposer plusieurs courbes sur un même graphique il faut utiliser la commande `hold on` : tant que `hold` est `on` tous les résultats des commandes `plot ...` seront affichés sur le même graphique. Pour recommencer un nouveau graphique il faut taper la commande `hold off`.
Les valeurs minimales et maximales sur les axes sont spécifiés par : `axis([xmin xmax ymin ymax])`
(Note : Pour la manipulation des graphiques en mode interactif, voir l'aide "Plotting Tools Interactive Plotting")

1. Représentez l'amplitude du signal s_1 correspondant note la (fréquence = 440 Hz) sur 10 ms en spécifiant le temps sur l'axe x et l'amplitude sur l'axe y.
Correction:

```
t=0.:1/10000.:0.01;
sig=sin(2*pi*440*t);
plot(t,sig,'-r')
xlabel('Temps (s)')
ylabel('Amplitude')
```

2. Représentez sur un même graphique l'amplitude des signaux s_1 et s_2 correspondant aux notes la et mi (fréquence = 329.63 Hz) en spécifiant les axes x et y ainsi que la légende. La courbe correspondant à la note la sera représentée en tiret rouge d'épaisseur 2 points et celle de la note mi en trait continu vert d'épaisseur 1 point en faisant apparaître les points d'échantillonnage par des cercles noirs de taille 10 points.
Correction:

```
t=0.:1/10000.:0.01;
sig=[sin(2*pi*440*t);sin(2*pi*329.63*t)];
plot(t,sig(1,:),'-','Color','red','LineWidth',2 )
hold on
plot(t,sig(2,:),'-o','Color','green','LineWidth',1,'MarkerSize', 10,'MarkerEdgeColor','black')
xlabel('Temps (s)')
ylabel('Amplitude')
Legend('la','mi')
hold off
```

3. Copier le fichier `tpTSA_son.m` dans votre répertoire de travail. Il contient la fonction `tpTSA_son(t,s)` joue le signal `s` en fonction du temps `t` puis affiche les 100 premières millisecondes. Générez des signaux similaires à ceux de la question 2 sur une durée de 3 secondes et échantillonnée à 10 kHz. Jouez-les avec la fonction `tpTSA_son`.

---

3 Générateur de nombre aléatoire

La commande `a = rand(m,n)` permet de créer une tableau contenant `m` × `n` valeurs aléatoires entre 0 et 1.
L'initialisation du générateur de nombre aléatoire se fait par la commande `rand('twister',12345)` où 12345 est la racine du générateur.
Une représentation usuelle de données pseudo-aléatoire est l'histogramme qui s'effectue par la commande `hist(d,nbins)` où `d` représente les données et `nbins` le nombre d'intervalle (voir `doc hist` pour plus d'informations).

1. Générer une série temporelle de points aléatoires n_1 d'une durée de 3 s et échantillonnée à 10 kHz. On utilisera 112233 comme racine du générateur aléatoire. Représentez n_1 graphiquement et vérifier approximativement son uniformité.

Correction:

```
t=0.:1/10000.:3.;
rand('twister',112233)
n1=rand(1,size(t,2));
plot(t,n1,'.','Color','red')
xlabel('Temps (s)')
ylabel('valeurs n1')
pause
hist(n1,30)
xlabel('Intervalle')
ylabel('Nombre de valeurs dans l intervalle')
```

Histogramme manuel :

```
xbins = 0.:0.05:0.95;
vbins = zeros(size(xbins));
for i=1:size(xbins,2),
   vbins(1,i)=length(find(xbins(1,i)<n1 & n1<xbins(1,i)+0.05));
end;
plot(xbins,vbins,'.-','Color','red')
axis([0. 1. 0. 2000.])
xlabel('Intervalle')
ylabel('Nombre de valeurs dans l intervalle')
```

2. Générer une deuxième série temporelle de points aléatoires n_2 d'une durée de 3 s et échantillonnée à 10 kHz. Vérifier graphiquement qu'il n'y a approximativement pas de corrélation entre n_1 et n_2.

Correction:

```
n2=rand(size(t));
plot(n1,n2,'.','Color','red')
xlabel('valeurs n1')
ylabel('valeurs n2')
```

3. Générer une série temporelle n_3 tel que $n_3 = 2n_1$. En utilisant une représentation graphique, déterminez si n_3 set uniforme. Déterminer graphiquement si n_3 et n_1 sont corrélés.

Correction:

```
n3=2*n1;
plot(t,n3,'.','Color','red')
xlabel('Temps (s)')
ylabel('valeurs 3')
pause
plot(n1,n2,'.','Color','red')
xlabel('valeurs n1')
ylabel('valeurs n3')
```

4. Générer une nouvelle série temporelle de points aléatoires n_4 identique à n_1 SANS copier n_1.

Correction:

```
rand('twister',112233)
n4=rand(size(t));
plot(t,n1,'o','Color','blue')
xlabel('Temps (s)')
ylabel('valeurs')
hold on
plot(t,n4,'.','Color','red')
Legend('n1','n4')
hold off
```

5. Ajouter un bruit sur le signal de la note la.

Correction:

```
t=0.:1/10000.:3.;
s=sin(2*pi*440*t);
n=rand(size(t));
sn=s+n;
tpTSA_son(t,sn)
```

---

4 Calcul d'écart-type

La somme des valeurs d'une série de données se fait par la commande `sum(d,dim)` où `d` correspond au données et `dim` à l'indice de la dimension considérée.

1. Générer une série n_5 de 10000 valeurs aléatoires. Calculer la moyenne et l'écart-type en utilisant la fonction `sum`. Comparer avec les résultats des fonctions `mean(X)`, `std(X,0)` et `std(X,1)` (`X` représente la série de données).

Correction:

```
n5 = rand(1,10000);
meann5 = sum(n5,2)/size(n5,2)
stdn5 = sqrt(sum((n5-meann5).^2,2)/size(n5,2))
mean(n5)
std(n5,0)
std(n5,1)
```

5 Lecture/écriture de fichier

Pour lire des données dans un un fichier, on utilise les commandes suivantes :
- `fin = fopen('NomFichier.txt','r')` **: ouvre le fichier** `NomFichier.txt` **en mode lecture.**
- `d=fscanf(fin,'%f %f',[2,inf])` **: Lit les données :** `'%f %f'` **décrit le format d'une ligne (deux réels séparés par un espace) et** `[2,inf]` **décrit le format des données de sortie (2 séries de données jusqu'à l'infini c'est-à-dire jusqu'à la fin du fichier).**
- `fclose(fin)` **: ferme le fichier**

Pour écrire des données dans un fichier, on utilise les commandes suivantes :
- `fout = fopen('NomFichier.txt','r')` **: ouvre le fichier** `NomFichier.txt` **en mode écriture.**
- `fprintf(fout,'%f %f\n',d)` **: écrit les données** `d` **avec pour chaque ligne le format** `'%f %f\n'` **(deux réels séparés par un espace puis un retour à la ligne** `\n`**).**
- `fclose(fout)` **: ferme le fichier**

1. Copier le fichier **tpTSA1_LaE_5s.txt** dans votre répertoire de travail. Sachant que la première colonne correspond au temps et la deuxième au signal, représentez les 10 premières millisecondes. Jouez le signal avec la fonction `tpTSA_son`.

Correction:

```
fIn = fopen('tpTSA1_LaE_5s.txt','r');
d=fscanf(fIn,'%f %f',[2,inf]);
fclose(fIn);
plot(d(1,:),d(2,:),'-','Color','red')
xlabel('Temps (s)')
ylabel('Amplitude')
pause
plot(d(1,1:440),d(2,1:440),'-','Color','red')
xlabel('Temps (s)')
ylabel('Amplitude')
pause
tpTSA_son(d(1,:),d(2,:))
```

2. Créez un nouveau fichier **tpTSA1_LaE_new.txt** contenant le signal du fichier **tpTSA1_LaE_5s.txt** échantillonné à une fréquence de 35 kHz (échantillonnage original = 44 kHz) et d'amplitude deux fois plus grande.

Correction:

```
fIn = fopen('tpTSA1_LaE_5s.txt','r');
d=fscanf(fIn,'%f %f',[2,inf]);
fclose(fIn);
d2=[d(1,:)*44./35.;d(2,:)*2.];
fOut=fopen('tpTSA1_LaE_new.txt','w')
fprintf(fOut,'%f %f\n',d)
fclose(fOut)
tpTSA_son(d2(1,:),d2(2,:))
```

3. Lire les données du fichier **tpTSA1_data.txt Voir comment l'on gère des fichiers de dimensions variables ...**

6 Définition de fonction

En MATLAB, il est possible de créer ses propres fonctions. Par exemple si l'on veut créer la fonction `[res1,res2]=MaFonction()` qui prend comme deux arguments `arg1` et `arg2` et qui retourne deux résultats `res1` et `res2`, il faut créer un fichier nommé `MaFonction.m` contenant la description de la fonction au format suivant :

```
function [res1,res2] = MaFonction(arg1, arg2)
% description de ma fonction
res1=arg1+arg2;
res2=arg1-arg2;
```

1. Créer une fonction qui renvoie la moyenne ET l'écart-type d'un série de données :

Correction:

Le fichier `MoyenneEcartType.m` contient :

```
function [rmoy,rsig] = MoyenneEcartType(X)
% Renvoie la moyenne et l'?cart-type de la serie de donnees X
rmoy = mean(X);
rsig = std(X,1);
```