TP No: 2 corrélation et convolution sous Matlab

1. Signaux périodiques

Exécuter le programme suivant :

```
Ts=.001; % pas déchantillonnage
t=-15:Ts:15; % support du signal
Fs=1/Ts; % fréquence d'échantillonnage
f0=2; y=sin(2*pi*f0*t+0); % voir l'influence de la
phase
cy=xcorr(y)/length(t); % autocorrélation
coy=conv(y,y)/length(t); % convolution du signal
avec lui même
tc=2*min(t):Ts:2*max(t); % durée du signal auto
corrélation
subplot(311); plot(t,y);grid; title('signal');
subplot(312); plot(tc,cy); grid; title('corrélation
Cyy');
subplot(313); plot(tc,coy); grid; title('convolution
y(t)*y(t)');
f=linspace(-Fs/2,Fs/2,length(t));
Yf=fftshift(fft(y)/length(t)); % transformée de
Fourier bilatérale
figure(2);
stem(f,abs(Yf));grid;
axis([-10,10,0,1]); title('spectre amplitude du
signal y(t)');
```

Changer la phase du signal de 0, $\pi/2$, $\pi/3$ et π :

- Montrer la différence entre la corrélation et la convolution par l'influence du changement de la phase et quand la corrélation sera égale ou opposée à la convolution ?
- Quelle est donc la période des signaux de corrélation et de convolution par rapport au signal original

2. Signal à durée limitée

Exécuter le programme suivant :

```
clc; clear all;
Ts=.001; Fs=1/Ts;% pas et fréquence échantillonnage
t=-10:Ts:10;      % t durée du signal
taux=5;          % taux position centre du signal
T=4;             % T durée 1/2 de la fenetre triangulaire
x=tripuls(t,2*T); % x signal choisi
y=tripuls(t-taux,2*T); % y signal choisi décalé
cxy=xcorr(x,y)/length(t); % signal corrélation cxy
cyx=xcorr(y,x)/length(t); % signal corrélation cyx
coxy=conv(x,y)/length(t); % convolution coxy
coyx=conv(y,x)/length(t); % convolution coyx
tc=2*min(t):Ts:2*max(t) ; % durée du signal
autocorrélation
subplot(411); plot(t,x,'b',t,y,'r');
grid;title('signal x en bleu et y en rouge'); %
signal
subplot(412); plot(tc,cxy,'b',tc,cyx,'r');grid;
title('intercorrelation cxy en bleu et cyx en
rouge');% autocorrélation
subplot(413);plot(tc,coxy,'m',tc,coyx,'r');grid;title
('convolution entre x et y'); % convolution
subplot(414);plot(tc,coyx,'r');grid;title('convolutio
n entre y et x'); % convolution
f=linspace(-Fs/2,Fs/2,length(t));
Yf=fftshift(fft(y)/length(t)); % transformée de
Fourier bilatérale
figure(2);
plot(f,abs(Yf));grid;
axis([-2,2,0,.3]); title('spectre amplitude du signal
y(t)');
```

- Changer taux pour différentes valeurs : que constatez-vous pour $\text{taux} = 0$?
- Vérifier le retard des signaux par l'inter corrélation
- Changer la durée T du signal et montrer son influence
- Changer le signal x et y par les instructions suivantes et interpréter les résultats:

```
x=rectpuls(t,T);x=x.*1; % x signal choisi
y=rectpuls(t-taux,T);y=y.*1; % y signal choisi
décalé
```

Conclusion

TP No1 : génération des signaux sous Matlab

Introduction :

MATLAB fournit un ensemble de fonctions pour générer des signaux de différentes formes et propriétés. Ces fonctions peuvent être utilisées pour créer des signaux simples, tels que des sinusoides, des carrés et des impulsions, ou des signaux plus complexes, tels que des signaux aléatoires, des signaux modulés et des signaux non stationnaires.

Fonctions pour générer des signaux périodiques : Les fonctions pour générer des signaux périodiques sont les plus courantes. Elles permettent de créer des signaux qui se répètent à l'infini.

$\sin()$, $\cos()$ et d'autres fonctions génèrent des sinusoides, des cosinusoides et etc... Ces fonctions prennent en entrée la fréquence du signal souhaitée.

Exemples :

- Signal sinusoidal de fréquence 10 Hz et d'amplitude 1 dans l'intervalle (0,1)

```
t = 0:0.001:1; y1 = sin(2*pi*10*t); y2 = cos(2*pi*10*t); plot(t, y1, 'b', t, y2, 'r');
```

- Signal carré : `t = 0:0.001:1; y = square(2*pi*10*t); plot(t, y, 'r');`
- Signal en dents de scies :

```
t = 0 : 1/1e3 : 1; % 1 kHz sample freq for 1 sec
d = 0 : 1/3 : 1; % 3 Hz repetition freq
y = pulstran(t,d,'tripuls',0.1,-1); plot(t,y);
```

Fonctions pour générer des signaux à durée limitée : ce sont des signaux qui sont nuls en dehors d'un certain intervalle

Exemples :

- Une impulsion triangulaire de largeur 0.4

```
fs = 10000; t = -1:1/fs:1; w = .4; x = tripuls(t,w); figure, plot(t,x)
xlabel('Time (sec)'); ylabel('Amplitude'); title('Triangular Aperiodic Pulse')
```

- Une impulsion rectangulaire de largeur 0.2

```
t = -1:0.01:1; y = rectpuls(t, 0.2); plot(t, y);
```

- signal signe : `t=-5:.001:5; y=sign(t); plot(t,y,'r');`
- signal échelon : `t=-5:.001:5; y=heaviside(t-.5); plot(t,y,'r'); grid;`

Fonctions pour générer des signaux aléatoires

- Signal aléatoire uniforme : `t = 0:0.01:1; y = rand(1,length(t)); plot(t, y,'b');`
- Signal aléatoire gaussien de moyenne unitaire et de variance σ^2 `N(mu, sigma2)`

```
mu=1, sigma2=2 ;
t = 0:0.01:1; y = mu+sqrt(sigma2)*randn(1,length(t)); plot(t, y,'b');
```

Questions :

- Générer un signal sinusoidal d'amplitude $a=1$, de fréquence $f=10\text{Hz}$ dans l'intervalle $(-1,1)$ et limité par une fenêtre rectangulaire de durée unitaire.
- Générer un signal aléatoire gaussien de moyenne unitaire et variance 4 et tracer son graphe

Conclusion

TP No2 : analyse spectrale de Fourier**Analyse de Fourier****1) Signaux à énergie finie**

Soit le programme matlab suivant :

```
Fe=8e3; Te=1/Fe;
N=1024; % changer la valeur de N 64 128 512 1024 2048
duree=N*Te;
t = (0:N-1)/Fe;
T=max(t)/20; % changer la durée et interpréter les résultats
% choisir le signal tripuls ou rectpuls et voir son spectre d'amplitude et
% de phase
x=10*rectpuls(t,T);
%x=tripuls(t,T);
X = (1/N)*fft(x); f = (0:N-1)/N*Fe;
xmax=max(x); xmin=min(x);
subplot(311); plot(t,x);axis([0 max(t) xmin-1 xmax+1]);grid;
xlabel('temps t'), ylabel('x(t)');
subplot(312); plot(f(1:N/2),abs(X(1:N/2)));
xlabel('fréquence f'), ylabel('X(f)');grid;
subplot(313);plot(f(1:N/2),angle(X(1:N/2)));grid;
% la valeur moyenne du signal
X1bar=mean(x); x2bar=X(1)
```

- Changer la valeur de N et interpréter les spectres des signaux **rectpuls** et **tripuls**
- Comparer pour chaque cas x1bar et x2bar : expliquer d'où viennent ces résultats

2) Analyse de signaux périodiques cas des signaux à PMF non nulle

Taper le programme suivant :

```
% Génération du signal
Fe = 8e3;Te=1/Fe;
N = 1024; %%changer la valeur de N pour 128 ; 256; 512; 1024; 2048
duree=N*Te;
t = (0:N-1)/Fe; %x = square(2*pi*Fe*t/50);
% x=sin(2*pi*Fe*t/50);
% x=sin(2*pi*Fe*t/50)+sin(2*pi*Fe*t/100)+cos(2*pi*Fe*t/25);
% TFD sur [0, Fe]
X = (1/N)*fft(x); f = (0:N-1)/N*Fe;
% Affichage
xmax=max(x); xmin=min(x);
subplot(1,2,1); plot(t,x);axis([0 max(t) xmin-1 xmax+1]);grid;
xlabel('temps t'), ylabel('x(t)');
subplot(1,2,2); plot(f(1:N/2),abs(X(1:N/2)));
xlabel('fréquence f'), ylabel('X(f)');;grid;
% la valeur moyenne du signal
X1bar=mean(x);
% la valeur moyenne se calcule en domaine fréquentielle à la fréquence
% nulle du spectre du signal
X2bar=X(1);
```

- Appliquer ce programme pour les différents signaux carré, sinus et somme de sinus pour différentes valeurs de N et interpréter.
- Comparer pour chaque cas X1bar et X2bar : expliquer d'où viennent ces résultats

CONCLUSION GENERALE

TP N04 : échantillonnage et reconstitution du signal

Taper le programme ci-dessous

```

F1 = 350; % fréquence du signal
Fs = 800; % fréquence d'échantillonnage
tmin = 0; tmax = 10/F1; % pour 10 périodes
pas_temp=(tmax-tmin)/10000; % pas temporel du signal
analogique simulé
t = tmin:pas_temp:tmax; % interval temporel du signal
analogique
A=2; % amplitude du signal
N=8; % nombre de bit de codage choisi
q=(2*A)/(2^N); % pas de quantification calculé à partir du
nombre de bit
Eq=-A+q/2:q:A-q/2; % plage de quantification
x = A*sin(2*pi*F1*t); % signal test
Ts = 1/Fs; % pas d'échantillonnage
ts = tmin:Ts:tmax; % instants d'échantillonnage
xlresampled = A*sin(2*pi*F1*ts); % signal échantillonné
samples = length(ts); % taille du signal échantillonné
temporel
xlq=q*round((xlresampled-q/2)/q)+q/2; % plage des états de
quantification
error_q=xlq-xlresampled; % erreur de quantification
subplot(311);grid; plot(t,x,'b');title('en bleu signal
analogique');
hold on;
stem(ts,xlresampled,'r');grid;title('en rouge signal
échantillonné');
subplot(312);stem(ts,xlq,'b');grid;title(' signal
quantifié');
subplot(313); stem(ts,error_q); grid; title('erreur de
quantification');
%-----calcul de la puissance de l'erreur de
quantification-----
Pe=(1/samples)*sum(error_q.^2);% Pe=(1/n)*somme_carrée de
l'erreur
Pxe=(1/samples)*sum(xlresampled.^2); % calcul de la puissance
du signal
SB=Pxe/Pe; % rapport signal sur bruit
SBdB=10*log10(SB) % rapport du signal sur bruit en décibel
Pel=q^2/12; % la puissance de l'erreur à partir du pas de
quantification
SB1=Pxe/Pel; % puissance du signal sur puissance du bruit à
partir de Pel
SB1dB=10*log10(SB1) ; % en décibels
% calcul de S/B à partir du nombre d'échantillons

```

```

% la formule est en decibel  SBdBformule=1.76+6.02*N
SBdB_formule=1.77+6.02*N ;
%-----reconstruction du signal -----
xlreconstructed = zeros(1,length(t)); %pré-allocation pour la
rapidité
for i = 1:1:length(t)    % indice temporel
    for n = 1:1:samples % indice des échantillons
        xlreconstructed(i) = xlreconstructed(i) +
xlresampled(n)*sinc((t(i)-n*Ts)/Ts); %%% signal reconstruit
    end
end
end
figure(2)
subplot(211)
plot(t,x); grid;% signal test
hold on
stem(ts,xlresampled,'r'); grid; title('signal de test en bleu
et signal échantillonné en rouge');
subplot(212)
plot(t,xlreconstructed);grid;title('signal reconstruit');

```

Questions

1. Choisir des fréquences F_s quelconques pour les trois cas d'échantillonnage (sous échantillonnage, échantillonnage stricte et sur échantillonnage), interpréter les résultats.
2. Changer le nombre de bits de codage et tracer un tableau qui représente les erreurs de quantifications en décibels. Interpréter ces résultats. Comparer les erreurs de quantifications en Décibels
3. Donner un schéma de reconstitution du signal analogique à partir de ces échantillons et donner sa formule qui correspond à celle utilisée par le programme.

Conclusion

TP5 : opérations sur signaux discrets

But : il s'agit d'un TP de familiarisation sur les opérations des signaux discrets

Taper les fonctions suivantes et les sauvegarder par leurs noms déjà affectés.

```
function [y, y1, y2, n]=sigadd(x1,n1,x2,n2)
n = min(min(n1),min(n2)):max(max(n1),max(n2));
y1=zeros(1,length(n));
y2= y1;
y1((n>=min(n1))&(n<=max(n1))==1)=x1;
y2((n>=min(n2))&(n<=max(n2))==1)=x2;
y = y1+y2;
end
```

```
function [y,y1,y2,n]=sigmult(x1,n1,x2,n2)
n =min(min(n1),min(n2)):max(max(n1),max(n2));
y1 =zeros(1,length(n));
y2 = y1;
y1((n>=min(n1))&(n<=max(n1))==1)=x1;
y2((n>=min(n2))&(n<=max(n2))==1)=x2;
y =y1.*y2;
end
```

```
function [y,n]=sigfold(x,n)
y=fliplr(x);
n=fliplr(n);
end
```

```
function [xr,tr]=sigrev(x,t)
%time reverting a signal
%inputs:x,t input signal amplitude and instants
%xr,tr time reversed signal amplitude and instants
lx=length(x);
tr=-t(lx:-1:1);
xr=x(lx:-1:1);
end
```

```
function [x,tx]=impseq(n0,n1,n2)
tx=n1:n2;x=((tx-n0)==0);
end
```

```
function [x,tx]=stepseq(n0,n1,n2)
tx=n1:n2;x=((tx-n0)>=0);
end
```

```
function [xs,ts]=sigshift(x,t,to)
%inputs: x,t input signal amplitude and instants ; to shift value
%xs,ts shifted signal amplitude and instants
xs=x; ts=t+to;
end
```

Ouvrir pour chaque programme un nouveau **m.file** puis exécuter

Programme 1

```
x1 = [1 2 3 4];
x2 = [1 2 3 4 5 6 7];
n1 = 1:length(x1);
n2 = 1:length(x2);
[y,y1,y2,n] = sigadd(x1,n1,x2,n2);
subplot(311); stem(n,y); xlabel('n'); ylabel('y'); grid;
subplot(312); stem(n,y1); xlabel('n'); ylabel('y1'); grid;
subplot(313); stem(n,y2); xlabel('n'); ylabel('y2'); grid;
```

Programme2

```
x1=[1 2 3 4]; x2=[1 2 3 4 5 6 7];
n1 =1:length(x1); n2 =1:length(x2);
[y,y1,y2,n]=sigmult(x1,n1,x2,n2);
subplot(311);stem(n,y);xlabel('n');ylabel('y');grid;
subplot(312);stem(n,y1);xlabel('n');ylabel('y1');grid;
subplot(313);stem(n,y2);xlabel('n');ylabel('y2');grid;
```

Programme 3

```
x = [1 4 9 16 0 0];
n = 1:6;
[y,n] = sigfold(x,n);
subplot(211);stem(n,x);xlabel('n');ylabel('x');grid;
subplot(212);stem(n,y);xlabel('n');ylabel('y');grid;
```

Energie du signal:

```
x=[1 2 3 4 5 6 7];
Ex1=sum(x.*conj(x))
Ex2= sum(abs(x).^2)
```

Comparer Ex1 et Ex2

Exercice :

Ecrire un programme qui trace le signal : $x(n) = 2\delta(n+2) - \delta(n-4)$ dans l'intervalle $-5 \leq n \leq 5$

On considère le signal :

```
x=[1 2 3 4 5 6 7 6 5 4 3 2 1]; pour n=-2:10;
```

Générer et tracer les séquences suivantes à l'aide des fonctions précédemment définies :

$$x1(n) = 2x(n-5) - 3x(n+4)$$

$$x2(n) = x(3-n) - x(n)x(n-2)$$

Tracer les signaux suivants et déterminer la période du signal si elle existe dans l'intervalle $n=0 :12$

$$x1(n) = \cos(0.8\pi n + 1)$$

$$x2(n) = \sin(\sqrt{2}\pi n - 2)$$