Université Djilali Bounaama de Khemis Miliana Faculté des Sciences et de la Technologie

Département de Technologie/ Niveau et Spécialité: 1ère Année Master Telecom

Matière : TP Signaux aléatoires et Processus stochastiques Enseignant :A.BOUNIF

TP N°04 : Filtrage des signaux aléatoires

I) Filtrage d'un bruit blanc.

On peut effectuer un filtrage passe-bas de type FIR(Finite Impulse Response) par un simple lissage, en remplaçant chaque échantillon par la moyenne de cet échantillon et des L−1 échantillons précédents :

$$y(l) = \frac{1}{L}\sum_{k=0}^{L-1} x(l-k) \tag{1}$$

a) Lissage du signal initial à des échelles de 8 et 32 échantillons en Matlab

```
x = randn(256,1); % réalisation de longueur 256 d'un bruit blanc Gaussien
L = 8;  h = ones(L,1)/L;% réponse impulsionnelle finie constante sur son support(de longueur 8)
 y8 = filter(h,1,x); % filtrage
L = 32; h = ones(L,1)/L; y32 = filter(h,1,x);
plot(x); axis tight
hold on ,plot(y8,'r'); plot(y32,'g')
```

On peut noter que la courbe verte est plus lisse que la rouge, elle même plus lisse que la bleue. On voit bien l'effet du lissage.

b) Filtrage d'un bruit blanc uniforme

- Générer un signal aléatoire d'entrée de 50 échantillons dont l'amplitude est uniformément distribuée sur l'intervalle [-4 5] ;
- Filtrer ce signal par une convolution directe en utilisant la réponse impulsionnelle suivante : h(n)=[1/8 1/4 1/4 1/4 1/8] ;
- Tracer sur le même graphe les signaux d'entrée et de sortie et expliquer l'effet du filtrage.

II) Simulation du filtre Moyenne mobile (Moving Average filter) dans Matlab

Le filtre à moyenne mobile est un simple filtre passe-bas FIR (Finite Impulse Response) couramment utilisé pour lisser un ensemble de données/signaux échantillonnés. Il prend des échantillons d'entrée à la fois, prend la moyenne de ces échantillons et produit une seule valeur de sortie. Ce filtre a une excellente réponse dans le domaine temporel mais une mauvaise réponse en fréquence. Dans Matlab, on peut utiliser les fonctions **filter** or **conv** (convolution) pour implementer ce filtre.

a) Implémentation

L'équation de différence pour un filtre à moyenne mobile à temps discret à L-point avec une entrée représentée par le vecteur {x} et le vecteur de sortie moyen {y}, est donnée l'équation 1.

Par exemple, un filtre FIR de moyenne mobile à 5 points prend les quatre échantillons d'entrée actuels et précédents et calcule la moyenne. Cette opération est représentée comme le montre la figure 1 avec l'équation de différence suivante pour la relation entrée-sortie en temps discret.

$$\begin{aligned} y[n] &= \frac{1}{5}\left(x[n] + x[n-1] + x[n-2] + x[n-3] + x[n-4]\right) \\ &= 0.2\left(x[n] + x[n-1] + x[n-2] + x[n-3] + x[n-4]\right) \end{aligned} \tag{2}$$

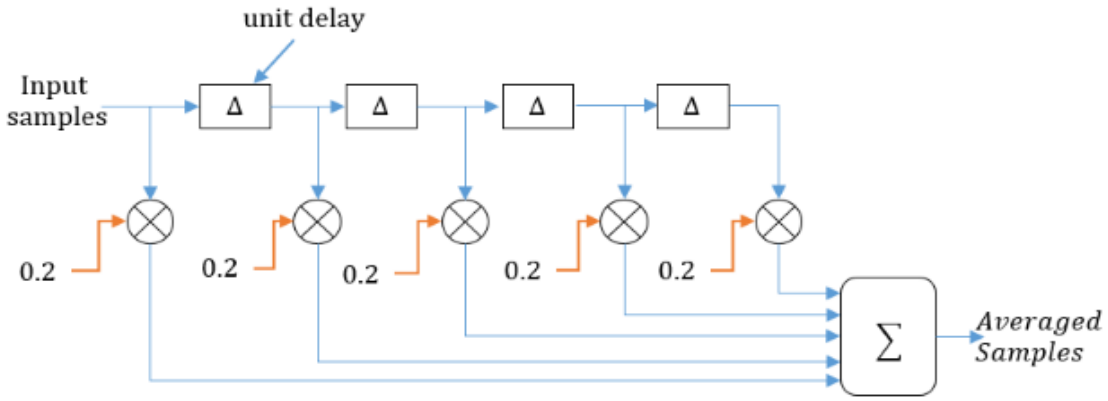

Figure 1: Discrete-time 5-point Moving Average FIR filter

Le retard unitaire illustré à la figure 1 est réalisé par l'une ou l'autre des deux options :

Représenter les échantillons d'entrée sous forme de tableau dans la mémoire de l'ordinateur et les traiter

Utilisation des registres à décalage D-Flip flop pour la mise en œuvre du matériel numérique.

Transformation en Z et Fonction de transfert

En traitement du signal, retarder un signal x[n] de k période d'échantillonnage (retard unitaire) équivaut à multiplier X[z] (la transformation en Z de x(n)) par z^{-k}. En appliquant cette idée, nous pouvons trouver la transformée en Z du filtre à moyenne mobile à 5 points dans l'équation (2) comme suit :

$$y[n] = 0.2\left(x[n] + x[n-1] + x[n-2] + x[n-3] + x[n-4]\right)$$
$$Y[z] = 0.2\left(z^0 + z^{-1} + z^{-2} + z^{-3} + z^{-4}\right)X[z]$$
$$Y[z] = 0.2\left(1 + z^{-1} + z^{-2} + z^{-3} + z^{-4}\right)X[z] \qquad (3)$$

De même, la transformée en Z du filtre générique de moyenne mobile de l'échantillon L de l'équation (1) est

$$Y(z) = \left(\frac{1}{L}\sum_{k=0}^{L-1} z^{-k}\right)X(z) \qquad (4) \qquad\qquad H(z) = \frac{Y(z)}{X(z)} = \frac{1}{L}\sum_{k=0}^{L-1} z^{-k} \qquad (5)$$

La fonction de transfert décrit la relation entrée-sortie du système et pour le filtre moyenne mobile du point L, est donnée par (5).

En général, $Y(z)$ (la transformée en Z de $y(n)$) de la sortie d'un filtre à temps discret est liée à $X(z)$ (transformée en Z de l'entrée $y(n)$) par :

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots + b_n z^{-n}}{a_0 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_m z^{-m}} \qquad (6)$$

où $\{b_i\}$ et $\{a_i\}$ sont les coefficients du filtre et

l'ordre du filtre est le maximum de *n* et *m*.

Pour implémenter l'équation (6) à l'aide de la fonction de filtre, la fonction Matlab est appelée comme

B = [b0, b1, b2, ..., bn] % coefficients du numérateur

A = [a0, a1, a2, ..., am] % coefficients du dénominateur

y = filter(B,A,x) %filter input x and get result in y

Nous pouvons noter à partir de l'équation de différence et de la fonction de transfert du filtre à moyenne mobile du point L, les valeurs suivantes pour les coefficients du numérateur $\{b_i\}$ et les coefficients du dénominateur$\{a_i\}$.

$$\{b_i\} = \left\{\frac{1}{L}, \frac{1}{L}, \dots, \frac{1}{L}\right\} \quad , i = 0,1, \dots, L-1$$

$$\{a_i\} = 1 \qquad\qquad , i = 0 \tag{7}$$

Il existe une différence entre l'utilisation de la fonction **conv** et de la fonction de **filter** pour implémenter un filtre FIR. La fonction **conv** donne le résultat d'une convolution complète et la longueur du résultat est **length(x)+ L -1**. Alors que la fonction de **filter** donne une sortie de même longueur que celle de l'entrée.

Exemple 1 : Pour un filtre à 5 coefficients on peut le simuler par :

```
L = 5; B = ones(1,L)/L; %numerator coefficients
A = [1]; %denominator coefficients
x = rand(1,100); %random samples for x
y1 = filter(B,A,x); %filter input x and get result in y
y2 = conv(x,ones(1,L)/L);
figure
subplot(311), plot(x)
title('original signal')
xlabel('samples')
ylabel('magnitude')
subplot(312), plot(y1)
title('filtered signal with "filter" function')
subplot(313), plot(y2)
title('filtered signal with "conv" function')
```

Exemple 2

```
%MATLAB code for generating 50 samples of MA process with b 0 = 1,b 1 = 2:
% Alternatively, we can use the convolution function in MATLAB:
b0=1; b1=2; N=50;
w=randn(1,N+1); % generate N+1 white noise samples
 b= [b0 b1]; % "b" is an vector
y=conv(b,w); % signal length is "N+1"+"2"-1
x=y(2:N+1); % remove the transient signals
figure
subplot(211), plot(x)
title('original signal')
xlabel('samples')
ylabel('magnitude')
subplot(212), plot(y)
title('filtered signal with "conv" function')
```

$$\Phi_{xx}(\omega) = \left|1 + 2e^{-j\omega}\right|^2 \cdot \sigma_w^2 = \left|1 + 2e^{-j\omega}\right|^2$$

La PSD pour un processus MA est

Elle peut être tracée en utilisant la **freqz** de MATLAB:

```
b0=1; b1=2; b= [b0 b1]; a=1;
[H,W] = freqz(b,a); % "H" is complex frequency response
PSD = abs(H.*H);
figure
plot(W/pi,PSD);
```

Pour évaluer le processus MA généré par MATLAB, nous utilisons :

$$\Phi_{xx}(\omega) = \lim_{N\to\infty} E\left\{\frac{1}{N}\left|\sum_{n=0}^{N-1} x(n)e^{-j\omega n}\right|^2\right\}$$

N → ∞ ⇒ N →100 E{} ⇒ average of 100 independent simulations

```
%MATLAB code:
N=100; b= [1 2]; % "b" is a vector
for m=1:100 % perform 100 independent runs
w=randn(1,N+1); % generate N+1 white noise samples
y=conv(b,w); % signal length is "N+1"+"2"-1
x=y(2:N+1); % remove the transient signals
p(m,:) = abs(fft(x).*fft(x));
end
psd = mean(p)./100;
index = 1/50:1/50:2;
figure
plot(index,psd);
axis([0, 1, 0 10]);
```

N → ∞ ⇒ N →10000 E{} ⇒ average of 10000 independent simulations

Il est également nécessaire de supprimer les signaux transitoires dans les processus AR et ARMA à cause de la non-stationnarité due aux pôles :

Pour un processus AR du premier ordre : $x(n) = ax(n-1) + w(n)$ et $R_{xx}(n, n+m) = a^m\left(\frac{1-a^{2(n+1)}}{1-a^2}\right)\sigma_w^2$

- Non stationnaire car $R_{xx}(n, n+m)$ dépend de n.
- Pour n suffisamment grand, $\left|a^{2(n+1)}\right| \ll 1$, on peut le considérer stationnaire.
- Puisque a est le pôle, extension aux processus généraux AR et ARMA : $\left|{p_i}^{2(n+1)}\right| \ll 1$ pour tous

les pôles.

On Suppose $\left|a^{2(n+1)}\right| \leq 0.0001$ est requis et le paramètre AR est a = −0,9. Le n requis est calculé comme

$$\left|(-0.9)^{2(n+1)}\right| = 0.9^{2(n+1)} \leq 0.0001 \quad \rightarrow \quad n \geq 43$$

```
%MATLAB code for generating 50 samples of the AR process:
M = 43; N = 50;a = -0.9;y(1) = 0;
for n=2:M+N
y(n) = a*y(n-1)+randn;
end
x=y(M+1:M+N);
plot(x);
```

Pour le système FIR, nous pouvons suivre le processus MA, tandis que pour le système IIR, nous utilisons le processus ARMA. Les signaux transitoires peuvent être supprimés si nécessaire comme dans les processus MA, AR et/ou ARMA.