TRAITEMENT DU SIGNAL

Cahier de Travaux Pratiques

3° année

AVANT-PROPOS

Les travaux pratiques de traitement du signal ont pour objectifs tout d'abord d'apprendre à utiliser le logiciel Matlab, logiciel très répandu dans l'industrie et le monde de la recherche, de mettre en pratique les notions théoriques du cours et enfin d'illustrer concrètement quelques applications.

L'évaluation ?

Contrairement aux années passées, l'évaluation ne sera pas basée sur les comptes-rendus de chaque séance mais sur un examen individuel qui se déroulera lors de la semaine d'examen.

Le sujet proposé sera tiré des études réalisées (théorie comprise) dans les trois premières séances qui participent donc pleinement à la préparation de l'examen de TP. Pour ces trois premières séances, il s'agira donc de bien comprendre les notions abordées et de réaliser les exercices proposés.

Ce qui n'empêche pas …

… de prendre ses propres notes sous la forme d'un compte-rendu « pour soi » ou « pour le binôme » ! Alors pour rappel (que ce soit pour un compte-rendu personnel ou évalué), un compte-rendu se doit d'être clair, synthétique et soigné (ce qui n'implique pas une rédaction dactylographiée !).

… de rédiger une introduction et une conclusion.

… de commenter, justifier, et critiquer les résultats obtenus ; en faire état ne suffit pas. Un graphique n'est utile que s'il illustre le compte-rendu, la courbe doit alors comportée un titre, une légende, des échelles … et doit être commentée et analysée.

L'idée générale est de rédiger un compte-rendu dont la lecture vous permettrait dans quelques années (lorsque vous ne saurez plus rien sur le sujet) ou quelques semaines (lorsqu'il s'agira de réviser pour l'examen) de comprendre et d'être capable de refaire le même type d'exercice. Le plus important est donc de faire apparaître les éléments-clés apparus au cours de votre travail. Ces éléments concernent les résultats de traitement du signal mais aussi les difficultés surmontées ou les éléments (mieux) compris.

… de développer proprement vos programmes avec commentaires, noms de variables explicites, utilisation d'appels à fonctions, etc. (tout est dit dans les cours d'informatique !).

SOMMAIRE

Rappels concernant le logiciel Matlab

Si ces rappels ne s'avéraient pas suffisants, il faut aussi savoir que de très nombreux cours sur Matlab sont disponibles sur Internet et que l'aide de Matlab est très utile et très complète avec un descriptif de chaque fonction et de nombreux exemples.

Matlab (MATrix LABoratory) a été originalement écrit pour fournir un moyen simple de programmer avec des matrices. Ses principales utilisations concernent : mathématiques et calculs ; développements d'algorithmes ; modélisation, simulation et prototypage ; analyse de données, exploration et visualisation ; graphiques scientifiques ; applications incluant un outil graphique pour construire des interfaces graphiques.

Matlab est un langage interprété conçu pour les calculs techniques. Matlab combine des calculs efficaces, une visualisation conviviale des résultats et un environnement de programmation facile à utiliser.

Matlab est un élément interactif dont l'élément de base est un tableau qui ne nécessite pas de dimensionnement. Ceci permet de résoudre de nombreux problèmes techniques particulièrement ceux qui sont formulés avec des matrices et des vecteurs. Sous cette condition ce sera infiniment plus rapide que d'écrire ou exécuter un programme de type C.

Enfin, Matlab est un logiciel de la société Mathworks et il faut savoir qu'il existe des clones de ce logiciel comme par Scilab. L'un des principaux avantages de Matlab est de fournir une aide en ligne détaillée et de nombreuses librairies de fonctions, l'ensemble de ces librairies n'étant disponible qu'à condition de posséder l'ensemble des « toolboxes », boites à outils dédiées à un domaine d'applications particulier.

Programmer avec Matlab

Les matrices

```
>> A=[1 2 3 ; 4 5 6 ; 7 8 9]
A  =   1 2 3
       4 5 6
       7 8 9
```

séparateur de colonnes : blanc ( ) ou virgule (,)

séparateur de ligne : point virgule (;)

début et fin de liste : […]

Les constantes

pi = 3,14159265

i ou j = sqrt(-1), unité imaginaire

Les opérateurs

+ addition

- soustraction

* multiplication

/ division

^ puissance

' transposition conjuguée

not ~ NON

and & ET

or | OU

xor OU EXCLUSIF

.* multiplication élément par élément

./ division élément par élément

Structures de contrôles

```
if expression1
   fonction1 ;
elseif expression2
   fonction2 ;
else fonction3 ;
end
```

```
switch expression
  case cas1
     fonction1 ;
  case cas2
     fonction2 ;
  otherwise
     fonction3 ;
```

```
while i<=10
   fonction ;
   i=i+1
end
```

```
for i=1 :10
   fonction ;
end
```

Fonctions

Fonctions élémentaires (trigonométriques, exponentielles, complexes, …)

>> help elfun

Fonctions pour les matrices

>> help elmat

Fonctions mathématiques (spécilisées, transformation des coordonées, …)

>> help specfun

Scripts et fonctions

Deux types de fichiers .m

Scripts

N'acceptent pas d'arguments d'entrée et ne retournent pas d'arguments de retour

Travaillent directement sur les données de l'espace de travai

Matlab exécute simplement les commandes du fichier

Fonctions

Peuvent accepter et retourner des arguments

Les variables internes sont locales à la fonction

Travaillent sur leur propre espace de travail

Le nom du fichier .m doit être le même que celui de la fonction

Vectorisation

Il faut éviter l'usage de boucle si possible pour gagner en temps de calcul.

```
N=50 ;
for i=1 :N
  y(i)=sin(2*pi*i/N);
end
```

```
N=50;
i=1:N;
y=sin(2*pi*i/N);
```

Précision et format des données

char, short, int, long, float, …

%s, %5d, …

Visualisation des données

Affichage sur l'écran (ligne de commande) : disp, sprintf, …

Types de représentation graphiques : plot, stem, stairs,hist …

Paramètres de représentation graphiques : subplot, grid, title,gtext, axis …

Fonctions pré-programmées

ones, zeros, sin, square, sawtooth, rand, randn, eye, …

Ecriture et lecture de fichiers

save, load, …

Aide à la programmation

edit « nom de la fonction matlab » (edit exemplematlab ;) : édition du fichier .m correspondant

whos : affichage de l'ensemble des variables et de leurs dimensions (équivalent du « workspace »)

help « nom de la fonction matlab » : description rapide pour utiliser la fonction

Aide à la création d'interface graphique

guide

Pour plus d'informations, se référer en premier lieu à l'aide de Matlab.

Séance 1 : Introduction au traitement du signal numérique

L'objectif de ce travail est d'aborder la manipulation de signaux discrets en utilisant le logiciel Matlab ; les exemples traités sont simples et illustrent les représentations temps et fréquence des signaux ainsi que les problèmes liés à l'échantillonnage. A l'issue de la séance, il faut notamment être capable d'écrire un programme Matlab qui génère tout type de signal, calcule sa transformée de Fourier, et trace les différentes représentations associées ET il faut aussi savoir analyser et comprendre une courbe obtenue dans un contexte donné !

1. Génération d'un signal numérique

1.1) Générer un signal numérique, x1, représentant N = 100 échantillons d'une cosinusoïde de fréquence f_0 = 1 kHz, la fréquence d'échantillonnage étant fixée à 8 kHz.

1.2) Tracer ce signal en faisant apparaître en abscisse le temps. Pour cela aidez vous de la fonction « plot ».

1.3) Déterminer l'énergie du signal : Comment varie l'énergie lorsque N augmente ? Commenter ce résultat.

1.4) Déterminer la puissance du signal : Comment varie la puissance lorsque N augmente ? Est-ce en accord avec la théorie ?

2. Transformation de Fourier Discrète

2.1) Calculer le spectre du signal x1 avec la fonction FFT pour N = 100.

2.2) Tracer le module du spectre correspondant en fonction de la fréquence. Commenter le spectre obtenu. Est-il en accord avec ce que vous vous attendiez à trouver ? Quelle est la fréquence exacte du signal indiquée par le spectre ?

2.3) Tracer à nouveau le spectre mais pour N = 1000. Commenter le spectre obtenu. Quelle est la fréquence du signal indiquée par le spectre ? Conclure sur l'influence du nombre de points dans la déduction de la fréquence du signal.

2.4) Tracer à nouveau le spectre mais pour N = 104 et N = 405. Quelle est la fréquence du signal indiquée par les spectres ? Est-ce en accord avec la conclusion de la question 23) ? Expliquer précisément pourquoi nous observons cela et conclure sur ce cas particulier.

2.5) Pourquoi l'amplitude des raies varient pour N = 1000 et N = 2000.

2.6) Déterminer l'énergie du spectre pour N = 100. Comparer avec l'énergie obtenue à la question

(en discret, l'énergie du spectre est : **1/N***sum(abs(spectre).^2)).

3. Transformée de Fourier inverse - Reconstruction

A partir du spectre du signal x1, reconstruire le signal grâce à la transformée de Fourier inverse (fonction ifft). Comparer le signal reconstruit à l'aide de la FFT inverse, au signal original de la question 11).

Quelle condition doit-on respecter pour effectuer cette opération ?

4. Théorème de Shannon

4.1) Tracer les spectres des signaux numériques sinusoïdaux de fréquence 1 kHz et 7 kHz, échantillonnés à la fréquence de 8 kHz avec N = 100 échantillons. Expliquer les résultats obtenus.

4.2) D'une façon générale, quelle est la fréquence maximale autorisée sans qu'apparaisse un phénomène de repliement de spectre ? Si cette fréquence est dépassée, quelle est la fréquence apparaissant en réalité dans le spectre échantillonné ?

4.3) Générer deux sinusoïdes de fréquence 200 Hz échantillonnée à la fréquence F_{e1} = 1000 Hz avec N1 = 80 échantillons et échantillonnée à la fréquence F_{e2} = 250 Hz avec N2 = 20 échantillons.

4.4) Tracer sur une même figure ces deux signaux en faisant apparaître les échantillons.

4.5) Grâce à la fonction interp de matlab, interpoler les deux signaux d'un facteur 5. Tracer sur une même figure les signaux obtenus. Expliquer les résultats : pour cela vous pouvez mesurer la fréquence de chaque signal et la comparer avec celle obtenue par la fonction FFT. Conclure sur l'intérêt du la limite de Shannon.

5. Exercice d'approfondissement

Réaliser un algorithme qui calcule directement la Transformée de Fourier Discrète (TFD) du signal généré en utilisant la formule théorique du cours. Ecrire le programme correspondant.

Séance 2 : Observation spectrale

L'objectif principal de ce TP est d'étudier plus particulièrement la représentation fréquentielle des signaux. Il s'agit de déterminer les bonnes (voire les meilleures) conditions d'observation du contenu spectral des signaux. Trois études de cas illustrent des exemples permettant de mettre en relief les méthodes d'observation et les options possibles. A l'issue de la séance, il faut être capable de déterminer les paramètres permettant d'observer correctement le spectre d'un signal, mettre en œuvre la procédure correspondante et avoir compris les notions théoriques associées.

Avant-propos : afin d'examiner plus précisément les caractéristiques des spectres des signaux, deux caractéristiques très importantes sont la précision et la résolution fréquentielle. La précision est rattachée à la mesure d'une fréquence. Par exemple, lorsque le spectre d'une sinusoïde est déterminé par une TFD, le tableau des valeurs obtenues donne une valeur approchée de la fréquence : son exactitude ou précision est directement liée au nombre de points. La précision ne doit pas être confondue avec la résolution fréquentielle qui est la capacité à détecter des fréquences distinctes contenues dans un même signal.

1. Observation d'une sinusoide

Soit la suite x(n) = cos(2π.F0.n.Te) obtenue par échantillonnage du signal cos(2πF0t). La fréquence F0 sera prise égale à 1 kHz, la fréquence d'échantillonnage à Fe = 8 kHz et la suite sera composée de Ns = 80 échantillons.

1.1) Représenter le module de la Transformée de Fourier à discrète X(f) de la suite x(n), en utilisant la fonction **fft**.

1.2) Calculer et représenter le spectre de X(f) en utilisant **fft(x,Nfft)** avec Nfft = 2000. Comparer le résultat avec celui de la question précédente.

En vous aidant de l'aide de matlab, expliquer l'opération effectuée par l'option Nfft. Expliquer alors le résultat obtenu.

1.3) Calculer et représenter le spectre de X(f) avec Ns = 2000. Comparer le résultat avec celui de la question précédente.

1.4) Représenter le module calculé en 12) en décibels. Quel est l'intérêt de cette représentation ? (attention sous matlab : la fonction log correspond au ln et la fonction log10 correspond au log).

2. Observation d'un signal modulé en amplitude

Soit le signal m(t), à temps continu, de spectre M(f) : $m(t) = \cos(2\pi f_1 t) + 1{,}8.\cos(2\pi f_2 t) + 0{,}9.\cos(2\pi f_3 t)$ avec f_1 = 2310 Hz, f_2 = 3750 Hz et f_3 = 4960 Hz.

Pour transmettre par ondes radiofréquences ce signal, une modulation d'amplitude avec porteuse est effectuée telle que le signal modulé s'écrit : $s(t) = (1 + k.m(t)).\cos(2\pi f_0 t)$ avec f_0 = 50 kHz et k = 0,25.

2.1) A faire avant le TP : Déterminer **théoriquement** l'expression du spectre de m(t), M(f).

2.2) A faire avant le TP : Déterminer **théoriquement** l'expression du spectre de s(t), S(f). En déduire les fréquences contenues dans le signal modulé.

2.3) Choisir **en justifiant** une fréquence d'échantillonnage puis en déduire la durée minimale d'observation du signal qui permet de distinguer par analyse de Fourier, les fréquences présentes dans le spectre de s(t).

2.4) Quel est le nombre de points de la FFT nécessaire, si l'on souhaite lire le spectre avec une précision de 10 Hz ?

2.5) Ecrire un programme qui trace le spectre du signal modulé avec une résolution de 10 Hz.

3. Effets du fenêtrage sur la résolution

3.1) Générer le signal x(n) obtenue par échantillonnage du signal exp($2\pi j F_0$*n*Te) à la fréquence d'échantillonnage F_e = 1/Te pour une fréquence normalisée $f_0 = F_0 / F_e = 0.2$. Vous prendrez un nombre d'échantillons N égal à 2000.

3.2) Tracer le spectre du signal x(n) en fonction de la fréquence normalisée F_0 / F_e. Quelle est la fréquence du signal mesurée sur ce spectre ?

L'objectif de l'exercice est d'extraire une partie du signal de taille Ns=32 grâce à une opération de fenêtrage. Pour le moment, vous avez surtout utilisé la fenêtre de type rectangle pour réaliser cette opération. En pratique, il existe beaucoup de fenêtre disponible. Nous allons à présent comparer la fenêtre rectangle à une autre fenêtre très connue, la fenêtre de Hamming.

Le fait de limiter à Ns le nombre d'échantillons d'un signal peut être vu comme la multiplication terme à terme de la totalité du signal x(n) (de longueur N) par une fenêtre (de longueur Ns). Chaque échantillon est multipliée par $c \cdot \omega(n)$, où c est une constante et $\omega(n)$ désigne :

- soit la fenêtre rectangulaire définie par : $\omega_R(n) = 1 \ \forall\, n \in [0\,;\, L-1]$, et 0 ailleurs ;

- soit la fenêtre de Hamming définie par : $\omega_{HL}(n) = 0{,}54 - 0{,}46 \cdot \cos\left(\frac{2\pi n}{L-1}\right) \ \forall\, n \in [0\,;\, L-1]$, et 0 ailleurs.

3.3) Pour une fenêtre quelconque, déterminer **théoriquement** la constante c telle que l'amplitude maximale de la TFD du signal fenêtré en f_0 soit égale à 1. Montrer que c = 1/L pour une fenêtre rectangle et 1/(0.54*L) pour celle de Hamming

3.4) Ecrire un programme qui affiche le module en décibels de la Transformée de Fourier de x(n) pour la pondération rectangulaire et la pondération de Hamming.

3.5) Pour les deux fenêtres, relever la largeur du lobe principal et la hauteur du premier lobe secondaire (la hauteur sera prise en décibels par rapport au lobe principal).

3.6) Illustrer vos conclusions à partir de signaux simples comme la somme de signaux sinusoïdaux de fréquences voisines ou/et d'amplitudes différentes. Montrer en particulier l'importance du choix de la fenêtre d'apodisation en fonction du signal à étudier.

Séance 3 : Filtrage RIF

La fréquence d'échantillonnage est choisie égale à F_e = 8 kHz.

1. Synthèse d'un filtre

1.1) A faire avant le TP : En utilisant la méthode de la fenêtre (décrite dans le cours), déterminer **théoriquement** la suite h(n) des N = 35 coefficients d'un filtre RIF passe-bas idéal dans la bande [0, 1000 Hz].

1.2) Représenter la réponse impulsionnelle déterminée. (Attention à la fonction sinc de Matlab. Voir la section « more about » de l'aide)

1.3) Vérifier le résultat obtenu en traçant la réponse en fréquence du filtre obtenu. Comparer gabarie intial du filtre, commenter et conclure.

1.4) Représenter en fonction de la fréquence, le module en décibels et la phase en radians du filtre calculé. Commenter et vérifier que le filtre synthétisé est un filtre à phase linéaire. Quel peut être l'avantage d'une telle caractéristique ?

2. Initiation aux outils graphiques de Matlab dédiés au traitement du signal

21. Synthèse de filtre : En utilisant l'outil graphique matlab « fdatool », synthétiser le filtre de la question 1 avec les mêmes spécifications (même fréquence d'échantillonnage, même filtre, même méthode de synthèse …). Visualiser alors les différentes représentations du filtre : réponse impulsionnelle, réponse en fréquence, pôles et zéros, coefficients de la réponse impulsionnelle, etc. Vérifier la concordance des résultats.

3. Génération d'un signal composite

3.1) Créer un signal sinusoïdal composite noté « sig » d'une durée totale de 2 secondes tel que :

si t $\in$ [0 ; 0,3 s], « sig » d'amplitude 3 et de fréquence 500 Hz ;

si t $\in$ [0,3 s ; 0,8 s], « sig » d'amplitude 1 et de fréquence 800 Hz ;

si t $\in$ [0,8 s ; 1,2 s], « sig » d'amplitude 2 et de fréquence 1200 Hz ; si t $\in$ [1,2 s ; 1,7 s], « sig » d'amplitude 2 et de fréquence 400 Hz ;

si t $\in$ [1,7 s ; 2 s], « sig » d'amplitude 3 et de fréquence 1500 Hz.

3.2) Représenter ce signal ainsi que le module de son spectre.

3.3) Représenter le spectre sous la forme d'un spectrogramme.

4. Filtrage

41. Appliquer le filtre synthétisé à la question 1 au signal de la question 3 en utilisant la fonction « filter ».

42. Représenter sur une même figure, d'une part le signal avant et après filtrage et d'autre part, le module du spectre du signal avant et après filtrage. Analyser et commenter.

43. Représentations temporelle et fréquentielle d'un signal à l'aide de l'outils sptool : A partir du signal généré à la question 3 et en utilisant l'outil graphique matlab « sptool », importer ce signal dans le logiciel et visualiser ses réponses temporelles et fréquentielles. Vérifier la concordance des résultats. Observer rapidement les différentes méthodes d'analyse spectrale proposées.

45. Filtrage à l'aide de sptool : A partir du filtre synthétisé à la question 1 et 2, importer dans « sptool » ce filtre. Appliquer le alors sur le signal précédent et observer les résultats obtenus.

<u>Exercice 3</u> : Erreur de quantification

Le but de cet exercice est de comprendre l'erreur commise lors de la quantification d'un signal aléatoire. Pour un gain de temps, les programmes sont donnés en annexe.

51. Générer un signal aléatoire de 2000 échantillons et d'écart-type égal à un, en utilisant la fonction « randn ». Prélever, à partir de l'aide de Matlab, les caractéristiques du signal ainsi généré. Déterminer sa puissance moyenne.

52. Quantifier (par arrondi) le signal aléatoire généré précédemment pour un nombre de bits allant de 1 à 7. La fonction « floor » ou ses semblables pourra être utilisée.

53. Dans chaque cas (nombre de bits allant de 1 à 7), calculer et tracer l'histogramme de l'erreur commise lors de la quantification. Cette erreur est définie comme la différence entre le signal aléatoire généré et celui quantifié. L'histogramme pourra être représenté en utilisant la fonction « hist ». Vérifier l'hypothèse d'équirépartition du bruit de quantification.

54. Toujours dans chaque cas, déterminer le rapport signal sur bruit en utilisant d'une part, le calcul direct à partir du signal généré et de l'erreur commise lors de la quantification, et d'autre part, la relation déterminée théoriquement. Pour la comparaison, représenter sur un même graphique ces rapports signal sur bruit en fonction du nombre de bits.

Remarque : pour déterminer le facteur de forme, il faut utiliser la relation $A = F.\sigma$, dans laquelle $\sigma 2$ représente la puissance du signal.

ANNEXE

```
% 1. Etude de la quantification
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% REMARQUE
% ---> SI le signal est à ddp uniforme :OK
% ---> SI le signal est à ddp gaussienne : il faudrait F=2,3
%      or théoriquement on trouve F=3
%      avec un décalage théorie - simulation de 5 dB
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initialisation
clear all ; close all ; clc

% Génération d'un signal aléatoire blanc gaussien
N=2000;              % Nombre d'échantillons
sigma=1/3;           % Par définition pour obtenir un signal compris entre -1 et
+1
x=sigma*randn(1,N); % Coefficient permettant d'obtenir une amplitude à un dans
l'intervalle + ou - 3 sigma
% x=2*rand(1,N)-1;  % Loi uniforme : OK avec F2 sans saturation
A=3*sigma;           % Amplitude du signal
F1=A/sigma;          % Facteur de forme sans tenir compte de la saturation

% Effet de saturation
eps=0.0001;
for ii=1:N
    if x(ii)>A
        x(ii)=A-eps;
    end
    if x(ii)<-A
        x(ii)=-A;
    end
end

% Calcul de la puissance et du facteur de forme
Ps=x*x'/length(x);
Ps2=var(x);                  % Remarque : 2° calcul de la puissance du signal
F2=max(x)/sqrt(Ps);          % Remarque : 2° calcul du facteur de forme en tenant
compte de la saturation

% Calcul du pas de quantification
Nbit=[1 2 3 4 5 6 7];
NbNiveau=2.^Nbit;
q=2*A./(NbNiveau');

% Quantification
for ii=1:length(Nbit)
    xQ(ii,:)=q(ii)*floor(x/q(ii))+q(ii)/2;
end

% % Quantification par arrondi
% for ii=1:length(Nbit)
%     x1(ii,:)=x*NbNiveau(ii)/2;
%     x2(ii,:)=floor(x1(ii,:))+0.5;
%     xQ(ii,:)=x2(ii,:)*q(ii);
% end

% Quantification par troncature
```

```
% for ii=1:length(Nbit)
%     x1(ii,:)=x.*NbNiveau(ii)/2;
%     x2(ii,:)=floor(x1(ii,:)); % ou fonction ceil
%     xQ(ii,:)=x2(ii,:)*q(ii);
% end

% Affichage des signaux quantifiés
figure(1)
subplot(221);plot(xQ(1,:));hold on;plot(xQ(1,:),'rx');grid
subplot(222);plot(xQ(3,:));hold on;plot(xQ(3,:),'rx');grid
subplot(223);plot(xQ(4,:));hold on;plot(xQ(4,:),'rx');grid
subplot(224);plot(xQ(7,:));hold on;plot(xQ(7,:),'rx');grid

% Calcul de l'erreur de quantification
xM=zeros(length(Nbit),N);
for ii=1:length(Nbit)
    xM(ii,:)=x;
end
b=xM-xQ;

% Affichage des histogrammes des erreurs
figure(2)
subplot(221);hist(b(1,:),10);grid
subplot(222);hist(b(2,:),10);grid
subplot(223);hist(b(3,:),10);grid
subplot(224);hist(b(4,:),10);grid

% Calcul du RSB
for ii=1:length(Nbit)
    % F3(ii)=max(xQ(ii,:))/sqrt(xQ(ii,:)*xQ(ii,:)'/length(xQ(ii,:))); % Facteur
de forme à partir du signal quantifié
    Pb(ii)=b(ii,:)*b(ii,:)'/length(b(ii,:));
    Pb2(ii)=q(ii).^2/12;     % Remarque : 2° calcul de la puissance du bruit
    Pb3(ii)=var(b(ii,:));    % Remarque : 3° calcul de la puissance du bruit
    RSBexp(ii)=10*log10(Ps/Pb(ii));
    Fcarre=F2^2;
    RSBthe(ii)=6*ii+10*log10(3/Fcarre);
end

% Affichage du RSB
figure(3)
plot(RSBthe,'b')
hold on
plot(RSBthe,'bx')
plot(RSBexp,'r')
plot(RSBexp,'rx')
grid
```