TP de Traitement Numérique du Signal

Il est demandé de préparer ce TP

Première partie

I. Analyse et synthèse de filtres numériques

Nous allons étudier dans la première partie de ce TP, l'analyse et la synthèse de filtres numériques à réponses impulsionnelles finie et infinie à l'aide des logiciels `Matlab` et d'une applet JAVA développée à l'ENSSAT.

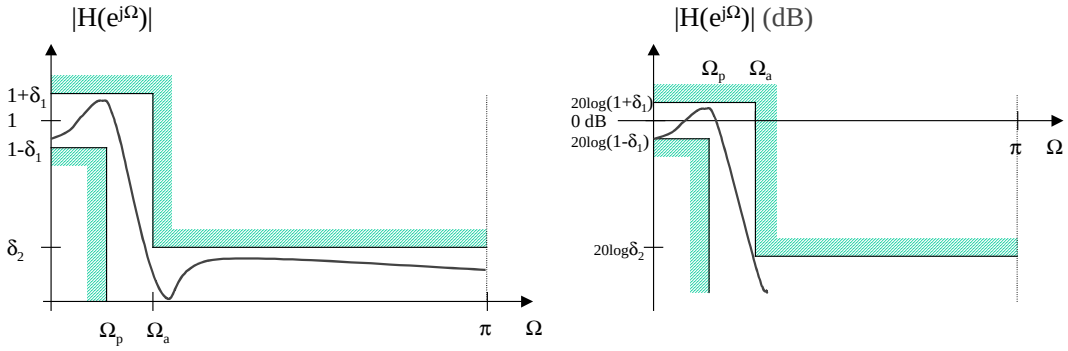
1 Analyse de filtres numériques passe-bas du second ordre

1.1 Analyse d'un filtre numérique RII passe-bas du second ordre

Le filtre numérique est spécifié par le gabarit en dB de la figure 1.(b) avec comme paramètres :

$$Fp = 1kHz, Fs = 3kHz, \Delta_1 = 20\log(1+\delta_1) = 1dB, \Delta_2 = 20\log(\delta_2) = -20dB, Fe = 10kHz$$

.



a) Gabarit fréquentiel linéaire

b) Gabarit fréquentiel en dB

FIG. 1 – Gabarit des filtres passe-bas à étudier

La synthèse par la méthode bilinéaire du filtre de Chebyshev $H(z)$ aboutit à la fonction :

$$H(z) = \frac{0.079(z+1)^2}{z^2 - 1.2z + 0.516}$$

Questions : filtre numérique RII en précision infinie

- Écrire le programme `Matlab` permettant de visualiser la réponse fréquentielle en module et en phase du filtre numérique (voir l'annexe A décrivant ce programme `Matlab`). Faites la relation entre l'échelle des abscisses de la commande `plot` de `Matlab` et les fréquences. Quelles sont donc les relations entre fréquence d'échantillonnage et fréquence de coupure ? Le filtre respecte-t-il le gabarit spécifié ?
- Visualisez la réponse impulsionnelle.
- Tracez dans le plan complexe les pôles et les zéros du filtre. Le filtre est-il stable ?

1.2 Analyse d'un filtre numérique RIF passe-bas du second ordre

Le gabarit du filtre passe-bas étant défini comme à la figure 1, la synthèse du filtre donne deux fonctions de transfert $H(z)$ selon le nombre de points considéré ($N = 11$ puis $N = 7$) :

$$H_{11}(z) = -0,0309396.(z^{-1}+z^{-9})-0,0390182.(z^{-2}+z^{-8})+0,0766059.(z^{-3}+z^{-7})+0,288307.(z^{-4}+z^{-6})+0,4.z^{-5}$$

$$H_7(z) = -0,0409365.(1+z^{-6})+0,078369.(z^{-1}+z^{-5})+0,289996.(z^{-2}+z^{-4})+0,4.z^{-3}$$

Questions : filtre numérique RIF en précision infinie
– Reprendre les questions du 1.1 (voir l'annexe B décrivant ce programme `Matlab`).

2 Synthèse d'un filtre passe-bas du second ordre

On désire réaliser un filtre numérique $H(z)$ équivalent à un filtre analogique de Butterworth passe-bas $H(j\omega)$ du deuxième ordre qui présente une fréquence de coupure Fc de $1KHz$. La fréquence d'échantillonnage Fe sera de $8kHz$. Son filtre de Butterworth normalisé est le suivant :

$$\begin{aligned} H_{Normalise}(j\omega_N) &= \frac{1}{1+\sqrt{2}j\omega_N-\omega_N^2} \\ H_{Normalise}(p_N) &= \frac{1}{1+\sqrt{2}p_N+p_N^2} \end{aligned}$$

2.1 Synthèse d'un filtre RII

Préparation
– Le gabarit d'un filtre passe-bas est défini comme indiqué sur la figure 1.(b). Calculer les paramètres $\Delta 1$ et $\Delta 2$ pour que le filtre passe bas de fréquence de coupure $Fc = 1KHz$ entre dans le gabarit, lorsque $Fp = 0.7Fc$ et $Fs = 3Fc$.
– Faire la synthèse par la méthode bilinéaire du filtre $H(j\omega)$ afin d'obtenir $H_b(z)$. On étudiera l'influence de la distorsion en fréquence impliquée par la méthode. On rappelle que la transformation bilinéaire est obtenue par $p = f(z) \triangleq 2Fe\frac{1-z^{-1}}{1+z^{-1}}$.
– Faire la synthèse par la méthode d'invariance impulsionnelle afin d'obtenir $H_i(z)$.
– Tracez dans le plan complexe les pôles et les zéros des filtres. Les filtres sont-il stables ?

Manipulations sur `Java_Filtre`
Un logiciel de synthèse de filtres numériques a été développé à l'ENSSAT. Il s'agit d'une applet JAVA que l'on peut lancer soit à l'aide de la commande `tns` sous un serveur SUN, soit en vous connectant à l'aide d'un navigateur à l'URL : `http ://lasti.enssat.fr/GroupeArchi/enseignements/Tns/Tns.php`.

Ce logiciel vous permet de choisir entre la synthèse de filtres RII (Butterworth ou Chebychev) selon la méthode bilinéaire, et la synthèse de filtres RIF par les méthodes de fenêtrage (Rectangle, Triangle, Hamming, Hanning, Blackman, Kaiser), d'échantillonnage fréquentiel, ou de Parks MacClellan. Une synthèse directe par les pôles et les zéros est possible. Après avoir entré les paramètres $\Delta 1, \Delta 2, Fp, Fs$ (cf gabarit de la figure 1.(b)), la synthèse s'effectue automatiquement et vous donne les valeurs des coefficients du filtre et la réponse impulsionnelle. Une aide est disponible à la même URL.
– Déterminez à l'aide de `Java_Filtre` les coefficients des filtres précédents.

Manipulations sur `Matlab`
– Écrire le programme `Matlab` permettant de faire la synthèse des filtres $H_b(z)$ et $H_i(z)$. L'annexe C est à compléter pour répondre à cette question.

– Visualiser la réponse fréquentielle en module et en phase des filtres numériques. Les filtres respectent-ils le gabarit spécifié ?
– Observer leurs réponses impulsionnelles.
– Comparer les deux méthodes dans les domaines fréquentiel et temporel.

2.2 Synthèse d'un filtre RIF

Préparation
– Soit le filtre passe bas idéal de fréquence de coupure Ωc. Déterminez la réponse impulsionnelle $h(n)$ de ce filtre. On prendra comme application numérique une fréquence de coupure $Fc = 1kHz$ pour une fréquence d'échantillonnage $Fe = 8kHz$ et un Δf de $2kHz$.
– Quel est la longueur théorique du filtre respectant ce gabarit ?

Le gabarit du filtre passe-bas est défini comme indiqué sur la figure du 1.(b). On utilisera les options suivantes de l'applet `Java_Filtre` qui correspondent à 3 méthodes différentes de synthèse :
– **Synthèse selon la méthode de fenêtrage.** Le gabarit idéal $H_i(\Omega)$ est échantillonné, et la réponse impulsionnelle correspondante $h_i(n)$ est calculée puis tronquée sur un nombre fini d'échantillons M. Cette réponse est ensuite multipliée par la fenêtre choisie. On obtient $h(n) = h_i(n).w(n)$. Le nombre de coefficients du filtre est donné par les estimations vues en cours.
– **Synthèse par la méthode de l'échantillonnage fréquentiel.**
– **Synthèse basée sur la méthode de Parks-MacClellan.** C'est une procédure itérative qui approxime la fonction de transfert par une somme de cosinus jusqu'à convergence. Le nombre de points de $h(n)$ est donné par :

$$M_{pm} = \frac{-10\log(\Delta 1\Delta 2) - 13}{14.6\delta fTe}$$

Les deux filtres obtenus sont à réponses impulsionnelles symétriques.

Manipulation sur `Java_Filtre`
– Observez et comparez les coefficients obtenus par les trois méthodes à partir de la spécification dans `Java_Filtre` correspondant au gabarit.
– Observez également le comportement de la synthèse pour d'autres types de fenêtres.

Manipulations sur `Matlab`
– Écrire le programme `Matlab` permettant de faire la synthèse du filtre RIF $H(z)$ par la méthode du fenêtrage pour des fenêtres rectangulaire et de Hamming. Le programme de l'annexe D est à compléter.
– Visualisez la réponse fréquentielle en module et en phase, observez la réponse impulsionnelle.
– Étudiez l'influence de la fenêtre sur le filtre numérique. Peut on mettre en évidence le phénomène de Gibbs.

– Écrire le programme `Matlab` permettant de faire la synthèse du filtre RIF $H(z)$ par la méthode de l'échantillonnage fréquentiel, en faisant attention à ce que le filtre soit à phase linéaire.
– Vérifiez que le filtre passe par les points fréquentiels spécifiés en module et en phase.
– Entrez les spécifications pour synthétiser un filtre passe bande puis un filtre dérivateur.

Deuxième partie

II. Algorithmes de traitement numérique du signal

3 Filtrage numérique en précision finie

3.1 Etude d'un filtre numérique RII passe-bas du second ordre en précision finie

L'étude suivante reprend l'exemple du filtre de la question 1.1 de la première partie.

- Les coefficients sont arrondis sur 7 puis 3 bits après la virgule. À l'aide du programme `Matlab` fourni en annexeE, étudier les modifications de la réponse fréquentielle du filtre quantifié, en particulier par rapport au gabarit. Quelles sont les problèmes apportés par la quantification sur la stabilité du filtre ? On placera un nombre suffisant de bits avant la virgule pour éviter les débordements.

  On se basera sur le diagramme des pôles et des zéros du filtre en vérifiant l'influence de la quantification sur les valeurs des coefficients du filtre puis sur les valeurs des pôles. Tracez le diagramme des pôles et des zéros avant après quantifications sur 8 et 4 bits.
- Étudiez les problèmes de débordements du filtre. A quelles conditions le filtre peut il diverger ?

3.2 Étude d'un filtre RII du quatrième ordre

Soit le filtre :

$$H(z) = 0.0599\frac{1 - 2.1832z^{-1} + 2.9976z^{-2} - 2.1832z^{-3} + z^{-4}}{1 - 3.1914z^{-1} + 4.17z^{-2} - 2.5854z^{-3} + 0.6443z^{-4}}$$

- Reprendre les questions précédentes afin de comparer l'implémentation de ce filtre en précision infinie et en précision finie. Que remarquez vous ?

La fonction de transfert peut être factorisée comme suit :

$$H(z) = \sqrt{0.0599}\frac{1 - 0.6511z^{-1} + z^{-2}}{1 - 1.5684z^{-1} + 0.6879z^{-2}}\sqrt{0.0599}\frac{1 - 1.5321z^{-1} + z^{-2}}{1 - 1.623z^{-1} + 0.9366z^{-2}}$$

- Cette factorisation affecte-t-elle la réponse fréquentielle du filtre en précision infinie ?
- Montrer que l'on peut stabiliser le filtre en précision finie en le mettant sous forme d'une cascade de cellules du second ordre. Qu'en concluez vous ?

3.3 Étude d'un filtre numérique RIF passe-bas en précision finie

- Reprendre les questions du 3.1 sur le filtre :

  $H_{11}(z) = -0.0309396(z^{-1}+z^{-9}) - 0.0390182(z^{-2}+z^{-8}) + 0.0766059(z^{-3}+z^{-7}) + 0.288307(z^{-4}+z^{-6}) + 0.4z^{-5}$

- On vérifiera particulièrement les points suivants : - caractéristique en phase du filtre ; - problèmes de débordement de l'accumulateur.

4 Convolution Discrète

Soit le filtre $h(n)$ défini à la figure 2 pour $M = 7$, nous allons étudier le filtrage d'un signal $x(n)$ de durée N limitée, défini par une convolution discrète

$$y(n) = \sum_{i=0}^{n} h(i)x(n-i), \qquad n = 0 \ldots N + M - 1,$$

au moyen de deux méthodes : une méthode temporelle directe puis une méthode fréquentielle rapide.

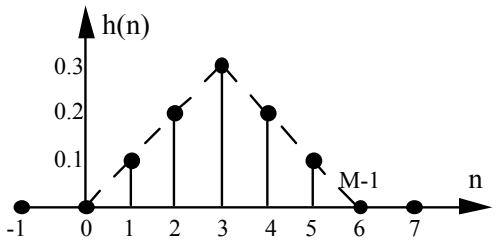


FIG. 2 – Signal $h(n)$

$x(n)$ est une impulsion rectangulaire de longueur $N = 11$.

- Écrire le programme `Matlab` réalisant la fonction de convolution dans le domaine temporelle à l'aide du programme de l'annexe F. La fonction `conv(x, h)` prend 2 vecteurs de longueur N et M et retourne un vecteur de longueur $N + M - 1$.
- Écrire le programme `Matlab` réalisant la fonction de convolution dans le domaine fréquentiel par l'utilisation de la fonction `FFT`.
- Vérifier que le résultat est identique dans les deux cas.
- Calculer précisément les complexités des deux approches. Comparer les complexités des méthodes fréquentielle et temporelle pour différentes valeurs de N et M. On pourra par exemple regarder les cas $N = 11$, $M = 7$; $N = M = 1024$; $N = 1024$, $M = 11$.

5 Analyse spectrale

5.1 Étude des fenêtres temporelles

- Étudiez les réponses temporelles et fréquentielles des fenêtres Rectangulaire, Bartlett, Hamming, Hanning, Blackman à l'aide du programme de l'annexe H.
- Retrouvez les caractéristiques du cours (largeur du lobe principal, atténuation du lobe secondaire) et étudiez l'influence du nombre de points N de la fenêtre.

5.2 Analyse spectrale d'un signal sinusoïdal

- Tracez la caractéristique idéale du spectre d'une sinusoïde échantillonnée. On prendra les valeurs suivantes : fréquence d'échantillonnage $Fe = 300$; fréquence de la sinusoïde $F_0 = 50$.

L'analyse spectrale d'un signal ne peut en pratique s'effectuer que sur un nombre limité de points. On appelle donc $T_0 = N.T$ l'horizon d'observation du signal. La multiplication temporelle du signal par une fenêtre de longueur T_0 implique une convolution dans le domaine des fréquences.

- Donner les finesses en fréquence et en amplitude pour les fenêtres rectangulaire et Hamming pour N = 32 et N = 256.
- Écrire le programme `Matlab` réalisant l'analyse spectrale de ce signal sinusoïdal dans le cas des fenêtres rectangulaire et Hamming pour N = 32 et N = 256.
- Que se passe-t-il lorsque la période de la sinusoïde est un multiple entier de l'horizon d'observation ($T = K.T_0 = K.N.T$).
- Écrire le programme `Matlab` réalisant l'analyse spectrale d'un signal formé de la somme de deux sinusoïdes de fréquence 50Hz et 60Hz dans le cas des fenêtres rectangulaire et Hamming pour N = 32 et N = 256.

6 Décimation et Interpolation

6.1 Décimation

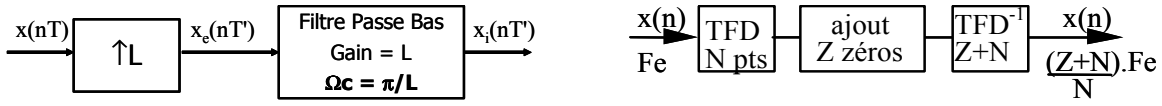
La décimation d'un signal consiste à diminuer sa fréquence d'échantillonnage. Ce signal doit être à bande étroite pour respecter le théorème d'échantillonnage. Ceci implique un filtrage passe bas avant l'opération de décimation.

```
N = 192;  n = 0:1:N-1;
x = sin(2*pi*0.02*n)+sin(2*pi*0.09*n);
```

- Étudiez les effets sur le spectre d'un signal d'une décimation par 2 puis par 4. On exprimera la décimation par M par : `y1 = x(1 :2 :N) ;`.
- Utilisez ensuite la fonction `decimate(x,M)` de `Matlab` qui effectue en plus un filtrage anti-repliement avant décimation. Comparez avec la version sans filtrage.
- Étudiez maintenant les effets sur le spectre d'un signal d'une décimation par 6. Expliquez le résultat obtenu.

6.2 Interpolation

L'interpolation d'un signal consiste à augmenter sa fréquence d'échantillonnage. Les deux principales méthodes sont l'insertion de $L-1$ zéros dans la réponse temporelle réalisant une interpolation par un facteur $\uparrow L$ (voir cours et figure 3.a), et l'ajout de Z zéros dans le domaine fréquentiel impliquant une interpolation par un facteur $L = \frac{N+Z}{N}$ où N est la taille de la FFT (voir figure 3.b).



(a) Interpolation par insertion de zéros et filtrage (b) Interpolation par TFD et ajout de zéros

FIG. 3 – Interpolation

- A partir du signal utilisé dans la section 6.1 décimé d'un facteur 4, réalisez l'insertion de zéros (élévateur de fréquence) et vérifiez son impact sur le spectre selon la méthode présentée en cours. On prendra un facteur d'interpolation $L = 4$.
- Réaliser l'interpolateur par insertion de zéros et filtrage RIF passe bas.
- Réaliser l'interpolateur par par TFD et ajout de zéros.
- Comparer qualitativement les deux méthodes d'interpolation.

6.3 Application au filtrage par décimation-interpolation

La réalisation de filtres numériques peut être simplifiée par une série de traitements interpolation-filtrage-décimation. L'exemple le plus connu est le filtre de lissage du Compact Disc qui réalise un filtrage très sélectif à $Fe/2$. Il est en pratique réalisé par le schéma ci dessous.

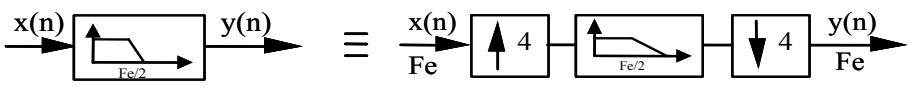


FIG. 4 – Filtrage par interpolation-décimation

Comparer les fonctions de transfert et les complexités des deux types de filtrage par la méthode directe et la méthode multicadence.

A Analyse d'un filtre numérique RII passe-bas du second ordre

close all ;

```
% Premiere partie
% 1 Analyse de filtres numériques passe-bas du second ordre
% 1.1 Analyse d'un filtre numérique RII passe-bas du second ordre

b = [0.079 2*0.079 0.079];          %Numérateur
a = [1 -1.2 0.516];                 %Dénominateur

N = 32; n=0:N-1;
delta = [1; zeros(N-1,1)];  %Impulsion
step = ones(N,1);           %Echelon unité
h = filter(b, a, delta);    %Réponse du filtre (ou impz(b,a);)
figure(1); stem(n,h);

L = 256;
[h,w] = freqz(b,a,L);
m = abs(h); p = angle(h);
figure(2); plot(20*log10(m)); title('Log Magnitude Filtre RII 2nd ordre (echelle Matlab non modifi
figure(3); plot(w(1:L-1),20*log10(m(1:L-1))); title('Log Magnitude Filtre RII 2nd ordre');
axis([0 pi -60 2]); grid
figure(4); plot(w,m); title('Magnitude'); figure(5); plot(w,p); title('Phase');

figure(6); zplane(b,a);
zero = roots(b); pole = roots(a);

Delta1 = 1; Delta2 = -20;
Fe = 10000;
Fp = 1000; NFp = round(L*Fp/(Fe/2));
Fs = 3000; NFs = round(L*Fs/(Fe/2));
gabh = [Delta1*ones(NFs,1); Delta2*ones(L-NFs,1)];
gabl = [-Delta1*ones(NFp,1); -5000*ones(L-NFp,1)];
figure(3); hold on; plot(w,gabh,'r'); plot(w,gabl,'r');
```

B Analyse d'un filtre numérique RIF passe-bas du second ordre

```
% 1.2 Analyse d'un filtre numérique RIF passe-bas du second ordre

b11 = [0 -0.0309396 -0.0390182 0.0766059 0.288307 0.4 0.288307 0.0766059 -0.0390182 -0.0309396 0];
b7 = [-0.0409365 0.078369 0.289996 0.4 0.289996 0.078369 -0.0409365];
[h11,w] = freqz(b11,1,L);
[h7,w] = freqz(b7,1,L);
figure; plot(w,20*log10(abs(h11)),'g'); title('Log Magnitude Filtre RIF 2nd ordre');
axis([0 pi -60 2]); hold on;
plot(w,20*log10(abs(h7)),'b'); grid
plot(w,gabh,'r'); plot(w,gabl,'r');
figure; plot(w,unwrap(angle(h11)),'g'); title('Phase Filtre RIF 2nd ordre');
grid hold on; plot(w,unwrap(angle(h7)),'b');

h11 = filter(b11, 1, delta);
h7 = filter(b7, 1, delta);
figure; subplot(2,1,1); stem(n,h11,'g'); subplot(2,1,2); stem(n,h7,'b')
figure; zplane(b11,1); figure; zplane(b7,1);
```

C Synthèse d'un filtre RII passe-bas

```
% 2 Synthèse d'un filtre passe-bas du second ordre
% 2.1 Synthèse d'un filtre RII
Fe = 8000; Fc = 1000; Fp = 700; Fs = 3000;

[N, Wn] = buttord(?,?,?,?);
disp(sprintf('Ordre du filtre de butterworth : %d \n',N));
[b a] = butter(?, ??,'s');         % Filtre analogique de Butterworth
[bb ab] = bilinear(b, a, Fe);      % Transformation bilinéaire sans prédistorsion
[bbp abp] = bilinear(b, a, Fe, Fc);% Transformation bilinéaire avec prédistorsion en Fc

[bi ai] = impinvar(b, a, Fe);      % Invariance impulsionnelle

L = 256; [hb,w] = freqz(bb,ab,L); figure;
plot(w,20*log10(abs(hb)),'g'); title('Log Magnitude Filtre RII Butterworth');
axis([0 pi -40 2]); grid

[hbp,w] = freqz(bbp,abp,L);
hold on; plot(w,20*log10(abs(hbp)),'b');

[hi,w] = freqz(bi,ai,L);
plot(w,20*log10(abs(hi)),'c');
```

D Synthèse d'un filtre RIF passe-bas

```
% 2.2 Synthèse d'un filtre RIF

% Réponse impulsionnelle théorique : h(n) = Wc/pi sinc(nWc/pi)
Fe = 8000; Fc = 1000; Wc = 2*pi*Fc/Fe;

% Méthode du fenetrage
N = 33;     % Longueur du filtre RIF, doit etre impaire pour la suite
alpha = (N-1)/2; n=0:N-1;
b = ?????; %RI du filtre idéal
bh = b.*hamming(N)';

[h,w] = freqz(b,1,L); figure; plot(w,20*log10(abs(h)),'b');
title('Log Magnitude Filtre RIF fenetrage'); axis([0 pi -60 2]); grid
[hh,w] = freqz(bh,1,L);
hold on; plot(w,20*log10(abs(hh)),'g');

figure; stem(n,b,'b'); hold on; stem(n,bh,'g'); grid

% Methode de l'echantillonnage frequentiel avec une TFD sur N points
N = 33;
Hef = [ones(???,1); zeros(???,1); ones(???,1)] .* expression_de_la_phase;
%Mettre eventuellement un terme de phase

figure; subplot(2,1,1); stem(n,abs(Hef));
subplot(2,1,2); stem(n,angle(Hef));

bef = ifft(Hef,N);
[hef,w] = freqz(bef,1,L);
```

```
figure; subplot(2,1,1);
plot(w,20*log10(abs(hef)),'b'); title('Log Magnitude Filtre RIF echantillonage frequentiel');
axis([0 pi -40 2]); subplot(2,1,2); plot(w,unwrap(angle(hef)),'b');
title('Phase Filtre RIF echantillonage frequentiel');

figure; stem(n,abs(bef),'b');  grid
```

E Filtrage numérique en précision finie

```
b = [0.079 2*0.079 0.079];          %Numérateur
a = [1 -1.2 0.516];                 %Dénominateur

L = 256;

[h,w] = freqz(b,a,L); figure; plot(w,20*log10(abs(h))); title('Log Magnitude Filtre RII 2nd ordre'
axis([0 pi -40 6]); grid
gabh = [Delta1*ones(NFs,1); Delta2*ones(L-NFs,1)];
gabl = [-Delta1*ones(NFp,1); -5000*ones(L-NFp,1)]; hold on;

% Arrondi sur b bits en Ca2 (1 bit de signe)
bit=8;                %Nombre de bits
q=2^(1-bit);
bQ8= ???;   %Quantification des coefficients
aQ8= ???;   %Quantification des coefficients
[hQ8,w] = freqz(bQ8,aQ8,L); plot(w,20*log10(abs(hQ8)),'g');

bit=4;                %Nombre de bits
q=2^(1-bit); bQ4=???; aQ4= ???; [hQ4,w] = freqz(bQ4,aQ4,L);
plot(w,20*log10(abs(hQ4)),'c');

plot(w,gabh,'r'); plot(w,gabl,'r'); legend('Précision infinie','8 bits','4 bits');

figure; zplane(b,a); zero = roots(b); pole = roots(a); figure;
zplane(bQ8,aQ8); figure; zplane(bQ4,aQ4);


% 3.2 Étude d'un filtre IIR du quatrième ordre

% Forme directe
%     a(1)*y(n) = b(1)*x(n) + b(2)*x(n-1) + ... + b(nb+1)*x(n-nb)
%                           - a(2)*y(n-1) - ... - a(na+1)*y(n-na)
b = 0.0599*[1 -2.1832 2.9976 -2.1832 1];
a = [1 -3.1914 4.17 -2.5854 0.6443];
......... A COMPLETER

% Forme cascade
b1 = sqrt(0.0599)*[1 -0.6511 1];     b2 = sqrt(0.0599)*[1 -1.5321 1];
a1 = [1 -1.5684 0.6879];     a2 = [1 -1.623 0.9366 ];
......... A COMPLETER

% 3.3 Filtre RIF en précision finie
b = [0 -0.0309396 -0.0390182 0.0766059 0.288307 0.4 0.288307 0.0766059 -0.0390182 -0.0309396 0];
......... A COMPLETER
```

F Convolution Discrete

```
close all;  %Ferme les figures en cours

% II.4 Convolution

N=11; M=7;
L=N+M-1; n=0:L-1;

h = [0; 0.1; 0.2; 0.3; 0.2; 0.1; 0; zeros(L-M,1)];
x = [ones(N,1); zeros(L-N,1)];
y1 = conv(x, h);
figure; stem(n,h,'r'); hold on; stem(n,x,'b'); stem(n,y1(1:L),'k');

..... A COMPLETER
```

G Analyse spectrale

```
% II.5 Analyse spectrale

N=64;
hr=boxcar(N);
figure; plot(hr,'y.'); hold on; grid on
ht=bartlett(N); plot(ht,'r:')
hn=hanning(N); plot(hn,'g--')
hm=hamming(N); plot(hm,'m-.')
hb=blackman(N); plot(hb,'b-')

xlabel('n');
ylabel('w(n)'); title('Reponses temporelles de différentes fenetres');
legend('Rectangulaire','Bartlett','Hanning','Hamming','Blackman')
AXIS([0 N 0 1.1])

figure
subplot(2,3,1)
grid on
HR = fft(hr,1024);
plot((0:510)/1024,20*log10(abs(HR(1:511)/HR(1))))
AXIS([0 0.5 -80 0]);
ylabel('W dB')
Title('Fenêtre rectangulaire')

subplot(2,3,2)
grid on
HT = fft(ht,1024);
plot((0:510)/1024,20*log10(abs(HT(1:511)/HT(1))))
AXIS([0 0.5 -80 0]);
ylabel('W dB')
Title('Fenêtre de Bartlett')

subplot(2,3,3)
grid on
HN = fft(hn,1024);
plot((0:510)/1024,20*log10(abs(HN(1:511)/HN(1))))
```

```
AXIS([0 0.5 -80 0]);
ylabel('W dB')
Title('Fenêtre de Hanning')

subplot(2,3,4)
grid on
HM = fft(hm,1024);
plot((0:510)/1024,20*log10(abs(HM(1:511)/HM(1))))
AXIS([0 0.5 -80 0]);
ylabel('W dB')
Title('Fenêtre de Hamming')

subplot(2,3,5)
grid on
HB = fft(hb,1024);
plot((0:510)/1024,20*log10(abs(HB(1:511)/HB(1))))
AXIS([0 0.5 -80 0]);
ylabel('W dB')
Title('Fenêtre de Blackman')

subplot(2,3,6)
grid on
HR = fft(hr,1024);
plot((0:510)/1024,abs(HR(1:511)/HR(1)))
AXIS([0 0.25 0 1]);
ylabel('|W| lineaire')
Title('Fenêtre rectangulaire')

% Analyse d'un sinus

N = 32;                 % nombre d'echantillons
n = 0 : N-1;            % vecteur des indices (nombres entiers)

a = 12;                 % amplitude de la sinusoïde
nu_o = 50;              % fréquence de la sinusoïde

nu_e = 300;             % fréquence d'échantillonnage
f_o = nu_o/nu_e;        % fréquence normalisée du signal
T_e = 1/nu_e;           % période d'échantillonnage
nT_e = n*T_e;           % vecteur des instants (en seconde)

y_n = cos(2*pi*f_o*n);

figure; subplot(3,1,1) ; stem(0:N-1+5,[y_n, zeros(1,5)]); axis([0 N-1+5 -1 1]);
title('Fonction cos(2pi f_o n) échantillonnée tronquée sur N=32 points'); ylabel('x_N(n)'); grid

fmin = -1/2; fmax = 1/2;
pas_f = (fmax-fmin)/1024 ; f=(fmin:pas_f:fmax-pas_f) ;
Y = fft(y_n,1024);
subplot(3,1,2) ; plot(f,abs(fftshift(Y)))
title('X_N(e^{j2pif})=TF\{x_N(n(\}') ; grid
Y = fft(y_n.*hamming(N)',1024);
hold on; plot(f,abs(fftshift(Y)),'r');
```

```
pas_f = (fmax-fmin)/N ; f=(fmin:pas_f:fmax-pas_f) ;
Y = fft(y_n);
subplot(3,1,3) ; stem(f,abs(fftshift(Y)))
title('X_N(k)=TFD\{x_N(n)\}') ; grid
Y = fft(y_n.*hamming(N)');
hold on; stem(f,abs(fftshift(Y)),'r')

..... A COMPLETER
```

H Décimation et Interpolation

```
% II.6 Décimation et interpolation
% 6.1 Décimation

N = 192;
n = 0:1:N-1;
x = sin(2*pi*0.02*n)+sin(2*pi*0.09*n);

figure; subplot(4,1,1); stem(x); grid
subplot(4,1,2); plot(x)

y1 = x(1:2:N); subplot(4,1,2); stem(y1); grid
y2 = x(1:4:N); subplot(4,1,3); stem(y2); grid
y3 = x(1:6:N); subplot(4,1,4); stem(y3); grid

y1 = decimate(x,2); subplot(4,1,3); stem(y1)
y2 = decimate(x,4); subplot(4,1,4); stem(y2)

X = fft(x,300);
Y1 = fft(y1,300);
Y2 = fft(y2,300);
Y3 = fft(y3,300);

figure; subplot(4,1,1); plot(abs(X)); grid
subplot(4,1,2); plot(abs(Y1)); grid
subplot(4,1,3); plot(abs(Y2)); grid
subplot(4,1,4); plot(abs(Y3)); grid

% 6.2 Interpolation

L=4;
x = y2; %sinus décimé par 4
y = zeros(N, 1);
y(1:L:N) = x;
figure; stem(y); grid

X = fft(x,300);
Y = fft(y,300);
figure; subplot(2,1,1); plot(abs(X)); grid
subplot(2,1,2); plot(abs(Y)); grid

Nrif = 33;     % Longueur du filtre RIF, doit etre impair pour la suite
...... A COMPLETER
y1 = filter(.....);
```

```
figure; plot(y1); grid

Y1 = fft(y1,300);
figure; plot(abs(Y1)); grid


% Interpolation par TFD
..... A COMPLETER

% 6.3 Filtrage
%filtre sans int/dec
Nrif = 21;     % Longueur du filtre RIF, doit etre impair pour la suite
Wc = pi/2;
b = Wc/pi*sinc(Wc/pi*(-(Nrif-1)/2:(Nrif-1)/2));
bh = b.*hamming(Nrif)';

N=256;
[h,t]=impz(bh,1,N);
H=fft(h,N);
figure; plot(abs(H)); grid

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%filtre équivalent avec int/dec

%%%% Interpolation
.......

%%%%  Filtre
.......

%%%% Decimation
......
```