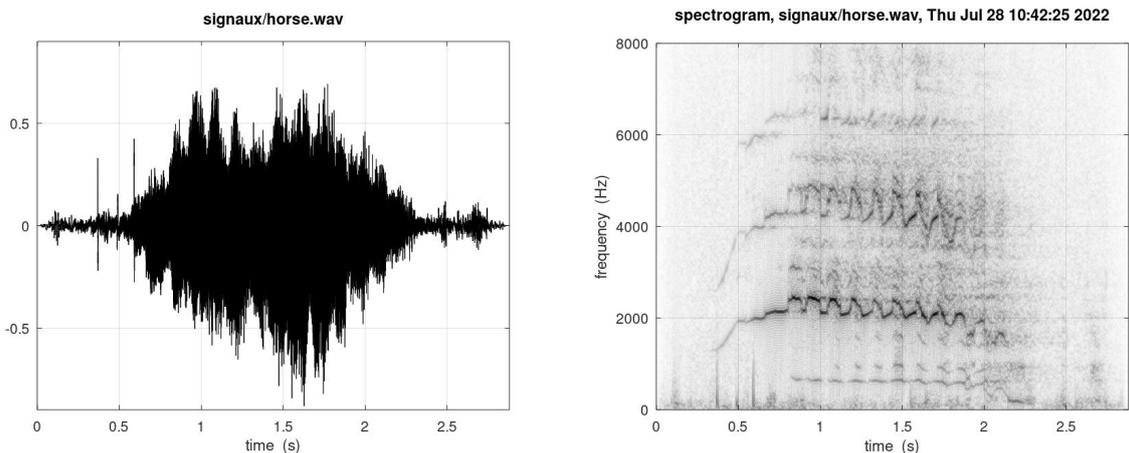


Nantes Université  
IUT de Saint-Nazaire  
Département Mesures Physiques  
Année 2022-2023

Travaux Pratiques d'Initiation  
au Traitement des Signaux de Mesure

F. Auger, 12 août 2022



Ressource	R3.04 : “Outils Mathématiques et Traitement du Signal”
Objectifs	Comprendre les bases mathématiques et les techniques de traitement du signal
Programme	<p><b>Représentation mathématique des signaux</b> : rappel des grandeurs associées aux signaux (valeur instantanée/valeur moyenne/valeur efficace), les différents types de signaux (continu/variable/périodique/harmonique/transitoire/aléatoire/quelconque).</p> <p><b>Décomposition en Série de Fourier (réelle et complexe)</b> : notion de fondamentale et d’harmonique, calcul des coefficients de Fourier, lien entre les coefficients de la décomposition en Série de Fourier réelle et ceux de la décomposition en Série de Fourier complexe.</p> <p><b>Transformée de Fourier et ses propriétés usuelles</b> : linéarité, décalage temps/fréquence, dérivation, dilatation en temps et en fréquence, conjugaison complexe, convolution, lien entre translation et convolution. Transformée de Fourier Inverse.</p> <p><b>Représentation de Fourier des signaux d’énergie infinie</b> : impulsion de Dirac, échelon de Heaviside, spectre des signaux périodiques, peigne de Dirac..., lien entre les coefficients issus des séries de Fourier et le spectre d’un signal. Représentations temporelles et spectrales : précaution à prendre lors de la lecture d’un spectre, notion de fréquence d’échantillonnage ...</p>
Mise en pratique	L’utilisation de logiciels pour le traitement du signal, d’analyseurs de spectre est recommandée. Les notions de filtrage, de modulation pourront être abordées en TP. L’illustration peut se faire en lien avec les ressources en acoustique, électronique, optique ... Illustration du théorème de Shannon-Nyquist.

Dans vos comptes-rendus de TP, vous aurez souvent l’occasion d’écrire des équations mathématiques. Il est donc très vivement conseillé de faire un compte-rendu **manuscrit** plutôt que dactylographié.

Ce document est mis à disposition selon les termes de la [Licence Creative Commons Attribution - Pas d’Utilisation Commerciale 4.0 International](https://creativecommons.org/licenses/by/4.0/). Les codes et les signaux sont disponibles [ici](#).

# Table des Matières

<b>1</b>	<b>Présentation du logiciel Octave</b>	<b>3</b>
1.1	L'essentiel de la programmation sous Octave . . . . .	3
1.2	Chargement de signaux de mesure dans Octave . . . . .	5
1.3	Exercice sur le chargement de signaux . . . . .	6
<b>2</b>	<b>Analyse spectrale</b>	<b>7</b>
2.1	L'analyse de Fourier sous Octave . . . . .	7
2.2	Exercice sur l'analyse de Fourier . . . . .	8
2.3	L'analyse spectrale sous Octave . . . . .	8
	2.3.1 Fonction d'analyse spectrale . . . . .	8
	2.3.2 Utilisation dans le cas d'un spectre de raies . . . . .	9
	2.3.3 Utilisation dans le cas d'un spectre continu . . . . .	10
2.4	Exercices sur l'analyse spectrale . . . . .	10
<b>3</b>	<b>Filtrage</b>	<b>11</b>
<b>4</b>	<b>Échantillonnage</b>	<b>14</b>
4.1	Représentation temps-fréquence de signaux non-stationnaires . . . . .	14
4.2	Exercices sur l'échantillonnage . . . . .	15

# Chapitre 1

## Présentation du logiciel Octave

### 1.1 L'essentiel de la programmation sous Octave

fichier Presentation\_Octave\_20220726.m

```
% Introduction a Octave
% F. Auger, juillet 2022
% N'hesitez pas a envoyer vos suggestions a francois.auger@univ-nantes.fr

% GNU Octave est un langage de haut niveau destine au calcul scientifique
% C'est un environnement libre et multiplateforme depuis 1988.
% La version 7.1.0 est disponible depuis le 06/04/2022 sur https://octave.org

% Dans un script, tout ce qui commence par un % est un commentaire
%-----
% clc efface la fenetre de commande
clc

%-----
% Octave peut etre utilise comme une calculatrice.
% La touche ENTREE lance le calcul
2+3

% On peut utiliser des variables sans avoir a les declarer
% et faire des affectations avec le signe =
z=2; z+3
% Le point virgule est le separateur d'instructions
% Octave n'affiche alors pas le resultat dans la fenetre de commande

% On peut faire du calcul scientifique avec Octave
t=1.35; x= 6.7*exp(-0.3*t)*sin(2*pi*120*t)

% pour obtenir de l'aide, il suffit de taper 'help'
help exp

%-----
% On peut facilement definir des vecteurs
i=1:11
j=1:2:11
x=linspace(0,3,11)
```

```

x(1) % Le premier indice est 1

% et faire du calcul sur tous les elements du vecteur en une fois
% avec des operateurs ponctuels .*, ./
exp(x)./i

% On peut aussi definir des matrices et faire du calcul matriciel
A=[1 2 3; 2 5 6; 3 6 9]
X=[1;5;9]
A*X
X'*A*X

%-----
% On peut facilement faire des graphiques
t=(0:1500)/1000; x= 6.7.*exp(-2.3*t).*cos(2*pi*20*t);

figure(1);
plot(t,x); grid; xlim([0 1.5]); ylim([-6.8 6.8]);
xlabel('temps (s)'); ylabel('tension (V)');
title('signal transitoire');
print('Graphique.jpg','-f1', '-djpg');

%-----
% On peut utiliser des structures conditionnelles
if (z>1)
    printf('z est plus grand que 1\n');
end

if (z>2)
    printf('z est plus grand que 2\n');
else
    printf('z est inferieur ou egal a 2\n');
end

% On peut utiliser des structures de repetition
printf('for : \n');
for i=1:5
    printf('i=%d\n',i);
end

printf('while : \n');
i=1;
while (i<=5)
    printf('i=%d\n',i);
    i=i+1;
end

printf('do until : \n');
i=1;
do
    printf('i=%d\n',i);
    i=i+1;
until (i>=6);

```

```
%-----
% Les fonctions simples peuvent etre definies facilement
sindeg = @(x) sin(x*pi/180);
sindeg(90)

% On peut aussi definir des fonctions dans des fichiers a part
cosdeg(90)
```

## 1.2 Chargement de signaux de mesure dans Octave

fichier Chargement\_Signaux\_Experimentaux\_20220726.m

```
% Chargement de signaux dans octave
% F. Auger, juillet 2022
% N'hesitez pas a envoyer vos suggestions a francois.auger@univ-nantes.fr

% Display information about all supported audio formats.
% voir https://docs.octave.org/v4.4.1/Audio-File-Utilities.html
audioformats()

%-----
% fichier wav
filename='signaux/horse.wav';
info = audioinfo(filename)
[Sig, Fs] = audioread (filename);
Ts=1.0/Fs; TotalSamples=length(Sig); t=(0:TotalSamples-1)*Ts;

figure(1);
plot(t,Sig); grid; xlim([0 TotalSamples*Ts]); ylim([-0.9 0.9]);
xlabel('temps (s)');
title(filename);

% ecoute du signal
% player = audioplayer(Sig, Fs); play (player);

%-----
% fichier mp3
% pour lire des fichiers mp3, convertir d'abord au format wav avec audacity
filename='signaux/vagues.wav';
info = audioinfo(filename)
[Sig, Fs] = audioread (filename);
Ts=1.0/Fs; TotalSamples=length(Sig); t=(0:TotalSamples-1)*Ts;

figure(2);
% signal stereo, ':,1' = toutes les lignes de la 1ere colonne
subplot(211);
plot(t,Sig(:,1)); grid; xlim([0 TotalSamples*Ts]); ylim([-0.45 0.45]);
title(filename);
% signal stereo, ':,2' = toutes les lignes de la 2eme colonne
subplot(212);
plot(t,Sig(:,2)); grid; xlim([0 TotalSamples*Ts]); ylim([-0.45 0.45]);
xlabel('temps (s)');
```

```

% ecoute du signal
% player = audioplayer(Sig, Fs); play (player);

%-----
% fichier au
filename='signaux/start.au';
info = audioinfo(filename)
[Sig, Fs] = audioread (filename);
Ts=1.0/Fs; TotalSamples=length(Sig); t=(0:TotalSamples-1)*Ts;

figure(3);
plot(t,Sig); grid; % xlim([0 1.5]); ylim([-6.8 6.8]);
title(filename);
xlabel('temps (s)');

% ecoute du signal
% player = audioplayer(Sig, Fs); play (player);

%-----
% fichier texte
filename='signaux/tremblement_terre_Nice_20010225.txt';
load(filename);
t =tremblement_terre_Nice_20010225(:,1);
Sig=tremblement_terre_Nice_20010225(:,2);
figure(4);
plot(t,Sig); grid; % xlim([0 1.5]); ylim([-6.8 6.8]);
title('tremblement terre Nice 20010225');
xlabel('temps (s)');

```

### 1.3 Exercice sur le chargement de signaux

**Exercice I.1:** Représenter graphiquement le signal contenu dans le fichier `Raven.wav`.

À l'aide des fonctions `min`, `max`, `mean` et `std`, déterminer la valeur minimale, la valeur maximale, la moyenne et la valeur efficace  $x_{\text{eff}}$  de ce signal.

Question subsidiaire : à l'aide des fonctions `sort` et `diff`, déterminer le pas de quantification  $q$  de ce signal. Vérifier qu'il est égal à 2/65536. En déduire le nombre de valeurs utilisées à la sortie du convertisseur analogique-numérique, et le rapport signal sur bruit de quantification, égal à

$$\text{RSB}_q^{\text{dB}} = 10 \log_{10} \left( \frac{12 x_{\text{eff}}^2}{q^2} \right)$$

Un signal est considéré comme correctement acquis si son  $\text{RSB}_q^{\text{dB}}$  est supérieur à 60 dB. Est-ce le cas pour ce signal ?

# Chapitre 2

## Analyse spectrale

### 2.1 L'analyse de Fourier sous Octave

fichier Introduction\_Fourier\_20220726.m

```
% Introduction a l'analyse de Fourier
% F. Auger, juillet 2022
% N'hésitez pas a envoyer vos suggestions a francois.auger@univ-nantes.fr

% Le nombre de points du signal
NbPoints=40;

% Frequence d'echantillonnage et periode d'echantillonnage
Fe=100.0; Te=1/Fe;

% instants ou le signal va etre calcule
t=(0:NbPoints-1)*Te;

f1=12; x1=2.5 * cos(2*pi*f1*t);
f2=24; x2=2.0 * cos(2*pi*f2*t);
x=x1+x2;

figure(1);
plot(t,x); grid; xlim([0 NbPoints*Te]); ylim([-5 5]);
xlabel('temps (s)'); ylabel('tension (V)'); title('signal');

% NbFreqs est le nombre de frequences. Il doit etre pair
NbFreqs=6000;

% Calcul de la representation frequentielle de x
X=fft(x, NbFreqs);

% axe des frequences : de -Fe/2 a +Fe/2 (environ)
Freqs=(-NbFreqs/2:(NbFreqs/2-1))*Fe/NbFreqs;

figure(2);
plot(Freqs,fftshift(abs(X))*2/NbPoints);
grid; xlim([-Fe/2 Fe/2]); ylim([0 3.0]);
xlabel('frequence (Hz)'); ylabel('amplitude (V)');
title('Représentation fréquentielle');
```

```

f3=32; x3=1.0 * cos(2*pi*f3*t);
bruit=0.0*randn(1,NbPoints);
Sig=x1+x2+x3+bruit;

figure(3);
plot(t,Sig); grid; % xlim([0 NbPoints*Te]); ylim([-5 7]);
xlabel('temps (s)'); ylabel('tension (V)'); title('signal');

Fourier_Sig=fft(Sig, NbFreqs);

figure(4);
plot(Freqs,fftshift(abs(Fourier_Sig))*2/NbPoints);
grid; xlim([-Fe/2 Fe/2]); ylim([0 3.0]);
xlabel('frequence (Hz)'); ylabel('amplitude (V)');
title('Representation frequentielle');

% Utilisation d'une fenetre de ponderation
theta=linspace(-pi, pi, NbPoints);
h=0.5+0.5*cos(theta); h=h/sum(h); % Hann

X2=fft(h.*x, NbFreqs);

figure(5);
plot(Freqs,fftshift(abs(X2))*2);
grid; xlim([-Fe/2 Fe/2]); ylim([0 3.0]);
xlabel('frequence (Hz)'); ylabel('amplitude (V)');
title('Representation frequentielle');

Fourier_Sig2=fft(h.*Sig, NbFreqs);

figure(6);
plot(Freqs,fftshift(abs(Fourier_Sig2))*2);
grid; xlim([-Fe/2 Fe/2]); ylim([0 3.0]);
xlabel('frequence (Hz)'); ylabel('amplitude (V)');
title('Representation frequentielle');

```

## 2.2 Exercice sur l'analyse de Fourier

**Exercice II.1:** On souhaite analyser un signal échantillonné à 1 kHz comprenant deux sinusoides de fréquences 100 et 150 Hz et d'amplitudes 1 et 100.

En utilisant une fenêtre de pondération rectangulaire, quelle est la plus petite durée d'enregistrement qui permet d'avoir une exactitude sur la mesure des fréquences et des amplitudes meilleure que 1 % ?

Même question en utilisant une fenêtre de pondération de Hann.

## 2.3 L'analyse spectrale sous Octave

### 2.3.1 Fonction d'analyse spectrale

fichier anaspec.m

```

function [Pxx,Freqs,k,factor]=anaspec(signal,Fe,Nframe,Nfft,TimeStep);
% [Pxx, Freqs, k, factor] = anaspec(signal,Fe,Nframe,Nfft,TimeStep)
% calcule la densite spectrale de puissance d'un signal
%
% Dans le cas d'un spectre de raies, il faut diviser Pxx par facteur
% signal      : signal analyse
% Fe          : frequence d'echantillonnage
% Nframe      : nombre de points du morceau de signal
% Nfft        : nombre de frequences
% TimeStep    : pas de temps entre deux morceaux
% Pxx         : densite spectrale de puissance
% Freqs       : vecteur des frequences
% k           : nombre de moyennes
% factor      : facteur par lequel diviser Pxx en cas de spectre de raies

% F. Auger, 26/07/2022

Npoints=length(signal);
n=0:(Nframe-1);

% h=ones(1,Nframe); % fenetre rectangulaire
h=0.42-0.50*cos(2*pi*n/Nframe)+0.08*cos(4*pi*n/Nframe); % Blackman window

factor=sum(h); h=h/sqrt(factor);

% figure(3); plot(n,h); grid; % juste pour verifier

Xfm=zeros(1,Nfft);
k=0;
while (k*TimeStep+Nframe<=Npoints)
    Xfm=Xfm+2*abs(fft(h.*signal(n+1+k*TimeStep),Nfft)).^2;
    k=k+1;
end

Xfm=Xfm/k;

Indices=1:Nfft/2+1;
Pxx=Xfm(Indices);
Freqs=(Indices-1)*Fe/Nfft;

```

### 2.3.2 Utilisation dans le cas d'un spectre de raies

fichier AnalyseurDeSpectre.m

```

% introduction a l'analyse spectrale : cas d'un spectre de raies
% F. Auger, juillet 2022

% le signal est une somme bruitee de sinusoides
NbPoints=6000; Fe=100; Te=1/Fe; t=(0:NbPoints-1)*Te;
f1=12; x1= 2.0 * cos(2*pi*f1*t);
f2=24; x2= 3.0 * cos(2*pi*f2*t);
f3=32; x3= 1.0 * cos(2*pi*f3*t);
f4=6;  x4= 3.0 * cos(2*pi*f4*t);
bruit=3*randn(1,NbPoints);

```

```

Sig=x1+x2+x3+x4+bruit;

[Pxx, Freqs, NbMoy, facteur] = anaspec(Sig, Fe, 300, 2048, 250);
fprintf('moyenne de %i spectres\n', NbMoy);

% Le spectre de puissance est obtenu en divisant Pxx par facteur
figure(1); plot(Freqs,Pxx/facteur); grid;
xlabel('frequence (Hz)'); ylabel('puissance');
title('densite spectrale de puissance (spectre de raies)');

```

### 2.3.3 Utilisation dans le cas d'un spectre continu

fichier Analyse\_frogs2.m

```

% introduction a l'analyse spectrale : cas d'un spectre continu
% F. Auger, juillet 2022

```

```

%-----
% chargement fichier wav
filename='signaux/frogs2.wav';
info = audioinfo(filename)
[Sig, Fs] = audioread (filename); Sig=Sig-mean(Sig);
Ts=1.0/Fs; TotalSamples=length(Sig); t=(0:TotalSamples-1)*Ts;

figure(1);
plot(t,Sig); grid; xlim([0 TotalSamples*Ts]); ylim([-0.9 0.9]);
xlabel('temps (s)');
title(filename);

% ecoute du signal
% player = audioplayer(Sig, Fs); play (player);

% Sig est un vecteur ligne
% On prend sa transposee Sig' pour avoir un vecteur colonne
[Pxx, Freqs, NbMoy]=anaspec(Sig', Fs, 15000, 16384, 5000);
fprintf('moyenne de %i spectres\n', NbMoy);

figure(2); plot(Freqs,Pxx); grid;
xlabel('frequence (Hz)'); ylabel('puissance');
title(['DSP (distribution continue) de ' filename]);

```

## 2.4 Exercices sur l'analyse spectrale

**Exercice II.2:** À l'aide de la fonction `csvread` (faire `help csvread` pour obtenir de l'aide sur cette fonction), charger le fichier `HairDryer_MedV_LowT_Current_Voltage.csv`, qui contient la tension appliquée aux bornes d'un sèche-cheveux, et le courant qu'il consomme. La tension est divisée par 600 et le courant est divisé par 2. Faire une analyse spectrale du signal de courant pour mettre en évidence les harmoniques de la fréquence du secteur.

**Exercice II.3:** Faire une analyse spectrale du signal contenu dans le fichier `Raven.wav`.

# Chapitre 3

## Filtrage

**Exercice III.1:** Un micro-contrôleur commande un convertisseur analogique-numérique pour acquérir un signal  $x(t)$  à intervalles de temps réguliers  $T_e$ . On construit ainsi un signal discret  $x[n] = x(nT_e)$ . Le micro-contrôleur met également en œuvre un filtre (un système linéaire stationnaire) en calculant le signal discret  $y[n]$  à partir de  $x[n]$  par l'équation de récurrence

$$y[n] = \frac{1 - \beta}{1 + \beta} y[n - 1] + \frac{\beta}{1 + \beta} (x[n] + x[n - 1])$$

1. Calculer le gain statique et le gain de Nyquist de ce filtre. Interpréter les résultats obtenus.
2. Montrer que sa réponse fréquentielle est égale à

$$H_{lp}(\lambda) = \frac{\beta \cos(\pi\lambda)}{\beta \cos(\pi\lambda) + j \sin(\pi\lambda)} = \frac{1}{1 + j \frac{\tan(\pi\lambda)}{\beta}}$$

On pourra éventuellement utiliser pour cela les relations

$$1 + e^{-j2\pi\lambda} = 2 \cos(\pi\lambda) e^{-j\pi\lambda} \quad \text{et} \quad 1 - e^{-j2\pi\lambda} = 2j \sin(\pi\lambda) e^{-j\pi\lambda}$$

3. En déduire que la fréquence de coupure de ce filtre est égale à  $\lambda_c = f_c T_e$  si  $\beta = \tan(\pi\lambda_c)$ .
4. Le code ci-dessous (fichier LP1\_Bode\_20220728.m) calcule  $H_{lp}(\lambda)$  et trace sa représentation de Bode. Analyser ce code et retrouver sur cette représentation de Bode les principales caractéristiques du filtre.

```
% reponse frequentielle d'un filtre passe-bas discret du 1er ordre  
% F. Auger, juillet 2022
```

```
% calcul de la reponse frequentielle  
NbPoints=2000; lambda=linspace(0,0.5,NbPoints);  
lambdac=0.1; beta=tan(pi*lambdac);  
cosinus=cos(pi*lambda); sinus=sin(pi*lambda);  
H=beta*cosinus./(beta*cosinus + 1j*sinus);
```

```
% representation de Bode : en haut le module  
figure(1);  
ax1=subplot(211);  
plot(lambda,abs(H),lambdac,sqrt(2)/2,'*',0,1,'*',0.5,0,'*');  
grid; xlim([0 0.5]); ylim([0 1.1]);
```

```

title('representation de Bode du filtre : module (echelle lin)');

% representation de Bode : en bas la phase
ax2=subplot(212);
plot(lambda, arg(H)*180/pi, 0,0, '*', lambdac, -45, '*', 0.5, -90, '*');
grid; xlim([0 0.5]); ylim([-100 0]);
xlabel('frequence normalisee');
title('representation de Bode du filtre : phase (degres)');

linkaxes([ax1 ax2], 'x'); % meme axe x sur les deux graphiques

% print('Bode_LP1.jpg', '-f1', '-djpg');

```

5. Le code ci-dessous (fichier LP1\_Utilisation\_20220729.m) montre comment appliquer le filtre à différents signaux. Analyser les résultats obtenus, et retrouver les résultats de la réponse fréquentielle pour un signal d'entrée sinusoïdal.

```

% application d'un filtre passe-bas discret du 1er ordre a un signal
% F. Auger, juillet 2022

% dimension du signal d'entree
NbPoints=1000; Instants=0:NbPoints-1;

% sinusoide bruitée
lambda=0.01; Sige=5*cos(2*pi*lambda*Instants)+0.2*randn(1, NbPoints);

% impulsion. On pourra retrouver le gain statique avec sum(Sige)
% et le gain en variance avec norm(Sigs)^2
% Sige=zeros(1, NbPoints); Sige(1)=1.0;

% signal rectangulaire
% Periode=200; DNH=80; A=5;
% Sige=A*(rem(Instants, Periode)>=DNH)-A*(rem(Instants, Periode)<DNH);

Sigs=zeros(1, NbPoints); % declaration du signal de sortie

% parametres et coefficients du filtre
lambdac=0.1; beta=tan(pi*lambdac);
coeff1=(1-beta)/(1+beta); coeff2=beta/(1+beta);

xpred=0.0; ypred=0.0; % valeurs initiales des deux suites

% algorithme de filtrage
for i=1:NbPoints
    x = Sige(i); y = coeff1 * ypred + coeff2 * (x + xpred);
    Sigs(i) = y;

    xpred = x; ypred = y; % preparation de la prochaine iteration
end

% representation des deux signaux sur deux graphiques separes
figure(1);
ax1=subplot(211); plot(Instants, Sige); grid;
xlim([1 NbPoints]); ylim([-6 +6]); title('signal d'entree');

```

```

ax2=subplot(212); plot(Instants, Sigs); grid;
xlim([1 NbPoints]); ylim([-6 +6]); title('signal de sortie');
xlabel('indices temporels');
linkaxes([ax1 ax2], 'x'); % meme axe x sur les deux graphiques
% print('LP1_utilisation.jpg', '-f1', '-djpg');

% representation des deux signaux sur le meme graphique
figure(2);
plot(Instants, Sige, Instants, Sigs); legend('entree', 'sortie');
grid; xlim([1 NbPoints]); ylim([-6 +6]);
xlabel('indices temporels'); title('signaux d''entree et de sortie');

```

6. Calculer à présent le gain statique, le gain de Nyquist et la réponse fréquentielle  $H_{hp}(\lambda)$  d'un second filtre défini par l'équation de récurrence

$$y_2[n] = \frac{1 - \beta}{1 + \beta} y_2[n - 1] + \frac{1}{1 + \beta} (x[n] - x[n - 1])$$

avec ici aussi  $\beta = \tan(\pi\lambda_c)$ . Rajouter sur la représentation de Bode de la question précédente la représentation de Bode de ce second filtre. Calculer  $H_{lp}(\lambda) + H_{hp}(\lambda)$  et  $|H_{lp}(\lambda)|^2 + |H_{hp}(\lambda)|^2$ . Que peut-on en déduire ? Appliquer ce second filtre aux mêmes signaux que précédemment et conclure.

**Exercice III.2:** On s'intéresse à un système linéaire stationnaire causal à temps discret qui, à partir d'un signal échantillonné  $x[n]$ , construit un nouveau signal  $y[n]$  par la relation

$$y[n] = K (x[n] - 2\alpha x[n - 1] + 2\alpha x[n - 3] - x[n - 4])$$

1. Calculer le gain statique et le gain de Nyquist de ce filtre. Interpréter les résultats obtenus.
2. Calculer les premiers termes de la réponse de ce filtre à une impulsion localisée à l'instant zéro. Ce filtre est-il à réponse impulsionnelle finie ou infinie ? Cette réponse impulsionnelle vérifie-t-elle une propriété de symétrie ?
3. Calculer la réponse fréquentielle  $H(\lambda)$  de ce filtre, c'est à dire le rapport entre le signal de sortie et le signal d'entrée lorsque ces deux signaux sont de la forme  $x[n] = X z^n$  et  $y[n] = Y z^n$ , avec  $z = e^{j2\pi\lambda}$ .
4. En utilisant des relations de symétrisation du type  $1 + x^{-2} = x^{-1}(x + x^{-1})$ , en déduire que cette réponse fréquentielle peut s'écrire sous la forme

$$H(\lambda) = (1 - z^{-2})(1 - 2\alpha z^{-1} + z^{-2}) = jA(\lambda) e^{-j4\pi\lambda}$$

avec  $A(\lambda) = 4K \sin(2\pi\lambda) (\cos(2\pi\lambda) - \alpha)$

5. Pour calculer  $\alpha$  et  $K$ , on impose que  $A(\lambda_0) = 1$  et  $\frac{dA}{d\lambda}(\lambda_0) = 0$ . On obtient alors un filtre passe-bande de fréquence centrale  $\lambda_0$ . En déduire les valeurs de  $\alpha$  et  $K$ .
6. Écrire un programme qui calcule et trace la représentation de Bode de ce filtre. Compléter le programme pour appliquer ce filtre à un signal d'entrée sinusoïdal, et retrouver les résultats de la réponse fréquentielle.

# Chapitre 4

## Échantillonnage

### 4.1 Représentation temps-fréquence de signaux non-stationnaire

fichier demo\_tfrsp\_20220727.m

```
Fe=1000; Te=1/Fe;
```

```
NbPoints=1024; t=(0:NbPoints-1)*Te;
```

```
fmin=50; fmax=450; alpha=2*pi*(fmax-fmin)/((NbPoints-1)*Te);
```

```
phi=2*pi*fmin*t+alpha*t.^2/2;
```

```
x=0.9*cos(phi);
```

```
% player = audioplayer(x, Fe); play (player);
```

```
figure(1);
```

```
plot(t,x); grid; xlim([0 NbPoints*Te]); % ylim([-0.9 0.9]);
```

```
xlabel('temps (s)');
```

```
title('signal analyse');
```

```
% Pour utiliser la fonction kaiser, charger la boite a outils signal
```

```
% voir https://octave.sourceforge.io/signal/overview.html
```

```
pkg load signal
```

```
% calcul du spectrogramme
```

```
NbFreqs=1024;
```

```
tfr = tfrsp(x',1:NbPoints, NbFreqs, kaiser(71), 1);
```

```
figure(2);
```

```
colormap(flipud(gray));
```

```
imagesc(t, linspace(0,0.5,NbFreqs/2)*Fe,tfr(1:NbFreqs/2,:)); grid;
```

```
set (gca, "ydir", "normal"); % put the 'y' axis in the correct direction
```

```
xlabel('temps (s)'); ylabel('frequence (Hz)');
```

```
title(['spectrogramme, ' ctime(time())]);
```

```
filename='signaux/horse.wav';
```

```
[Sig, Fs] = audioread (filename); Sig=Sig-mean(Sig);
```

```
Ts=1.0/Fs; TotalSamples=length(Sig); t=(0:TotalSamples-1)*Ts;
```

```
% player = audioplayer(Sig, Fs); play (player);
```

```

figure(3);
plot(t, Sig, 'k'); grid; xlim([0 TotalSamples*Ts]); ylim([-0.9 0.9]);
xlabel('time (s)'); title(filename);
print('horse_signal.jpg','-f3', '-djpg');

NbFreqs=4096;
tfr = tfrsp(Sig,1:50:TotalSamples, NbFreqs, kaiser(401), 1);

figure(4);
colormap(flipud(gray));
imagesc(t, linspace(0,0.5,NbFreqs/2)*Fs,tfr(1:NbFreqs/2,:).^0.2); grid;
set(gca, "ydir", "normal"); % put the 'y' axis in the correct direction
ylim([0 8000]);
xlabel('time (s)'); ylabel('frequency (Hz)');
title(['spectrogram, ' filename ', ' ctime(ctime())]);
print('horse_sp.jpg','-f4', '-djpg');

```

## 4.2 Exercices sur l'échantillonnage

**Exercice IV.1:** Le programme ci-dessus génère un enregistrement de 1024 échantillons d'une sinusoïde dont la fréquence varie de  $f_i = 50$  Hz à  $f_f = 450$  Hz.

Le signal sinusoïdal est de la forme  $x(t) = 0.9 \cos(\varphi(t))$ , avec  $\varphi(t) = \omega_i t + \alpha t^2/2$ . Calculer  $\Omega(t) = \frac{d\varphi}{dt}(t)$  et  $\frac{\Omega(t)}{2\pi}$ . Comment varie la fréquence en fonction du temps ? Lancer le programme avec une fréquence d'échantillonnage  $F_e = 1$  kHz et écouter le résultat.

Relancer ensuite ce programme après avoir fait passer la fréquence d'échantillonnage à 400 Hz. Calculer et analyser le spectrogramme du signal obtenu. Que s'est-il passé ? Faire le lien avec le théorème de Shannon.

**Exercice IV.2:** Utiliser le programme de l'exercice 1 pour générer un signal sinusoïdal d'une durée de 20 secondes, dont la fréquence augmente linéairement de 400 à 8000 Hz et échantillonné avec une période d'échantillonnage de 170  $\mu$ s. Écouter (au casque) et faire un spectrogramme du signal. Analyser et expliquer le résultat obtenu.

# Fiche d'appréciation

N'hésitez pas à donner votre avis sur ce document, et suggérer des modifications pour améliorer la qualité de cet outil de formation. Vous êtes invité pour cela à utiliser cette page, en y écrivant votre nom et votre commentaire, puis à la remettre dans mon casier au secrétariat du département Mesures Physiques.

Votre nom :

Votre commentaire :