Notes Matlab - Son

Table des matières

SON

Dualité temps/fréquences

> **Pour rappel, le dualité temps/fréquence désigne le fait que l'on ne peut pas être précis dans les deux domaines en même temps.**
> **Si nous voulons une précision temporelle, nous aurons une imprécision fréquentielle (étalement de spectre) et si nous voulons une précision fréquentielle, nous aurons une imprécision temporelle (longue durée).**

1. Sinus de 440Hz

```
f=440; Fe=8000; T=2;
t=0: 1/Fe: T;
s = sin(2*pi*f*t);
% sound(sin,Fe); Pour écouter le signal.
% FFT - Schema du spectre
f = -Fe/2: 1/T: Fe/2;
y = abs(fftshift(fft(s)));
plot(f,y)
```

2. Sinus de 440Hz, amplitude 1 plus sinus de 880Hz d'amplitude 0,5

```
Fe=8000; T=2;
t=0: 1/Fe: T;
% Sinus 1
f1=440; amp1=1;
sin1 = amp1*sin(2*pi*f1*t);
% Sinus 2
f2=880; amp2=0.5;
sin2 = amp2*sin(2*pi*f2*t);
s = sin1 + sin2; % Total
% FFT - Schema du spectre
f = -Fe/2: 1/T: Fe/2;
y = abs(fftshift(fft(s)));
plot(f,y)
```

Manipulation de son et filtrage de bruit

1. Exo Alien avec sptool

```
[s, Fs] = audioread('alien_bruit1.wav');
bruit = s(6 * Fs: 15 * Fs);
sptool :
    -import data from workspace => data=s frequency=Fs
    -new > filter : bandstop, butterworth, order 5, Fc1=2490, Fc2=2510
    -apply filter to signal(s) => export sig2
s = [s(1:6*Fs); sig2.data; s(15*Fs: length(s))];
spectrogram(s, 1024, [], 1024, Fs, 'yaxis');
```

2. Exo Alien sans sptool

```
[s, Fs] = audioread('alien_bruit1.wav');
%subplot(2,1,1);
%spectrogram(s, 1024, [], 1024, Fs, 'yaxis');
%title('Signal original');
bruit = s(6 * Fs: 15 * Fs);
[b,a]=butter(5,[2490,2510]/(Fs/2),'stop');
filt = filter(b,a,bruit);
%spectrogram(filt, 1024, [], 1024, Fs, 'yaxis');
s = [s(1:6*Fs); filt; s(15*Fs: length(s))];
%subplot(2,1,2);
spectrogram(s, 1024, [], 1024, Fs, 'yaxis');
%title('Signal filtré');
```

3. Exo Morse avec sptool

```
[s, Fs] = audioread('alien_bruit1.wav');
sptool :
    -import data from workspace => data=s frequency=Fs
    -new > filter : bandpass, butterworth, order 4, Fc1=650, Fc2=800
    -apply filter to signal(s) => export sig2
spectrogram(sig2.data, 1024, [], 1024, Fs, 'yaxis');
```

4. Exo Morse sans sptool

```
[s, Fs] = audioread('MORSE2.wav');
%subplot(2,1,1);
%spectrogram(s, 1024, [], 1024, Fs, 'yaxis');
%title('Signal original');
[b,a]=butter(4,[650,800]/(Fs/2)); % Ordre 4, low:650, high:800, toujours diviser
par Fs/2
filt = filter(b,a,s);
%subplot(2,1,2);
spectrogram(filt, 1024, [], 1024, Fs, 'yaxis');
%title('Signal filtré');
```

5. Afficher le spectre du son MORSE2.wav sans sptool

```
% Afficher le spectre via fft de -Fs/2 à Fs/2
[s,Fs] = audioread('MORSE2.WAV');
duree = length(s)/Fs; % Temps d'un son quoi
y = abs(fftshift(fft(s)));
f = -Fs/2 :1/duree :Fs/2;
plot(f(1:length(f)-1),y);
```

```
% Afficher le spectre via fft de 0 à Fs/2
[s,Fs] = audioread('MORSE2.WAV');
duree = length(s)/Fs;
f = 0: 1/duree: Fs/2;
u = abs(fftshift(fft(s)));
y = u(Fs*duree/2 :Fs*duree);
plot(f,y);
```

Echantillonnage

> **Passer d'une note à une autre note => f*(racine 6ème de 2)***
> ***Passer d'une octave à une autre => f2***

notes	octave 0	octave 1	octave 9
do	32,70	65,4	16744
ré	36,71	...	18 795
mi	41,20	...	21 096
fa	43,65	...	22 351
sol	49,00	...	25 088
la	55,00	...	28 160
si	61,74	...	31 609

Exemples :

- De "do" à "ré" dans l'octave 0, on fait 32,70 *(racine 6ème de 2) = valeur de "ré"
- De "do" de l'octave 0 à l'octave 1, on fait 32,70*2 = valeur de "do" dans octave 1

1) Créer un fichier représentant une suite de notes allant du "la" de la 3ème octave au "la" de la 9ème octave (vous aurez donc 7 notes). Choisir la fréquence échantillonnage à 8000Hz.

a) Ecouter le résultat et visualisez le spectre du nouveau signal.

> ***"la" de la 3ème octave => 55222=440Hz***

```
f=440; Fe=8000; t=0: 1/Fe: 1;

s=0
duree=0;
for i = 0: 6 % vu qu'il reste 6 octaves étant donné qu'on est à la 3ème...
    la = sin(2*pi*f*t);
    s=s+la;
    duree = length(s)/Fe;
    f=f*2; % Fréquence suivante => Passe d'une octave à une autre => multiplier
par 2.
end
% Affichage du spectre via FFT (0 à Fe/2)
f = 0: 1/duree: Fe/2;
u = abs(fftshift(fft(s)));
y = u(Fe*duree/2 :Fe*duree);
plot(f,y);
```

b) Prédire les fréquences que vous allez réellement entendre

f	-f	FS-f	f-FS	2FS-f	FS	Shanon-Nyquist
440	-440	7560	-7560	15560	8000	Fs >= 2*f autrement dit : est-ce que f < Fs/2 ?
880	-880	7120	-7120	15120		Toutes valeur supérieur à Fs/2 sont des fréquences fantômes
1760	-1760	6240	-6240	14240		
3520	-3520	4480	-4480	12480		Valeurs en rouges = Valeurs entendues
7040	-7040	960	-960	8960		Valeurs en vert = fréquences phantômes
14080	-14080	-6080	6080	1920		
28160	-28160	-20160	20160	-12160		

On va entendre :

- Les fréquences de **440 => 3520** (rouge).
- Les fréquences (***fantômes***) **960** et **1920** (vert).

c) Vérifiez si votre prédiction est correcte

Elle est correcte !

On remarque néanmoins, qu'il y a une autre fréquence fantôme qui est sur le spectre que nous n'avons pas observer dans le exel : fréquence vers 3800Hz.

d) Filtrer le signal pour éviter la présence de fréquences fantômes

On a 3 fréquences à retirer et donc 3 filtres à réalisé :

- 960 => Coupe bande d'ordre 4 entre 900 et 1000Hz
- 1920 => Coupe bande d'ordre 4 entre 1850 et 2000Hz
- 3800 => Coupe bande d'ordre 4 entre 3600 et 3900Hz

```
% Filtre 960
[b,a]=butter(4,[900,1000]/(Fe/2),'stop');
filt1 = filter(b,a,s);

% Filtre 1920
[b,a]=butter(4,[1850,2000]/(Fe/2),'stop');
filt2 = filter(b,a,filt1);

% Filtre 3800
[b,a]=butter(4,[3600,3900]/(Fe/2),'stop');
filt3 = filter(b,a,filt2);

% Affichage du spectre - FFT 0 à Fe/2
f = 0: 1/duree: Fe/2;
u = abs(fftshift(fft(filt3)));
y = u(Fe*duree/2 :Fe*duree);
plot(f,y);
```

FFT

```
-) TRES IMPORTANT => FFT dans sptool prends TOUJOURS de 0 à Fe/2 !!!!!
+) La multiplication d'un signal par un autre singal = modulation.
-) La puissance d'un signal = le signal au carré !
```

1) Filtrer le signal ”tarzan.wav” au moyen d’un filtre de Butterworth passe-bas d’ordre 2 de fréquence de coupure Fc=1000Hz en utilisant les instructions ”butter” et ”filter”. Utiliser les fonctions fft et fftshift pour visualiser le spectre avant et après filtrage.

> *PS : Le spectrogram n'est pas demandé dans cette exo, je l'ai gardé pour avoir une autre "vue" du signal.*

```
[s, Fs] = audioread('tarzan.wav');
duree = length(s)/Fs;
subplot(4, 1, 1);
spectrogram(s, 1024, [], 1024, Fs, 'yaxis');
title('Spectrogram Signal de base')

% Affichage du spectre du signal de base - FFT 0 à Fe/2
f = 0: 1/duree: Fs/2;
u = abs(fftshift(fft(s)));
y = u(Fs*duree/2 :Fs*duree);
subplot(4, 1, 2);
plot(f,y);
title('Spectre Signal de base- FFT')

% Filtre
[b,a]=butter(4,1000/(Fs/2),'low');
filt1 = filter(b,a,s);

subplot(4, 1, 3);
spectrogram(filt1, 1024, [], 1024, Fs, 'yaxis');
title('Spectrogram Signal filtré')

% Affichage du spectre du signal filtré - FFT 0 à Fe/2
f = 0: 1/duree: Fs/2;
u = abs(fftshift(fft(filt1)));
y = u(Fs*duree/2 :Fs*duree);
subplot(4, 1, 4);
plot(f,y);
title('Spectre Signal filtré - FFT')
```