CHATIRON Thibault

DENG Fangzhou

SRT3

Automne 2013

TP n°4 :

Spectrogramme

SY06

Sommaire

Introduction

Nous allons étudier ici la limite de l'analyse de Fourier. Cette analyse consisteà décomposer un signal sur une base de fréquences pures, chacune d'entre elles ayant un caractère temporel permanent sur $t \in]-\infty, +\infty[$.

Si le signal de départ est nul sur un certain intervalle de temps, ce sont les interférences entre les différentes fréquences pures (qui apparaissent dans l'analyse de Fourier) qui, lors de la sommation de ces différentes fréquences pures, vont conduire `a la valeur nulle du signal sur l'intervalle considéré. Il s'agit d'un zéro "dynamique".

En fait, si le signal était nul sur l'intervalle de temps considéré, c'est que rien n'a été observé sur cet intervalle (situation de zéro "statique"). Nous allons mettre en évidence cette limitation de l'analyse de Fourier et proposer une des méthodes classiques d'analyse temps-fréquence : le spectrogramme.

Déclaration des variables

Code Matlab

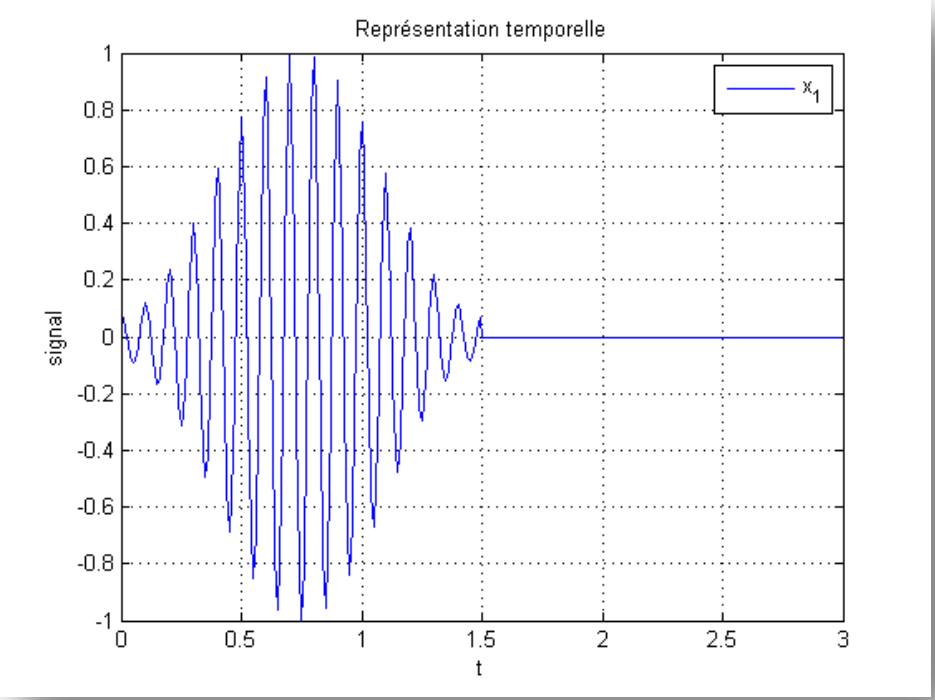
```
clc
clear
close all

f1 = 10;
f2 = 30;
Fe = 100;
```

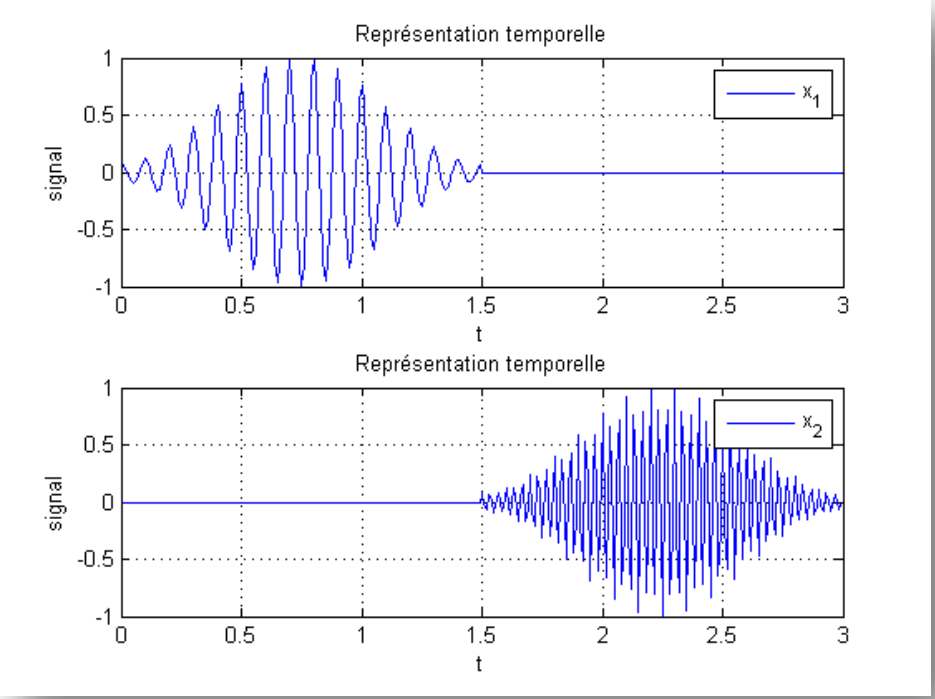
Signaux x_i dans le domaine temporel

Soit les signaux : x1(t) = exp(2jπf1t)h(t − T/2) et x2(t) = exp(2jπf2t)h(t − 3T/2) où h(t) est une fenêtre de Hamming.
On va ici représenter ces deux signaux dans le domaine temporel et les décaler l'un par rapport à l'autre pour pouvoir ensuite les sommer.

Code Matlab

```
%%xi(t)

N = 150;
t = (0:1/Fe:(2*N-1)/Fe)';

x1 = exp(2*pi*j*f1*t);
x2 = exp(2*pi*j*f2*t);

h1 = [ hamming(N) ; zeros(N,1)];
h2 = [ zeros(N,1) ; hamming(N)];

x1 = x1.*h1;
x2 = h2.*x2;

figure(1);
plot(t,real(x1));
xlabel('t');
ylabel('signal');
grid on;
legend('x_1');
title('Représentation temporelle');

figure(2);
subplot(2,1,1),plot(t,real(x1));
xlabel('t');
ylabel('signal');
grid on;
legend('x_1');
title('Représentation temporelle');
subplot(2,1,2),plot(t,real(x2));
xlabel('t');
ylabel('signal');
grid on;
legend('x_2');
title('Représentation temporelle');
```


Représentation temporelle
x1
signal
t
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.5
1
1.5
2
2.5
3

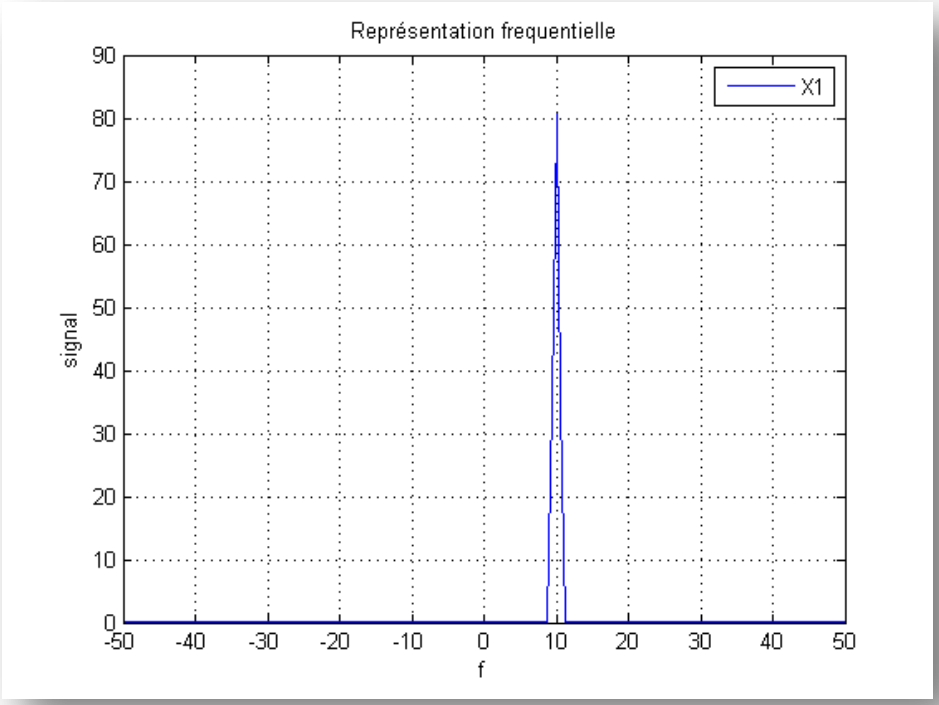

Représentation temporelle
x1
signal
t
1
0.5
0
-0.5
-1
0
0.5
1
1.5
2
2.5
3
Représentation temporelle
x2
signal
t

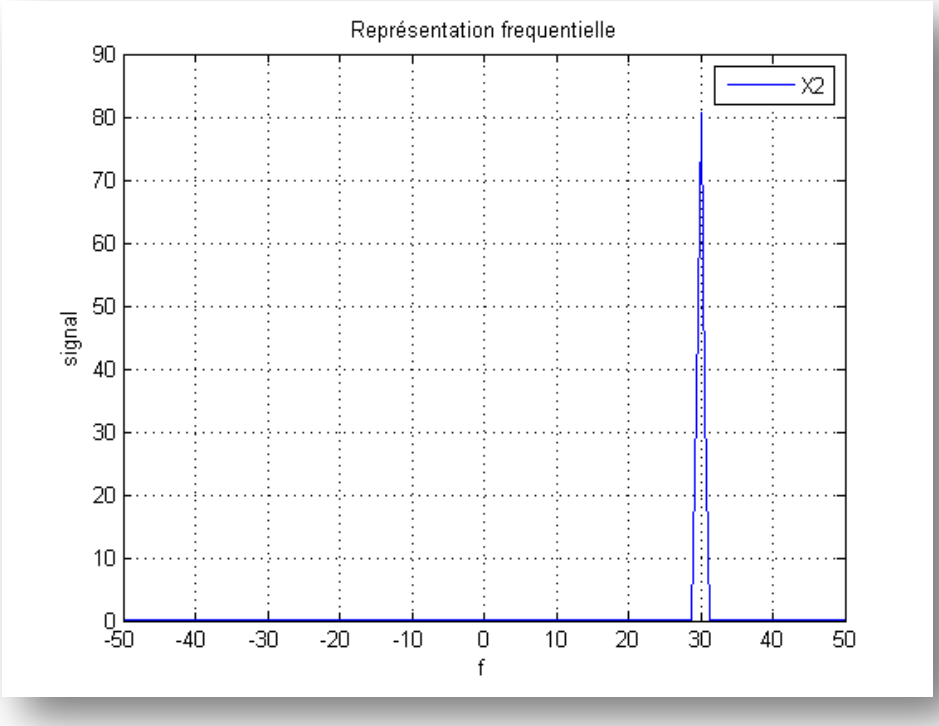
Signaux X_i dans le domaine fréquentiel

On va ici représenter les signaux xi dans le domaine fréquentiel.
On observe des spectres de raies à 10 et 30Hz pour respectivement X1 et X2.

Code Matlab

```
%%fft

k=0;

for f = -Fe/2 : Fe/N : Fe/2
k = k+1;
X1(k) = x1' * exp(2*j*pi*f*t);
end

k=0;

for f = -Fe/2 : Fe/N : Fe/2
k = k+1;
X2(k) = x2' * exp(2*j*pi*f*t);
end

figure(3);
plot((-Fe/2 : Fe/N : Fe/2),abs(X1));
xlabel('f');
ylabel('signal');
grid on;
legend('X1');
title('Représentation frequentielle');

figure(4);
plot((-Fe/2 : Fe/N : Fe/2),abs(X2));
xlabel('f');
ylabel('signal');
grid on;
legend('X2');
title('Représentation frequentielle');
```


Représentation frequentielle
X1
signal
f
90
80
70
60
50
40
30
20
10
0
-50
-40
-30
-20
-10
0
10
20
30
40
50

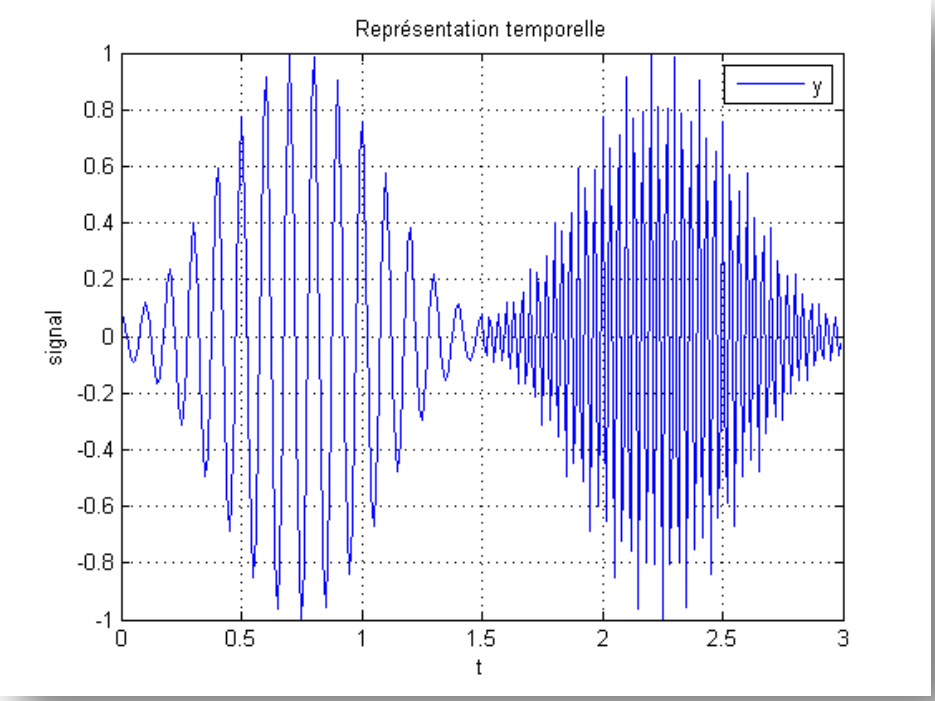

Représentation frequentielle
X2
signal
f
90
80
70
60
50
40
30
20
10
0
-50
-40
-30
-20
-10
0
10
20
30
40
50

Somme des signaux x_i dans le domaine temporel

Soit y(t) = x1(t) + x2(t).
On va ici représenter le signal y dans le domaine temporel.

Code Matlab

```
y=x1+x2;

figure(5);
plot(t,real(y));
xlabel('t');
ylabel('signal');
grid on;
legend('y');
title('Représentation temporelle');
```



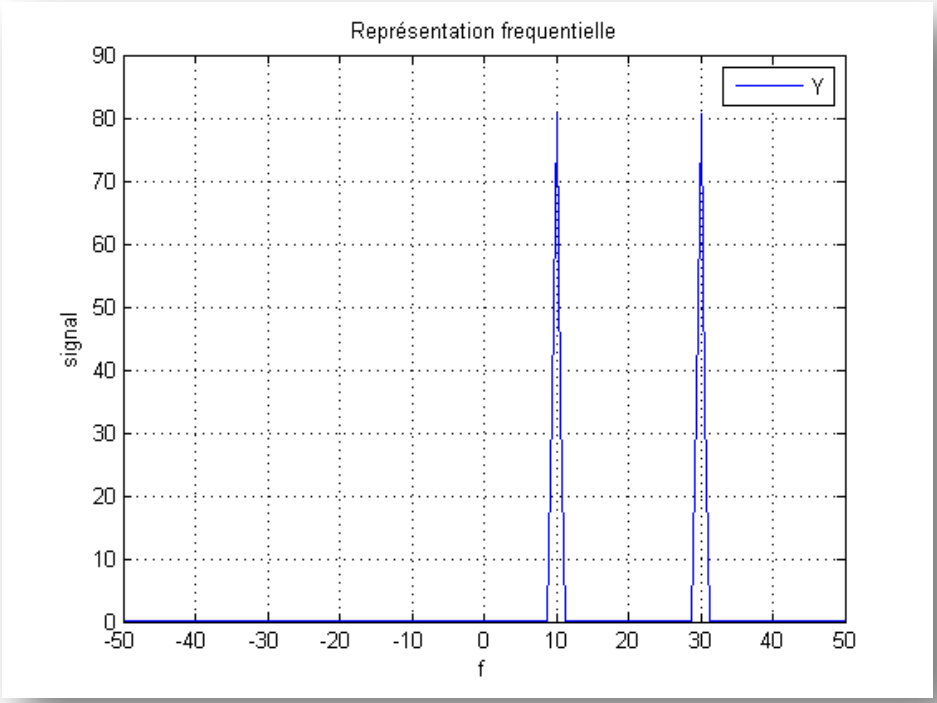
Somme des signaux X_i dans le domaine fréquentiel

On calcule ici la transformée de Fourier de y(t). On trace donc Y dans le domaine fréquentiel.
On voit les deux raies des signaux Xi soient 10 et 30Hz.

Code Matlab

```
%%Y(f)

k=0;

for f = -Fe/2 : Fe/N : Fe/2
```

```
k = k+1;
Y(k) = y' * exp(2*j*pi*f*t);
end

figure(6);
plot((-Fe/2 : Fe/N : Fe/2),abs(Y));
xlabel('f');
ylabel('signal');
grid on;
legend('Y');
title('Représentation frequentielle');
```



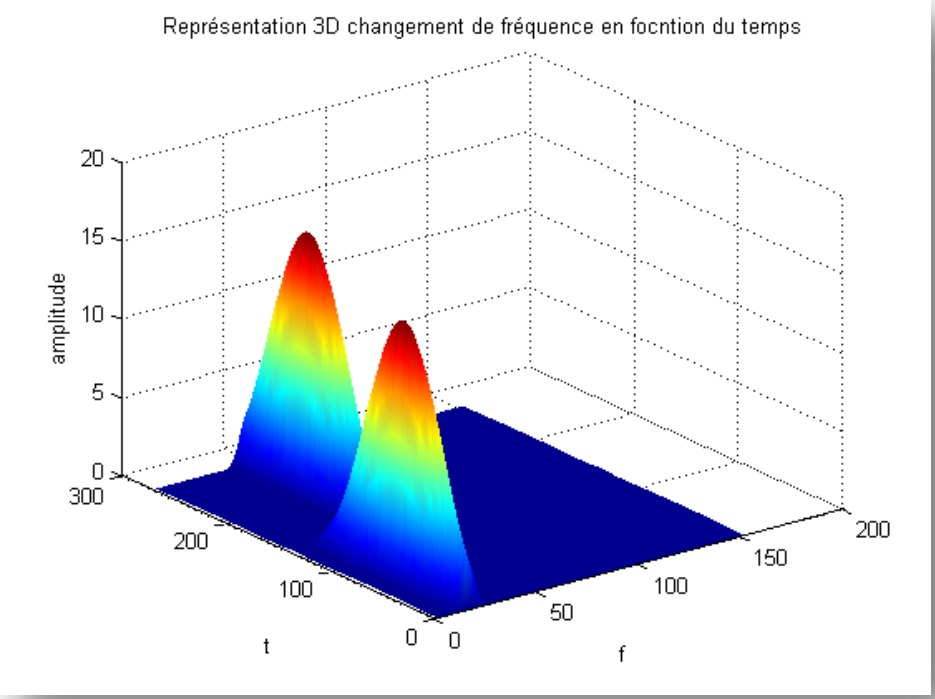
Représentation 3D

On va représenter l'amplitude des deux signaux xi (soit y) en fonction de la fréquence et du temps afin d'observer s'il y a présence ou non de recouvrement.

Code Matlab

```
%%imagesc, surf, mesh

taillefenetre = 32;

for m = 0 : 1: 2*N-1-taillefenetre

hy = [zeros(m,1);hamming(taillefenetre);zeros(2*N-m-taillefenetre,1)];
y1 = y.*hy;
```

```
m = m+1;

    k=0;

    for f = 0 : Fe/N : Fe
    k = k+1;
    Y1(m+1,k) = y1' * exp(2*j*pi*f*t);
    end

end

figure(7);
surf(abs(Y1));
shading('interp');
xlabel('f');
ylabel('t');
zlabel('amplitude');
grid on;
title('Représentation 3D changement de fréquence en focntion du temps');
```



Etude du son wav

On charge le son wav pour pouvoir traiter et analyser les données qu'il contient : une fréquence d'échantillonnage et un tableau (de deux colonnes pour un son stéréo).
On créé une fonction *importfile* pour pouvoir charger les données du fichier wav (code-source en annexe).

On utilise le spectrogramme pour représenter ces données. Pour cela, on va procéder par étapes pour pouvoir comparer leur vitesse d'exécution.

1ère étape :
On utilise un calcul de méthode de transformée de Fourier (algorithme manuel). Ici, le temps est trop long pour pouvoir faire une représentation (même pour une seule boucle).

2ème étape :
On utilise maintenant *fft* et on obtient un résultat (pour une seule boucle) de 450ms en moyenne pour avoir un résultat à analyser. Pour la boucle entière, il aurait donc fallu plus d'1h.

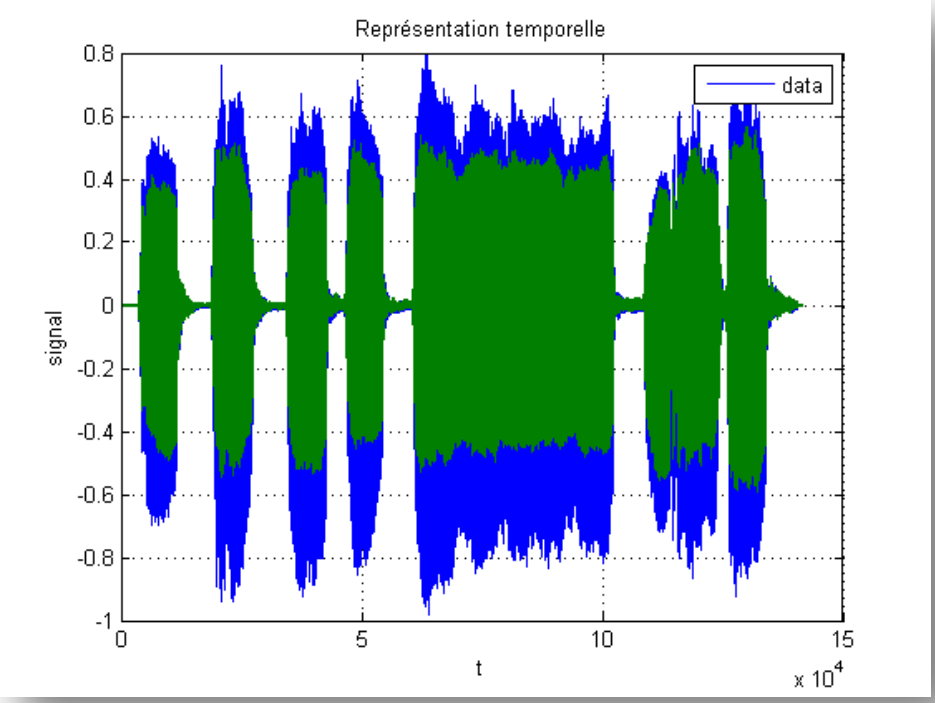
3ème étape :
On utilise directement *spectrogram* et obtient immédiatement un résultat graphique 3D.

Code Matlab

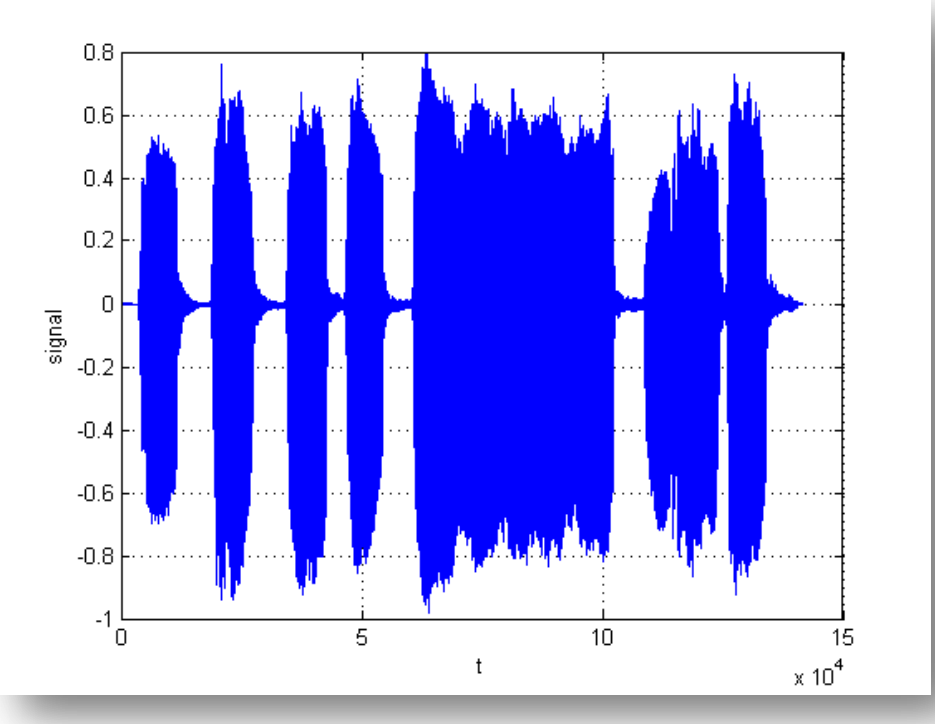
```
% clear;
% close all;

importfile('SY06_2013_A_TP4_Exemples.wav'); %creation d'une fonction

data1 = data(:,1);
data2 = data(:,2);

taillefenetre = 1024;

N = length(data);
t = [0:1/fs:(N-1)/fs]';

% tic
% for m = 0% : 1: 2*N-1-taillefenetre
% hy = [zeros(m,1);hamming(taillefenetre);zeros(N-m-taillefenetre,1)];
% y1 = data1.*hy;
% m = m+1;
%%
k=0;
%%
for f = -fs/2 : fs/N : fs/2
% k = k+1
% Y1(m+1,k) = y1' * exp(2*j*pi*f*t);
% end
%%
end
% toc
%%
figure(8);
% surf(abs(Y1));
% shading('interp');
% xlabel('f');
% ylabel('t');
% zlabel('amplitude');
% grid on;
% title('Représentation 3D changement de fréquence en focntion du temps');
% trop long!!!! meme pour une seule boucle

tic
```

```
for m = 0% : 1: 2*N-1-taillefenetre
hy = [zeros(m,1);hamming(taillefenetre);zeros(N-m-taillefenetre,1)];
y1 = data1.*hy;
m = m+1;
fft(y1);
end
toc % 1.358265 seconds.

tic
for m = 0% : 1: 2*N-1-taillefenetre
hy = [zeros(m,1);hamming(taillefenetre);zeros(N-m-taillefenetre,1)];
y1 = data2.*hy;
m = m+1;
fft(y1);
end
toc
% Elapsed time is 0.006700 seconds.
% Elapsed time is 0.008249 seconds.

figure(9);
plot(data);
xlabel('t');
ylabel('signal');
grid on;
legend('data');
title('Représentation temporelle');

data1 = data(:,1);
data2 = data(:,2);

figure(10);
plot(data1);
xlabel('t');
ylabel('signal');
grid on;

figure(11);
plot(data2);
xlabel('t');
ylabel('signal');
grid on;

figure(12);
spectrogram(data1);
shading('interp');
xlabel('f');
ylabel('t');
zlabel('amplitude');
grid on;
title('Spectrogramme data1');

figure(13);
spectrogram(data2);
shading('interp');
xlabel('f');
ylabel('t');
zlabel('amplitude');
grid on;
title('Spectrogramme data2');
```

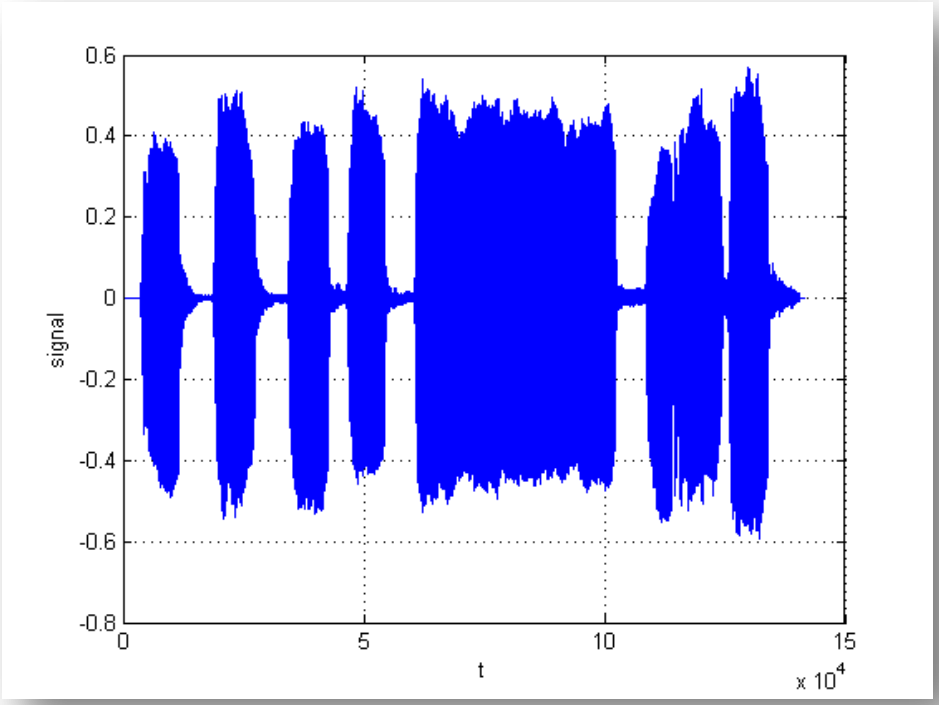
On va ici représenter notre son wav stéréo dans le domaine temporel :



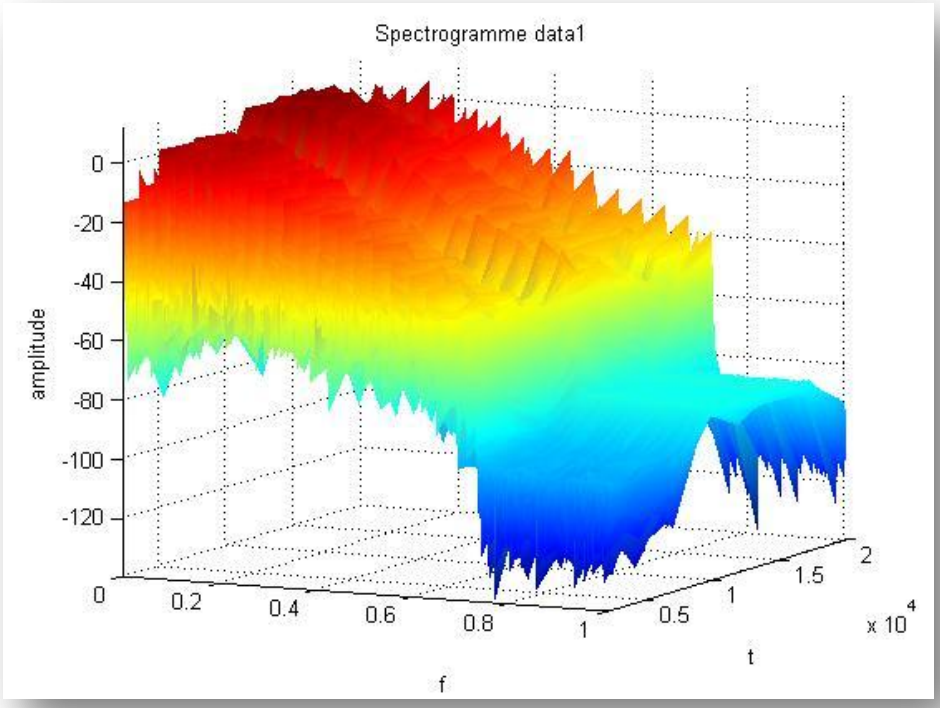
On ne représente ici que le canal 1 du son wav dans le domaine temporel :



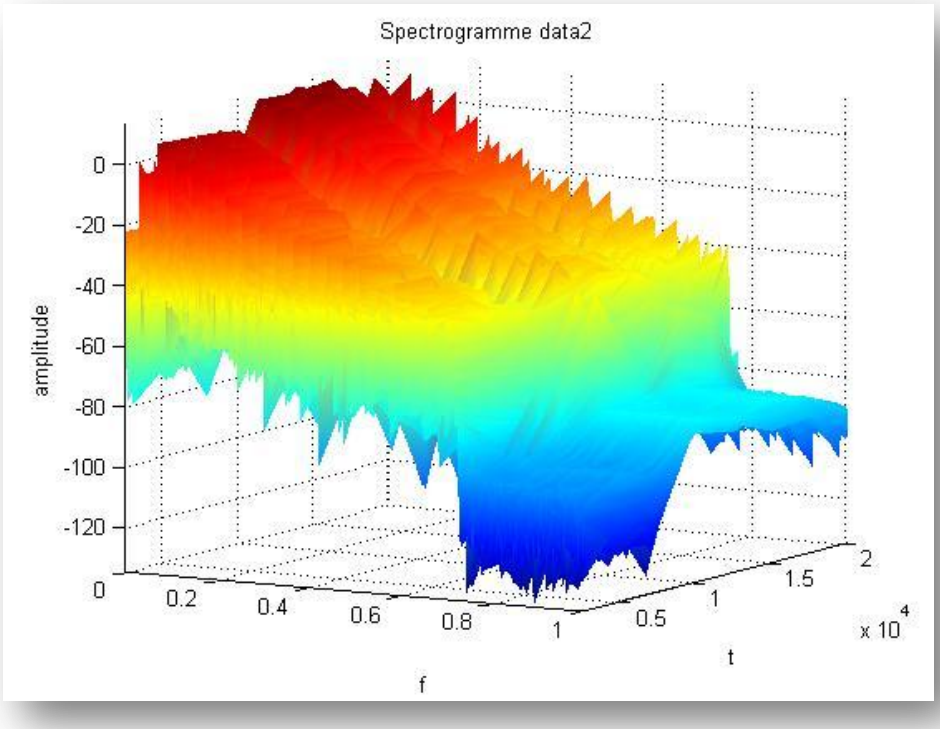
On ne représente ici que le canal 2 du son wav dans le domaine temporel :



On représente ici les données du canal 1 du son wav :



On représente ici les données du canal 2 du son wav :


Spectrogramme data2
amplitude
0
-20
-40
-60
-80
-100
-120
0
0.2
0.4
0.6
0.8
1
f
0.5
1
1.5
2
x 10^4
t

Conclusion

Le TP nous a permis d'étudier les limites de l'analyse de Fourier grâce à l'analyse temps-fréquence par spectrogramme.

Il nous a également montré quels étaient les outils les plus efficaces en matière de résultat mais surtout de temps.

Le spectrogramme de x(t) est le module au carré de la transformée de Fourier "à court terme" du signal, c'est à dire la transformée de Fourier du signal multiplié par une fenêtre centrée sur l'instant t.

On obtient donc une représentation qui traduit l'évolution du contenu fréquentiel du signal en fonction du temps t.

Annexe

Code Matlab

```
function importfile(fileToRead1)
%IMPORTFILE(FILETOREAD1)
% Imports data from the specified file
% FILETOREAD1: file to read
% Auto-generated by MATLAB on 02-Dec-2013 18:43:14
% Import the file

newData1 = importdata(fileToRead1);

% Create new variables in the base workspace from those fields.

vars = fieldnames(newData1);

for i = 1:length(vars)
assignin('base', vars{i}, newData1.(vars{i}));
end
```