Compte rendu des Travaux Pratiques du cours d'ES101

Ouzone Yann-Ilya

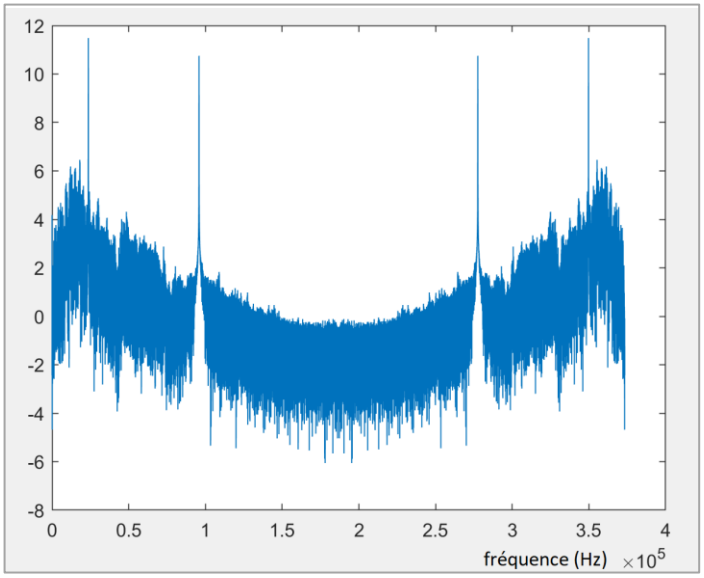
Mesnard Jules-Yann

SOMMAIRE

A. Sujet 1

I. Présentation

Dans ce TP, on considère un signal qui a été brouillé par deux autres signaux de fréquence f1 & f2.



L'objectif de cet exercice est donc de supprimer ces deux signaux parasites pour isoler le signal d'origine. On utilise pour cela un filtre RII. On va donc construire ce filtre RII : on souhaite obtenir un filtre à encoche qui supprime les fréquence f1 & f2.

On pose H(z)=$H_1(z) * H_2(z)$

H(z)=$\frac{(z-z1)(z-z1*)}{(z-p1)(z-p1*)} * \frac{(z-z2)(z-z2*)}{(z-p2)(z-p2*)}$ avec $z1 = e^{2\pi jf1/Fe}$, z2=$e^{2\pi jf2/Fe}$ et p1=ρz1, p2=ρz2 en prenant ρ=0,95

Zéros de H

Pôles de H

On a:

$$H1(z)= \frac{(z-z1)(z-z1*)}{(z-p1)(z-p1*)} = \frac{z^2-2*cos(\frac{2\pi f1}{Fe})+|z1|^2}{z^2-2\rho*cos(\frac{2\pi f1}{Fe})+\rho^2} = \frac{z^{-2}-2*cos(\frac{2\pi f1}{Fe})*z^{-1}+1}{\rho^2 z^{-2}-2*cos(\frac{2\pi f1}{Fe})*z^{-1}+1}$$

De même,

$$H2(z)=\frac{(z-z2)(z-z2*)}{(z-p2)(z-p2*)}=\frac{z^2-2*cos(\frac{2\pi f2}{Fe})+|z2|^2}{z^2-2\rho*cos(\frac{2\pi f2}{Fe})+\rho^2}=\frac{z^{-2}-2*cos(\frac{2\pi f2}{Fe})*z^{-1}+1}{\rho^2 z^{-2}-2*cos(\frac{2\pi f2}{Fe})*z^{-1}+1}$$

Ainsi, on a mis sous la forme canonique :

$$H(z)=\frac{a0+a1*z^{-1}+a2*z^{-2}+\ldots+an*z^{-n}}{b0+b1*z^{-1}+b2*z^{-2}+\ldots+bn*z^{-n}}$$

Ce qui nous permettra d'utiliser la fonction *filter* de Matlab :

Y=filter(x1(a0,a1,…,an) ;x2(b0,b1,…,bn))

II. Code

```
clear all;

%récupération des signaux
[x1,Fe] = audioread('Moll.wav');
X1 = fft(x1);
L1=length(x1);



N=floor(L1/2);
x=[1:N];
Y=X1(1:N);

%plot(x,Y);
freq_vect=(0:L1-1)*Fe/L1;
plot(freq_vect,abs(X1));

%analyse frequentielle:détermination des fréquences réduites des bruits
[m1,N1]=max(Y);
f1=(N1-1)*Fe/L1;

tmp=Y;
tmp(N1)=0;
[m2,N2]=max(tmp);
f2=(N2-1)*Fe/L1;

fred1=2*pi*f1/Fe;
```

```
fred2=2*pi*f2/Fe;

%construction du filtre

Z1=exp(i*fred1); %zéros de la fonction de transfert
Z2=exp(i*fred2);
rho=0.98;
P1=rho*Z1;  %pôles de la fonction de transfert
P2=rho*Z2;


A1=[1,-2*real(Z1),1];  %coefficiants de la fonction de filtrage
B1=[1,-2*real(P1),norm(P1)^2];

A2=[1,-2*real(Z2),1];
B2=[1,-2*real(P2),norm(P2)^2];

%filtrage du signal
S1=filter(A1,B1,x1);

%2ème filtrage
S2=filter(A2,B2,S1);
M=max(S2);
S2=S2/norm(M)  %normalisation du signal

% affichage de l'audio filtré
fft_s2=fft(S2);
figure
plot(freq_vect,abs(fft_s2));



audiowrite('resultat.wav',S2,Fe) ;
sound(S2,Fe) ;
```

B. Sujet 2

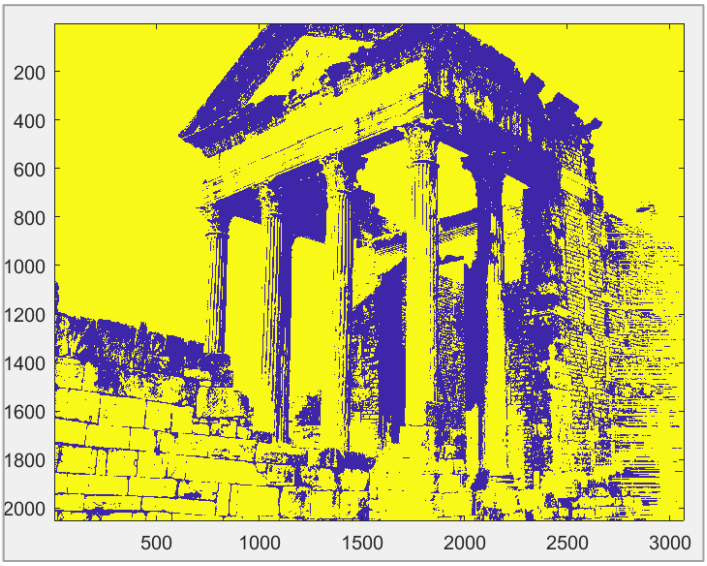
a) Exercice 1

I. Présentation

Dans cet exercice, on se propose de reconstituer une image pixelisée dont on a décalé les pixels ligne par ligne. Comme ce sont des pixels, il y a très peu de différence entre chaque ligne consécutive, c'est pourquoi on se propose d'utiliser la fonction corrélation afin de comparer chaque ligne avec la précédente (on suppose que la ligne de pixel du haut est au bon endroit. Chaque ligne constituera une liste de pixels. On utilisera alors la corrélation entre deux listes de pixels.

On va donc utiliser une boucle *for* pour corriger l'image ligne par ligne. On utilise ensuite la fonction c*ircshift* prenant en paramètre un vecteur et un décalage en décalant ce vecteur selon le décalage donné.

Dans un deuxième temps, (deuxième boucle *for*), on souhaite décaler l'image de 280 pixels (valeur donnée par le professeur) pour la recentrer. On obtient alors l'image corrigée.



II. Codes

```
clear all;
close all;
clc;
B=imread('fichier2.bmp','bmp');
B=255*B;

%colormap(GRAY);

[L,H]=size(B);
 for j = 1:L-1

    x1=B(j,:);
    x2=B(j+1,:);
    C=xcorr(x1,x2);
    [M,i_max] = max(C);
    tmp=circshift(x2,i_max);
    B(j+1,:)=tmp;
 end
i=280;
 for j = 1:L-1

    x=B(j,:);

    tmp=circshift(x,-i);
    B(j,:)=tmp;
 end
```

b) Exercice 2

I. Présentation

Dans cet exercice, on cherche à retrouver un signal d'origine dont les hautes fréquences ont été permutées avec les basses fréquences. Il faut donc isoler chaque haute/basse fréquence et la permuter.

Pour cela, nous avons tout d'abord procédé à la récupération du signal y et de la fréquence d'échantillonage à l'aide de la fonction *audioread* de matlab. Ensuite, on effectue une analyse spectrale, pour cela il faut transformer notre signal temporel en un signal fréquentiel à l'aide de la transformée de Fourier. On utlise pour cela la fonction *fft* de matlab. On obtient alors le spectre en fréquence du signal échantillonné F. Il faut maintenant inverser les fréquences, donc les composantes du vecteur F. Enfin, on applique la transformée de Fourier inverse au signal F avec la fonction *ifft* pour obtenir le signal en temporel corrigé. Il suffit alors de reconstituer le signal audio avec la fonction *sound.*

II. Codes

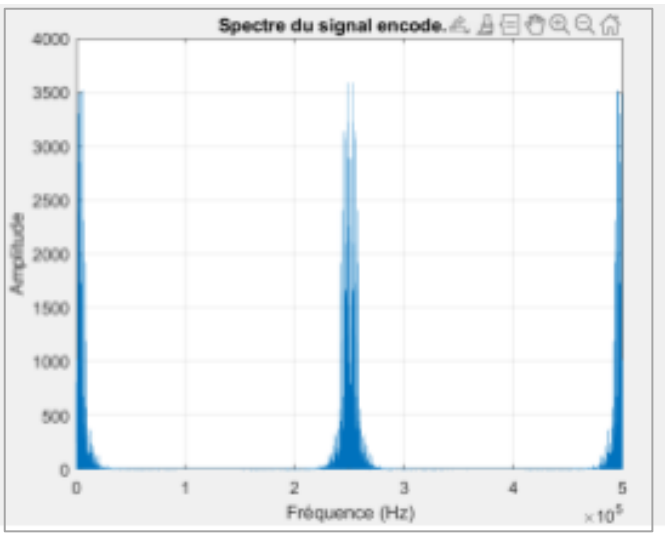
```
clear all;
[Y,FE]=audioread("canal.wav");
N=length(Y);

F=fft(Y);
tmp1=F(2:(N/2));
tmp2=F((N/2+2):N);
tmp1b=tmp1(N/2-1:-1:1);
tmp2b=tmp2(N/2-1:-1:1);
F(2:(N/2))=tmp1b;
F((N/2+2):N)=tmp2b;
y=ifft(F);
sound(y,FE);
```

c) Exercice 3

I. Présentation

Dans cet exercice, on dispose d'un signal composé de deux signaux originaux qu'on souhaite retrouver.



On utilise ensuite l'indice n°2 : on calcule l'effet sur la transformée de Fourier de l'opération qui consiste à ajouter un 0 entre chaque valeur du signal.

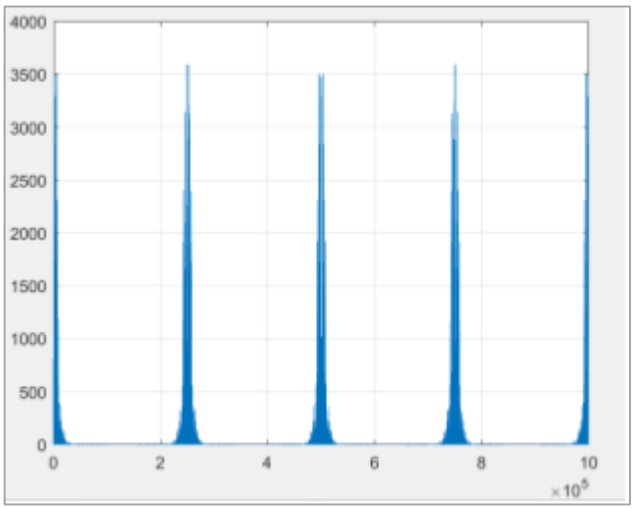
En notant x le nouveau signal, on calcule sa transformée de Fourier :

$$TF(v) = \sum_{k=1}^{2n} x_k * e^{2i\pi vk}$$

$$TF(\nu') = \sum_{k=1}^{n} y_k * e^{2i\pi\nu' k}$$

En ayant posé $\nu' = 2\nu$

On observe que la fréquence d'échantillonnage est doublée, ce qui crée des pics supplémentaires symétriquement.



Il suffit alors de créer deux signaux avec les valeurs paires et impaires de Encode.wav.

II. Codes

```
clear all;
close all;
clc;

[y,Fe] = audioread('encode.wav');
y1=abs(fft(y));
L=length(y);
x1=y(1:2:L-1);
x2=y(2:2:L);



 x=[1:L/2];
 plot(x,x1);

audiowrite('resultat23.wav',x2,Fe) ;
sound(x2,Fe) ;
```

C. Sujet 3

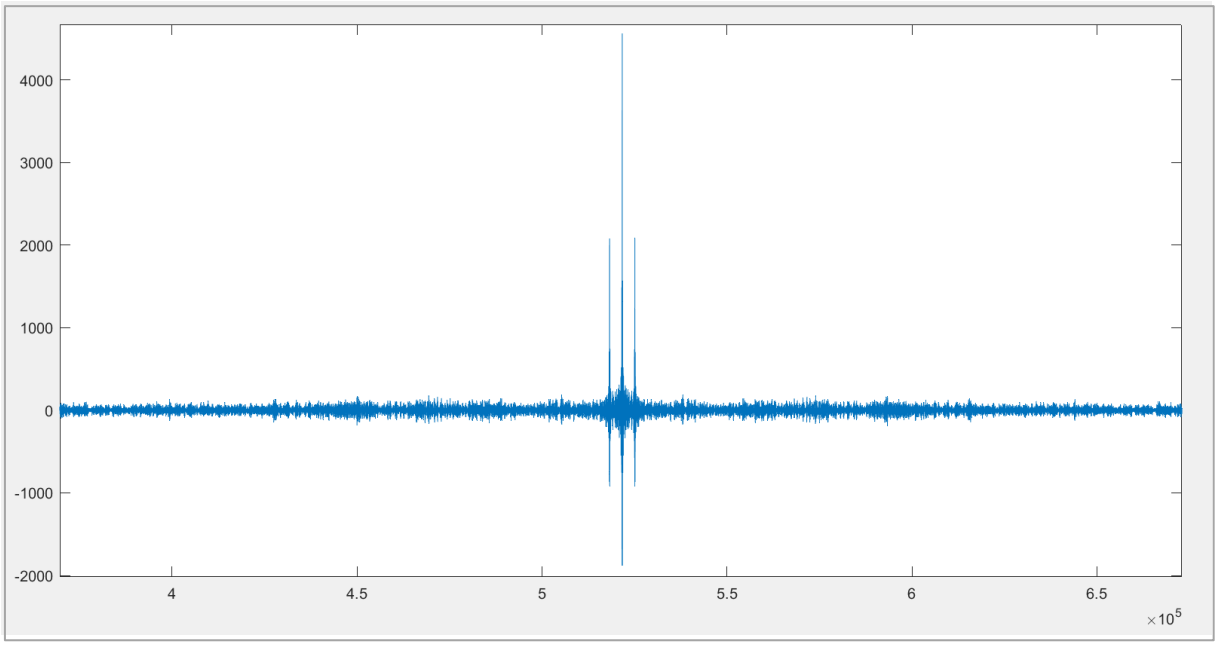
a) Exercice 1

I. Présentation

L'objectif de cette partie du TP est de retirer l'echo du signal sonore qui nous a été donné. Pour celà, on fait l'hypothèse que le signal d'origine que l'on cherche à extraire est un signal blanc. Ensuite, nous introduisons la fonction de transfert d'un filtre RFI qui permet de modéliser l'apparition de l'écho à partir de ce signal original

$$Z = \frac{1}{1 + \alpha Z^{-p0}}$$

Pour déterminer p0 et α on affiche la courbe d'autocorrélation du signal :



On trouve p0=3418

Maintenant, pour déterminer α, il faut écrire les autocorrélations en 0 et p0 :

$r_{xx}(0) = E\big(x(n)x(n)\big)$

$= E[s(n) + \alpha s(n - p0)] * [s(n) + s(n - p0)]$

$= E[s2] + \alpha E[s2]$

$= (1 + \alpha^2)\square^2$ car s est un bruit blanc

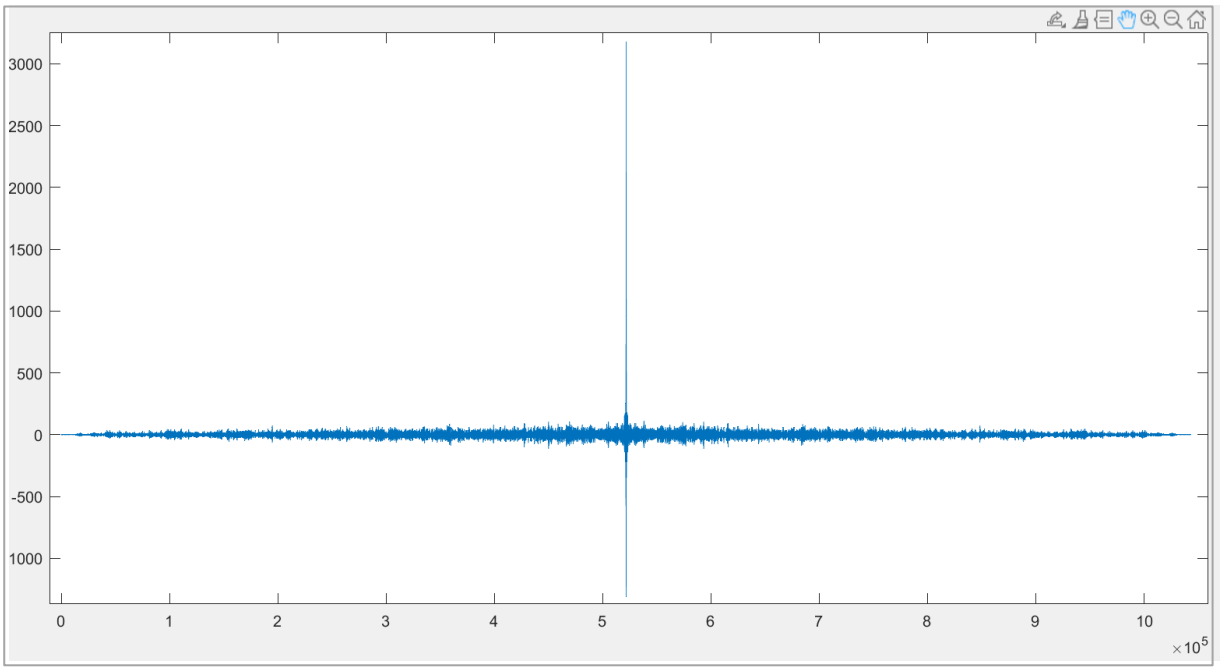
$r_{xx}(p0) = E[x(n)x(n - p0)]$

$= \alpha\square^2$

En résolvant ce système, on trouve α=0,6472

Il ne nous reste plus qu'à corriger le signal à l'aide du filtre donné dans l'énoncé.

On trouve alors un signal sans écho :



Autocorrélation du signal sans écho

II. Codes

b) Exercice 2

I. Présentation

Le but de l'exercice est de mettre en place la technique d'inversion de puissance.

On cherche ρ tel qu'il minimise $E[\varepsilon(n)^2]$: pour cela on calcule $E[\varepsilon(n)^2]$: puis sa dérivée.

$$\rho = \frac{a1a2+b1b2}{a1^2+a2^2} = -\frac{E[x1(n)x2(n)]}{E[x2^2(n)]}$$

De plus, par le calcul avec ρ trouvé précédemment, $E[\varepsilon(n)x2(n)] = 0$ donc on peut en effet dire que ε(n) et $x2(n)$ sont décorrélés.

On calcule ensuite le rapport signal sur bruit des signaux x2 et ε.

$$SNR(x2(n)) = \frac{E[|a_2 * s(n)|^2]}{E[|b_2 * w(n)|^2]} = \frac{{a_2}^2}{{b_2}^2}$$

A présent, l'énoncé nous rappelle que $E[\varepsilon(n)x2(n)] = 0$

On trouve ainsi que :

$$SNR(x2(n)) = SNR(\varepsilon(n))^{-1}$$

III. Codes