Enseignant principal du module
M.FIORINA Jocelyn

Tuteur projet
M. ROBLIN Christophe

29/05/2024

Abdoulaye TRAORE

ES101

RAPPORT DES PROJETS

Année universitaire 2023-2024

ENSTA

IP PARIS

Sommaire

Introduction *2*

1. Projet 1 ..... *3*

1.1. Sujet 1 ..... **3**
1.1.1. Observation et analyse de la situation 3
1.1.2. Méthodologie et Expérience 3
1.1.3. Résultats et Commentaires 4
1.1.4. Code 5

1.2. Sujet 2 ..... **6**
1.2.1. Observation et analyse de la situation 6
1.2.2. Méthodologie et Expérience 6
1.2.3. Résultats et Commentaires 7
1.2.4. Code 8

2. Projet 2 ..... *9*

2.1. Sujet 1 ..... **9**
2.1.1. Observation et analyse de la situation 9
2.1.2. Méthodologie et Expérience 10
2.1.3. Résultats et Commentaires 10
2.1.4. Code 11

2.2. Sujet 2 ..... **12**
2.2.1. Observation et analyse de la situation 12
2.2.2. Méthodologie et Expérience 12
2.2.3. Résultats et Commentaires 13
2.2.4. Code 14

2.3. Sujet 3 ..... **15**
2.3.1. Observation et analyse de la situation 15
2.3.2. Méthodologie et Expérience 15
2.3.3. Résultats et Commentaires 17
2.3.4 Code 18

3. Projet 3 ..... *19*

3.1. Sujet 1 ..... **19**
3.1.1. Observation et analyse de la situation 19
3.1.2. Méthodologie et Expérience 19
3.1.3. Résultats et Commentaires 21
3.1.4. Code 22

3.2. Sujet 2 ..... **23**
3.2.1. Observation et analyse de la situation 23
3.2.2. Méthodologie et Expérience 24
3.2.3. Résultats et Commentaires 24
3.2.4. Code 25

3.3. Sujet 3 ..... **26**
3.3.1. Observation et analyse de la situation 26
3.3.2. Méthodologie et Expérience 26
3.3.3. Résultats et Commentaires 27
3.3.4. Code 28

Conclusion *29*

Introduction

Ce rapport présente l'application des techniques de traitement du signal vues dans le cours ES101 pour résoudre divers problèmes pratiques à l'aide de Matlab, qui offre une large gamme de fonctions prédéfinies pour ce type de traitement.

Ces projets se sont déroulés en 3 séances et s'articulent autour du filtrage de signaux corrompus par des bruits, du mélange de deux signaux utiles, du décalage d'une image et de la présence d'échos, entre autres.

1. Projet 1

1.1. Sujet 1

Dans ce sujet, il s'agit à partir de la lecture d'un signal sonore et l'observation de son spectre, de filtrer ce dernier de deux fréquences f1 et f2 qui brouillent le signal utile.

1.1.1. Observation et analyse de la situation

Dans cette section, nous observons et analysons le signal audio brouillé contenu dans le fichier **Mo11.wav**. (figure 1-2)
Pour analyser le contenu fréquentiel du signal, nous calculons la transformée de Fourier du signal à l'aide de la fonction fft.
La transformée de Fourier est ensuite centrée autour de zéro avec fftshift, et les fréquences correspondantes sont tracées sur un graphique. (figure 1-1)
Ce spectre fréquentiel permet d'identifier les composantes en fréquence qui composent le signal, ce qui est essentiel pour localiser les fréquences brouillées.

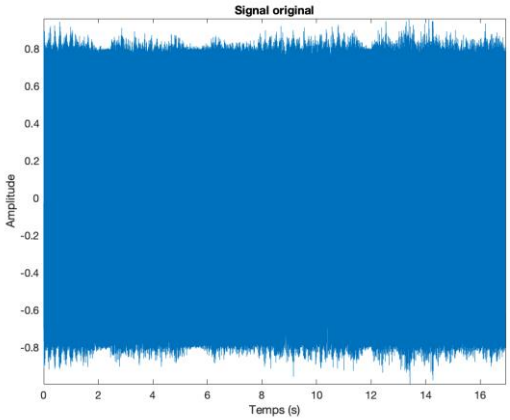

Figure 1-2 – Signal original

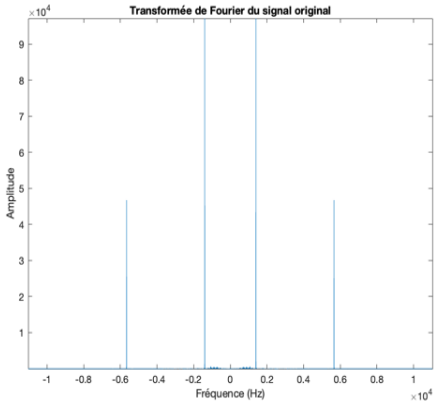

Figure 1-1 – Spectre du signal original

1.1.2. Méthodologie et Expérience

La méthodologie adoptée pour supprimer les fréquences indésirables du signal repose sur l'application d'un filtre réjecteur de type RII (Réponse Infinie Impulsionnelle).
Nous identifions les fréquences brouillées f1 et f2 en trouvant les pics principaux dans le spectre fréquentiel du signal original (figure 1-1).

Pour chaque fréquence identifiée, nous calculons les zéros correspondants du filtre (z1 et z2) en utilisant l'exponentielle complexe et ajoutons également les conjugués de ces zéros pour obtenir une réponse symétrique.
Les pôles du filtre sont positionnés juste à l'intérieur du cercle unitaire avec un facteur de

Le filtre utilisé est donc de la forme :

$$H(Z) = \prod_{i \in \{1,2\}} H_i(Z) \text{ avec } H_i(Z) = \frac{(Z - e^{\frac{2i\pi f_i}{Fe}})(Z - e^{\frac{-2i\pi f_i}{Fe}})}{(Z - 0.99 * e^{\frac{2i\pi f_i}{Fe}})(Z - 0.99 * e^{\frac{-2i\pi f_i}{Fe}})}$$

réduction alpha pour assurer la stabilité du filtre. Ici alpha empirique est : 0.99
Les coefficients des polynômes des zéros (b) et des pôles (a) sont obtenus à l'aide de la fonction poly.
Le filtre est appliqué au signal original à l'aide de la fonction filtre, produisant ainsi le signal filtré y2.

1.1.3. Résultats et Commentaires

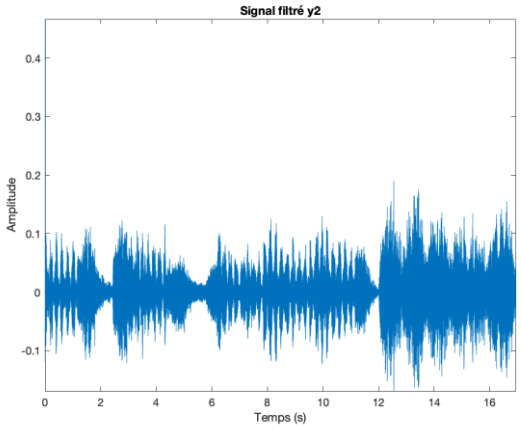

Figure 1-4 – Signal filtré

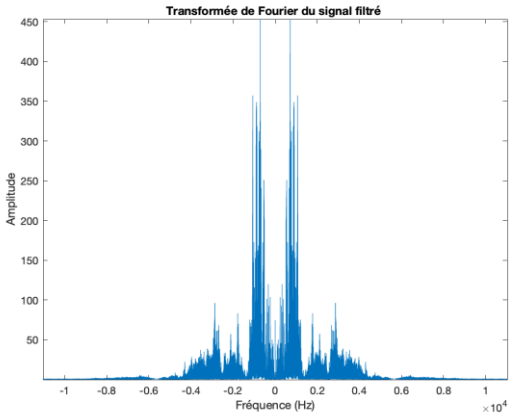

Figure 1-3 - Spectre du signal filtré

Les résultats de l'application du filtre sont observés à travers les graphiques (figure 1-3 et figure 1_4) et des analyses auditives .

Le signal filtré y2 (figure 1-4) est tracé en fonction du temps, montrant une réduction notable des perturbations présentes dans le signal original. Cela est visible par une forme d'onde plus régulière et moins bruitée.

Le spectre fréquentiel du signal filtré est tracé (figure 1-3) pour vérifier l'atténuation des fréquences f1 et f2. Les pics correspondant à ces fréquences sont considérablement réduits ou totalement supprimés, indiquant l'efficacité du filtre.

Le signal filtré est enregistré dans un nouveau fichier Mo11_new.wav.
Cette étape permet de vérifier audiblement la qualité du filtrage et de s'assurer que les fréquences indésirables ont été supprimées convenablement.

Il faut toutefois noter que cette approche peut être plus complique lorsque les fréquences qui brouillent le signal sont moins faciles à détecter (le bruit est dominant).
Dans notre cas, un essai sur un enregistrement vocal WhatsApp converti en un fichier wav a fourni un résultat bien moins spectaculaire .

1.1.4. Code

```
clear all ; close all

%Chargement,lecture et échantillonnage du signal audio
filename = 'Mo11.wav';
[y, fe] = audioread(filename);
T = 1/fe;

% Affichage du signal original brouillé
t = (0:length(y)-1)*T;
figure(1)
plot(t, y);
title('Signal original');
xlabel('Temps (s)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_1','Format','png','color','cmyk');

%TFy : transformée de fourier de y ;
% Calcul de la transformée de Fourier des signaux
TFy = fftshift(fft(y));
f = linspace(-fe/2, fe/2, length(TFy));

% Affichage de la transformé de Fourier su signal original
figure(2)
plot(f, abs(TFy));
title('Transformée de Fourier du signal original');
xlabel('FrV©quence (Hz)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_2','Format','png','color','cmyk');
figure(3)

%Recherche des valeurs des frequences f0 et f1 V† couper par un filtre
N=length(y);
f_temp=0:fe/N:fe/2; % Pour des raisons de symetrie

% PremiV®re frV©quence
[~,indice_1]=max(abs(TFy(1:N/2)));
f0=f(indice_1);

%  Deuxième fréquence
TFy(indice_1)=0;
[~,indice_2]=max(abs(TFy(1:N/2)));
f1=f(indice_2);


%Filtrage du signal original
z0=exp(-1j*2*pi*f0*T);
z1=exp(-1j*2*pi*f1*T);
alpha=0.8;
Z=[z0,conj(z0),z1,conj(z1)];
P=alpha*Z;
b=poly(Z); %Fonction pour trouver les coefficients étant donné des zéros;
a=poly(P);
y2=filter(b,a,y);

% Affichage du signal filtré
figure(3)
plot(t, y2);
title('Signal filtrV© y2');
xlabel('Temps (s)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_3','Format','png','color','cmyk');


%Spectre du signal filtré y2 et affichage
figure(4)
plot(f, abs(fftshift(fft(y2))))
title('Transformée de Fourier du signal filtre ');
xlabel('Fréquence (Hz)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_4','Format','png','color','cmyk');

%Ecriture et lecture du son filtré
audiowrite('Mo11_new.wav',y2,fe);
sound(y2, fe);
```

1.2. Sujet 2

Dans ce sujet, il s'agit de fabriquer un signal stéréo avec un effet de mouvement de la source autour du récepteur à partir d'un signal initial mono accusant une différence de marche temporelle entre les deux oreilles.

1.2.1. Observation et analyse de la situation

Dans cette section, nous avons procédé à l'observation et à l'analyse initiale du signal audio contenu dans le fichier **Erlk.wav**.
Le spectre du signal (figure 1-5) est ensuite tracé pour observer les composantes fréquentielles présentes dans le signal.

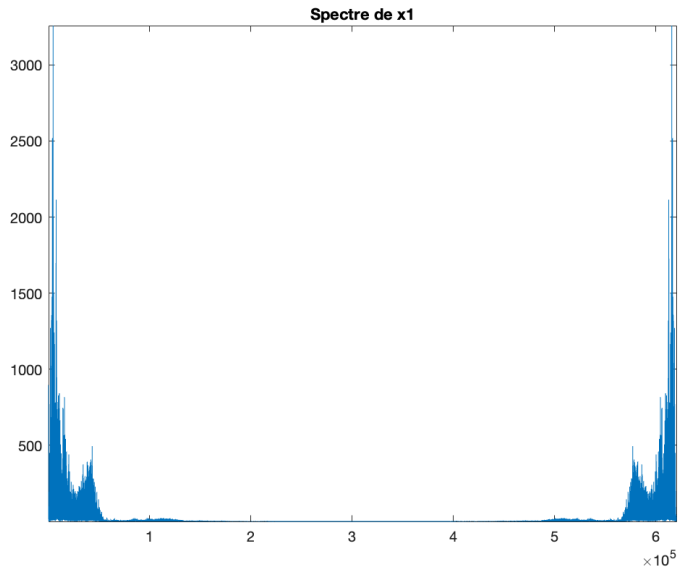

Figure 1-5 - Spectre du signal original

1.2.2. Méthodologie et Expérience

Pour cette expérience, nous avons conçu un processus de manipulation du signal audio afin de créer un effet stéréo basé sur une méthode de décalage temporel. Le signal original est traité pour générer un second signal décalé temporellement, en fonction d'un paramètre de distance et de la vitesse du son. Le code utilise une boucle pour construire le signal décalé x2 en calculant le décalage temporel $\tau = d*\sin(\Theta)/c$ basé sur l'angle Θ et la distance d.

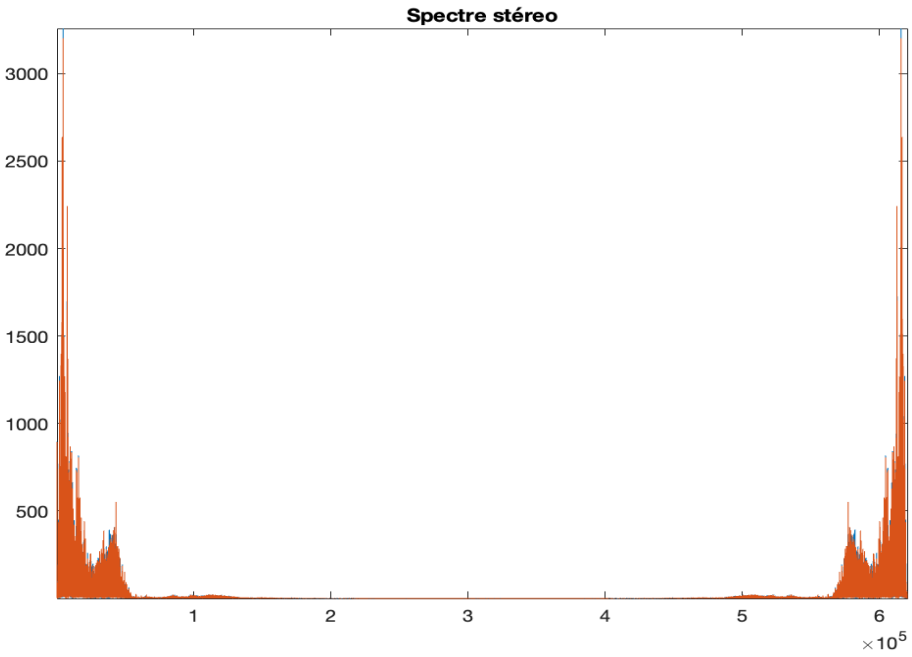

Figure 1-6 - Spectre du signal corrigé

1.2.3. Résultats et Commentaires

Les résultats de l'expérience et le graphe généré (figure 1-6) montrent les spectres de fréquence du signal original et du signal stéréo, permettant de visualiser les effets du décalage temporel appliqué.

Les résultats indiquent que le signal audio original a été transformé efficacement, avec le spectre de fréquence du signal stéréo montrant des variations par rapport au signal original. Ce filtrage et cette transformation permettent une meilleure analyse et application pratique des techniques de traitement du signal audio.

1.2.4. Code

```
clear all; close all

v=340;
d=0.25;
[x1,Fe]=audioread('Erlk.wav') ;
y1=fft(x1);
figure(1)
plot(abs(y1));
title("Spectre de x1");
axis tight;
exportfig(gcf,'x1','Format','png','color','cmyk');
L=length(x1);
x2=zeros(L,1);

%Construction de x2
for n=1:L
   theta=pi*(1/2-n/L);
   tau=d*sin(theta)/v;
   if(n-floor(tau*Fe)<=L && n-floor(tau*Fe)>0)
   x2(n)=x1(n-floor(tau*Fe));
   end
end

 y=[x1 x2];
 Stereo=fft(y);
 figure(2)
 plot(abs(Stereo));
 title("Spectre stéréo")
 axis tight;
 exportfig(gcf,'Spectre','Format','png','color','cmyk');
 audiowrite('resultat.wav',y,Fe) ;
 sound(y,Fe);
```

2. Projet 2

2.1. Sujet 1

Ce sujet porte sur la correction des décalages de lignes dans une image.
L'objectif est donc de réaligner correctement les lignes de l'image pour restaurer son apparence originale.

2.1.1. Observation et analyse de la situation

Dans cette section, nous avons procédé à l'observation et à l'analyse initiale du signal visuel contenu dans le fichier **fichier2.bmp**.
L'image observée (figure 2-1) stipule que l'image d'origine a été modifiée de manière que chaque ligne soit décalée de quelques pixels par rapport à la ligne précédente, créant une distorsion visuelle significative.
L'idée est donc , étant donné une ligne du signal visuel, de comparer les pixels de cette ligne avec la ligne suivante et regarder leurs ressemblances.
Cette notion suggère naturellement d'introduire l'intercorrélation entre ces lignes.

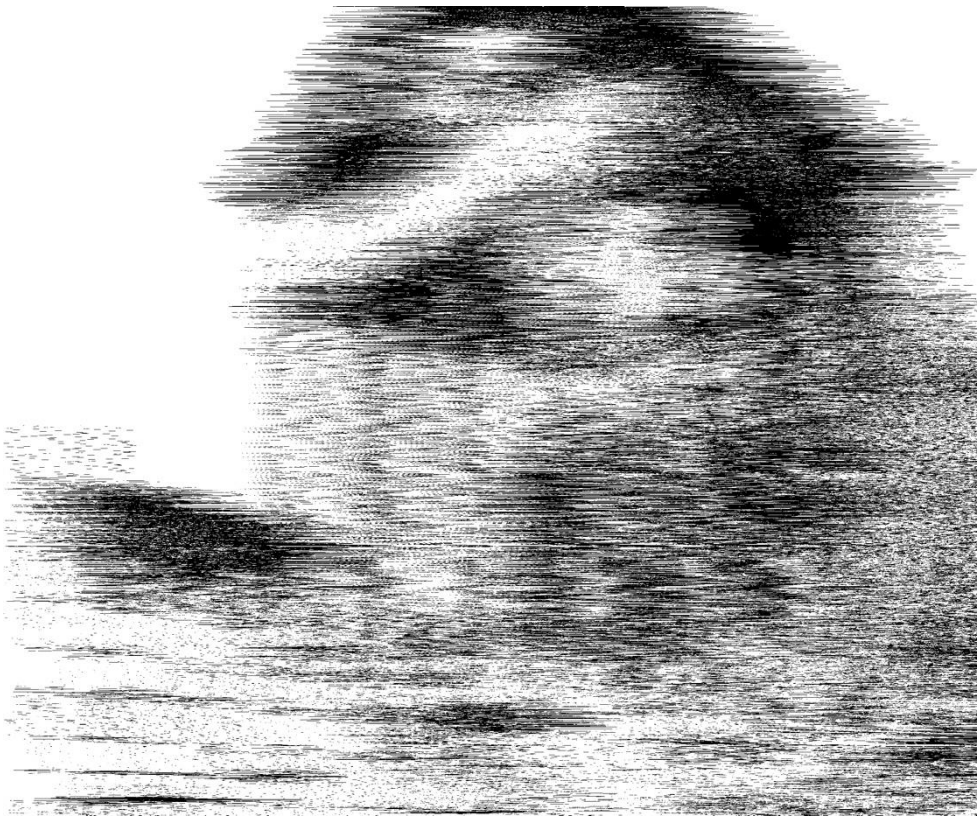

Figure 2-1 – Image corrompue

2.1.2. Méthodologie et Expérience

Pour corriger ces décalages, chaque ligne de l'image est comparée avec la suivante en utilisant l'intercorrélation entre ces deux lignes.
Le but est de trouver le décalage entre les lignes qui maximise la corrélation, et appliquer ce décalage pour réaligner la ligne suivante grâce à une permutation circulaire.

2.1.3. Résultats et Commentaires

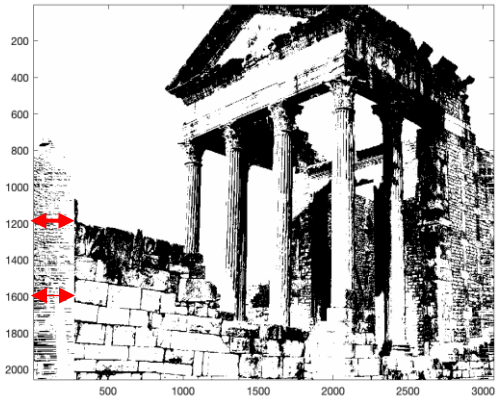

Figure 2-2 Image corrigée automatiquement

Figure 2-3 Image corrigée manuellement

L'image obtenue après la correction automatique (figure 2-2) montre une image nettement plus lisible en comparaison à l'image d'origine décalée.

Cependant, malgré un certain réalignement des lignes, l'image présente toujours un décalage global vers la droite.

On entreprend alors de faire un décalage manuel en se fiant au rendu visuel de l'image.

Le décalage manuel supplémentaire améliore l'alignement général de l'image, produisant une version plus correcte visuellement (figure 2-3)

Ainsi, la méthode d'intercorrélation est efficace pour corriger les décalages ligne par ligne, mais introduit un décalage accumulé qu'il faut ajuster manuellement.

L'utilisation de circshift est judicieuse pour réaligner les lignes, mais le choix du nombre de pixels pour le décalage manuel (275 pixels) est empirique et peut nécessiter des ajustements supplémentaires selon l'image originale.

2.1.4. Code

```
clear all; close all

clear B;
B=imread('fichier2.bmp','bmp');
B=255*B;
image(B);
colormap(gray);
[row_B,col_B ]=size(B);

for k=1:row_B-1
      ligne_1=B(k,:);
      ligne_2=B(k+1,:);
      correlation = xcorr( ligne_1, ligne_2 );
      [~, indice] = max (correlation) ;
      NO=circshift(ligne_2,indice);
      B(k+1,:)=NO;
end
figure(1)
image(B)
colormap(gray)
exportfig(gcf,'image_1_auto','Format','png','color','cmyk');

%L'image obtenue est un peu decaler par la droite ; on fait alors un nouveau
%decalage vers la gauche de façon retrouver une image meilleure
for k=1:row_B-1
   B(k,:)=circshift(B(k,:),-275);
end
figure(2)
image(B)
colormap(gray)
exportfig(gcf,'image_2_manuel','Format','png','color','cmyk');
```

2.2. Sujet 2

Dans le cadre du sujet 2, nous avons à traiter un signal audio nommé canal.wav.
Ce signal a été altéré en permutant ses hautes et basses fréquences via sa Transformée de Fourier (à l'exception des fréquences nulle et Fe/2).
L'objectif est de corriger cette modification en effectuant la permutation inverse, permettant ainsi de récupérer le signal audio original.

2.2.1. Observation et analyse de la situation

Dans cette section, nous avons procédé à l'observation et à l'analyse initiale du signal audio (figure 2-4) contenu dans le fichier **canal.wav**.
Cela permet de voir la forme d'onde et d'identifier des anomalies potentielles introduites par la permutation des fréquences.
Le spectre du signal (figure 2-5) est ensuite tracé pour observer les composantes fréquentielles présentes dans le signal.

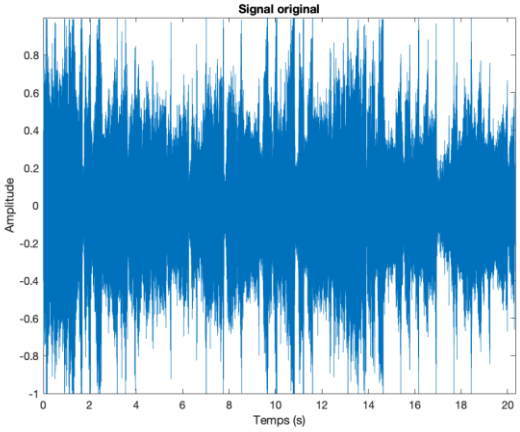

Figure 2-4 Signal original

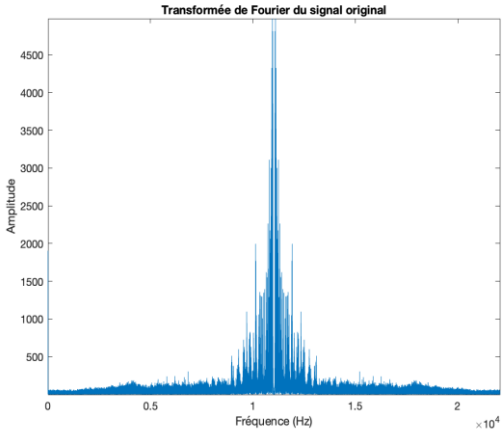

Figure 2-5 Spectre du signal original

2.2.2. Méthodologie et Expérience

La Transformée de Fourier du signal original est calculée et affichée, permettant de visualiser la distribution des fréquences dans le signal. (figure 2-5)
Les fréquences du signal sont ensuite permutées de manière inverse pour corriger la modification initiale : Le spectre de fréquence est modifié en permutant les hautes et basses fréquences pour revenir à l'état original. Cela implique de séparer le spectre en deux parties. La fonction flip est utilisée pour inverser les segments de fréquences de part et d'autre de Fe/2.
Après la permutation inverse, une Transformée de Fourier inverse (IFFT) est effectuée pour revenir au domaine temporel.

Le spectre du signal corrigé est obtenu et tracé pour une analyse visuelle. (figure 2-6)

2.2.3. Résultats et Commentaires

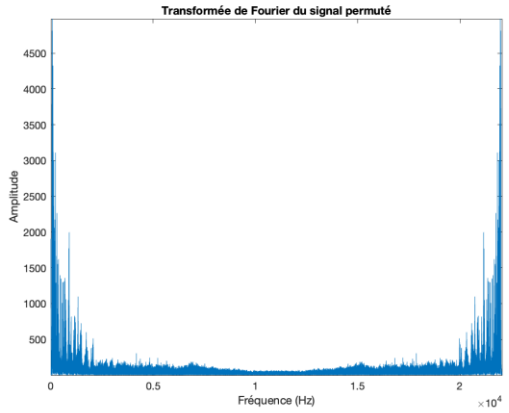

Figure 2-6 Spectre du signal permuté

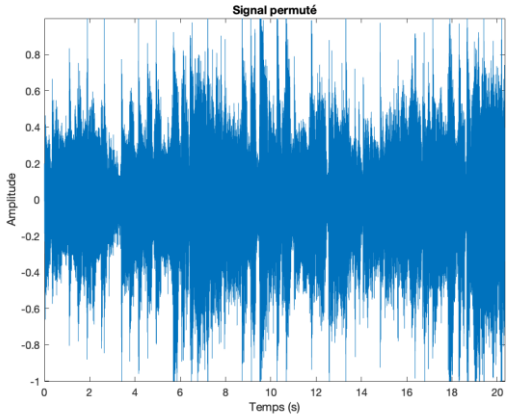

Figure 2-7 Signal permuté

Le signal corrigé, une fois reconstruit en domaine temporel, montre une forme d'onde plus régulière et conforme à ce qui serait attendu d'un signal non altéré.

Visuellement, il est possible de constater une amélioration notable par rapport au signal original. (figure 2-7)

L'écoute du signal corrigé confirme que la permutation inverse a été efficace. Le son est cohérent et ne présente plus les anomalies auditives dues à la permutation des fréquences.

2.2.4. Code

```
clear all; close all

%Chargement, lecture et √©chantillonnage du signal audio
filename = 'canal.wav';
[x, fe] = audioread(filename);
T = 1/fe ;
N=length(x);

% Affichage du signal original
t = (0:N-1)*T;
figure(1)
plot(t, x);
title('Signal original');
xlabel('Temps (s)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_1','Format','png','color','cmyk');

% Transform√©e de Fourier du signal original et affichage
TFx = fft(x) ;
f = linspace(0, (N-1)*fe/N, N );
figure(2)
plot(f, abs(TFx));
title('Transform√©e de Fourier du signal original');
xlabel('Fr√©quence (Hz)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_2','Format','png','color','cmyk');

%Permutation inverse du signal en separant en deux avant
et apres fe/2
TFx_gauche=TFx(2:N/2);
TFx_droite=TFx(N/2+2:N);

TFx_new=[TFx(1);flip(TFx_gauche);TFx(N/2+1);flip(TFx_droit
e)] ;
% flip est une fonction Matlab permettant d'inverser les
valeurs du signal ( comme si on lisait de droite a gauche )
```

```
% Affichage de la transform√©e de Fourier du signal
permut√©
figure(3)
plot(f, abs(TFx_new));
title('Transform√©e de Fourier du signal permut√©');
xlabel('Fr√©quence (Hz)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_3','Format','png','color','cmyk');

%Obtention du signal permut√©
x_new=ifft(TFx_new) ;

%Affichage du signal permut√©
figure(4)
plot(t, x_new);
title('Signal permut√©');
xlabel('Temps (s)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_4','Format','png','color','cmyk');

%Ecriture et jouer le signal final
audiowrite('canal_new.wav',x_new,fe);
sound(x_new, fe);
```

2.3. Sujet 3

Dans le sujet 3, nous devons analyser le signal audio encode.wav obtenu par une manipulation spécifique impliquant l'insertion de deux zéros entre chaque échantillon de deux signaux originaux. L'objectif est de retrouver les deux signaux originaux à partir de ce signal encodé.

2.3.1. Observation et analyse de la situation

Dans cette section, nous avons procédé à l'observation et à l'analyse initiale du signal audio contenu dans le fichier **encode.wav**.
Le signal audio original est tracé en fonction du temps pour une première visualisation de ses caractéristiques temporelles. (figure 2-9)
Cela permet d'identifier toute anomalie introduite par l'encodage au moyens des indices.
Le spectre de fréquence obtenu est tracé pour observer les amplitudes des différentes composantes fréquentielles du signal. (figure 2-8)

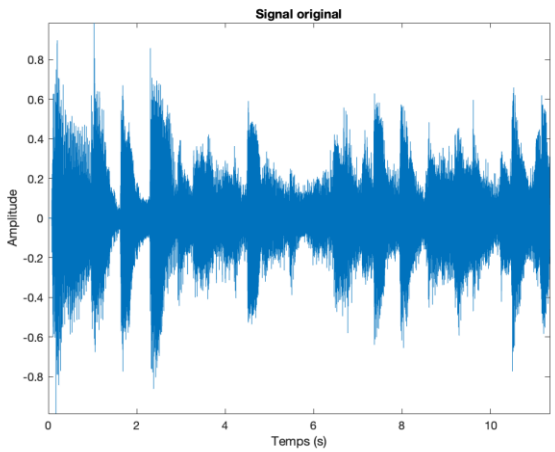

Figure 2-9 Signal original

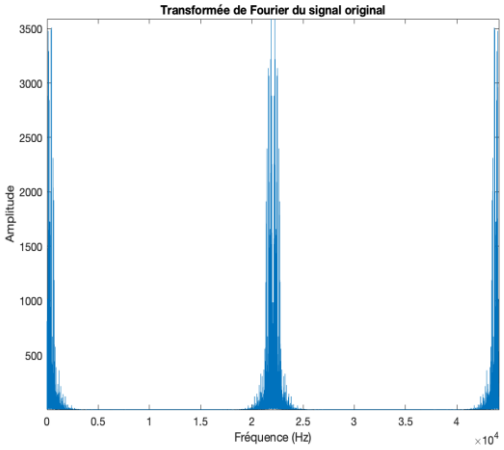

Figure 2-8 Spectre du signal original

2.3.2. Méthodologie et Expérience

Le signal est modifié en insérant deux zéros entre chaque échantillon d'origine, doublant ainsi la longueur du signal.
Cette étape permet de préparer le signal pour une analyse plus approfondie en fonction de l'indice donné dans le sujet.
La présence de motifs périodiques dans le spectre indique l'effet de l'encodage par zéros intercalés, justifiant la nécessité de la séparation des signaux basé sur la parité des échantillons. (figure 2-10)
Les Transformées de Fourier des deux signaux séparés sont calculées et tracées pour évaluer la récupération des signaux originaux. (figure 2-11 et figure 2-12)
Les signaux originaux sont enfin reconstruits et combinés pour créer un nouveau signal final représentant le signal utile.

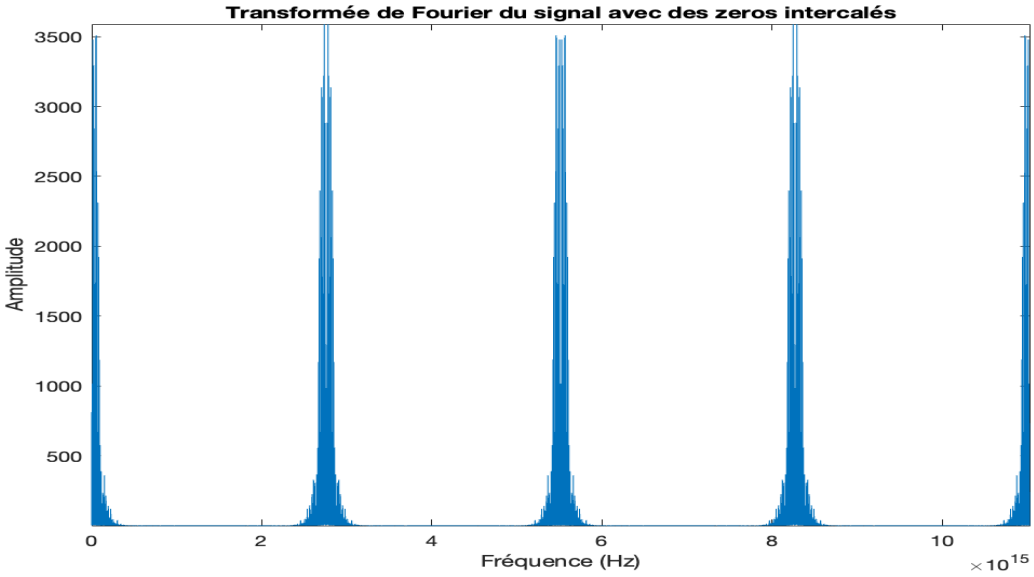

Figure 2-10 Spectre du signal avec zéros intercalés

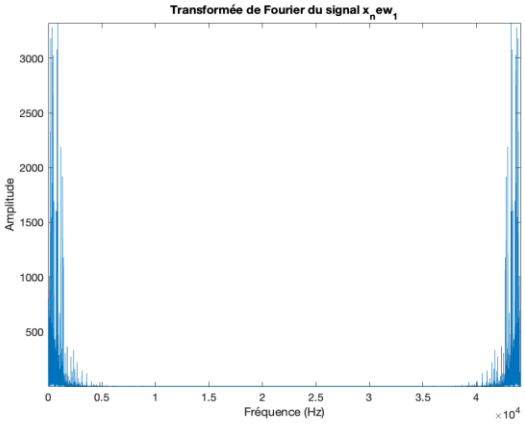

Figure 2-12 Spectre du signal séparé x1

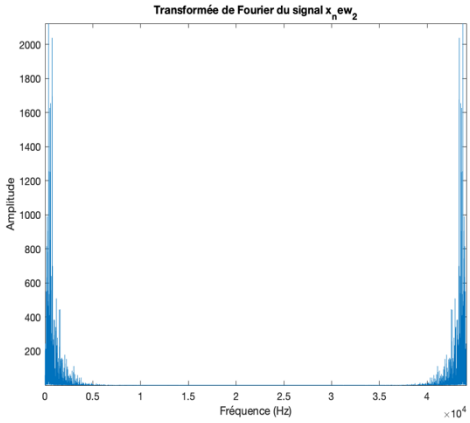

Figure 2-11 Spectre su signal séparé x2

2.3.3. Résultats et Commentaires

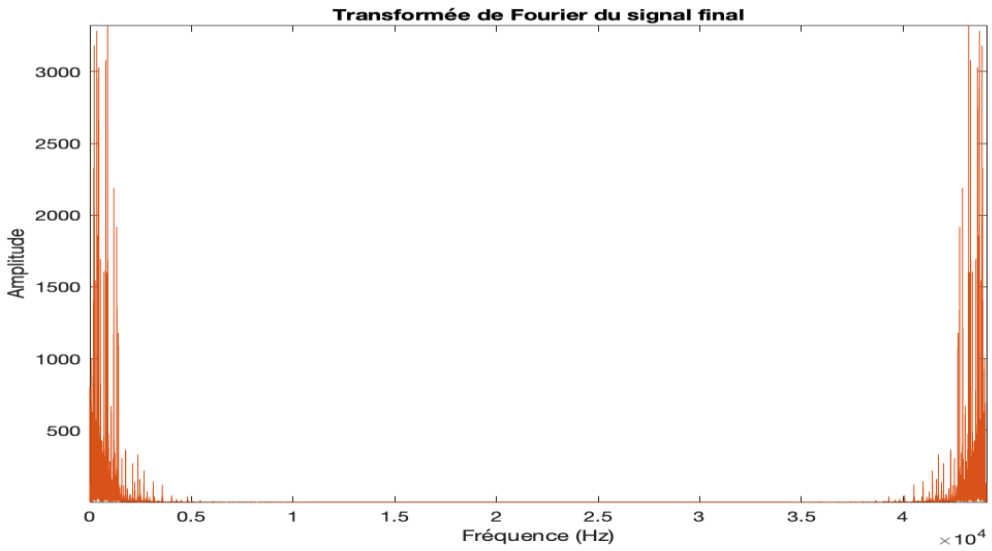

Figure 2-13 Spectre du signal corrigé

Les spectres de fréquence des deux signaux séparés (figure 2-11 et figure 2-12) montrent des caractéristiques distinctes. Cela confirme que la séparation a réussi à isoler les composantes fréquentielles d'origine.
La reconstruction du signal final à partir des deux signaux séparés montre une amélioration significative par rapport au signal encodé. (figure 2-13)
Enfin l'écoute du signal final confirme que la séparation et la reconstruction ont été efficaces.
Le son est cohérent et ne présente plus les anomalies introduites par l'encodage initial.

2.3.4 Code

```
clear all; close all

%Chargement, lecture et echantillonnage du signal audio
filename = 'encode.wav';
[x, fe] = audioread(filename);
T = 1/fe ;
N=length(x);

% Affichage du signal original
t = (0:N-1)*T;
figure(1)
plot(t, x);
title('Signal original');
xlabel('Temps (s)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_1','Format','png','color','cmyk');

% Transformee de Fourier du signal original et affichage
TFx = fft(x) ;
f = linspace(0, (N-1)*fe/N, N );
figure(2)
plot(f, abs(TFx));
title('TransformV©e de Fourier du signal original');
xlabel('FrV©quence (Hz)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_2','Format','png','color','cmyk');


%Completion par des zero
y=zeros(2*N,1);
for i=1:N
   y(2*i)=x(i);
end

% TransformV©e de Fourier du signal modifiV© et
affichage ( cf indice )
TFy = fft(y) ;
f = linspace(0, (2*N-1)*fe/2*N, 2*N );
figure(3)
plot(f, abs(TFy));
title('TransformV©e de Fourier du signal avec des zeros
intercalV©s');
xlabel('FrV©quence (Hz)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_3','Format','png','color','cmyk');
```

```
%SV©paration des deux signaux selon la paritV© aprV®s
visualisation
x_new_1=zeros(N/2,1);
x_new_2=zeros(N/2,1);
for i=1:N/2
   x_new_1(i)=x(2*i);
   x_new_2(i)=x(2*i-1);
end

%TransformV©e de Fourier des signaux obtenus apres
separation et affichage
%Celle de x_new_1
TFx_1=fft(x_new_1);
f = linspace(0, ((N/2)-1)*fe/(N/2), N/2 );
figure(4)
plot(f,abs(TFx_1));
title("TransformV©e de Fourier du signal x_new_1");
xlabel('FrV©quence (Hz)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_4','Format','png','color','cmyk');

%Celle de x_new_2
TFx_2=fft(x_new_2);
f = linspace(0, ((N/2)-1)*fe/(N/2), N/2 );
figure(5)
plot(f,abs(TFx_2));
title("TransformV©e de Fourier du signal x_new_2");
xlabel('FrV©quence (Hz)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_5','Format','png','color','cmyk');

%Ecriture affichage et jouer le signal final
x_new=[x_new_1,x_new_1];
TFx_new = fft(x_new) ;
f = linspace(0, (N-1)*fe/N, length( TFx_new));
figure(6)
plot(f, abs(TFx_new));
title('TransformV©e de Fourier du signal final');
xlabel('FrV©quence (Hz)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_6','Format','png','color','cmyk');

audiowrite('encode_new.wav',x_new,fe);
sound(x_new,fe);
```

3. Projet 3

3.1. Sujet 1

L'objectif de ce projet est de corriger l'écho dans un signal audio.
L'écho peut être modélisé comme la traversée d'un filtre avec une réponse impulsionnelle $H(Z)=1+\alpha Z^{-p0}$.
Cette considération signifie que le signal d'origine est sommé avec une version retardée et atténuée.
L'objectif est donc de retrouver le signal sans écho en identifiant les paramètres α et po du filtre H puis réaliser un filtre de la version retardée.

3.1.1. Observation et analyse de la situation

Dans cette section, nous avons procédé à l'observation et à l'analyse initiale du signal audio contenu dans le fichier **Pa11.wav**.
Le signal audio original est tracé en fonction du temps pour une première visualisation de ses caractéristiques temporelles. (figure 3-1)
Le calcul et le tracé de l'autocorrélation du signal sont ensuite réalisés pour observer les différents pics du signal et identifier les pics à filtrer.(figure 3-2)

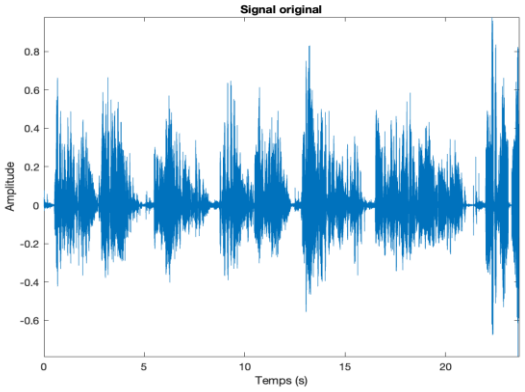

Figure 3-1 Signal original

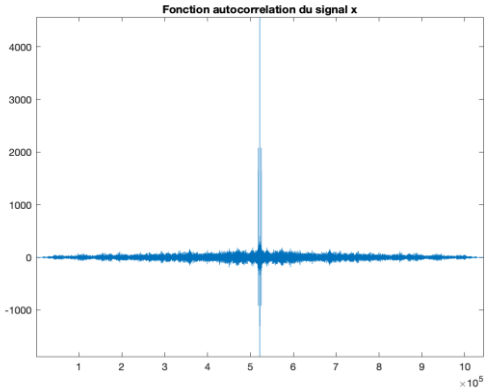

Figure 3-2 Spectre du signal original

3.1.2. Méthodologie et Expérience

La fonction d'autocorrélation r_{xx} du signal est calculée et tracée.

Cette fonction est cruciale pour identifier les retards p0 car elle met en évidence les similarités entre le signal et ses versions retardées.
La fonction d'autocorrélation r_{xx} montre un pic principal au décalage zéro, suivi de pics secondaires. (figure 3-2)
Le deuxième pic important, après élimination du pic principal, correspond au retard p0.
Le coefficient α est ensuite estimé en utilisant la relation entre les valeurs des pics de

La fonction d'autocorrélation rxx(p) est définie comme suit :

$$rxx(p) = E(x(n)x(n-p))$$

En développant cette expression, on obtient :

$$rxx(p) = \sigma_s^2 \delta_p (1 + \alpha^2) + \alpha \sigma_s^2 (\delta_{p+p0} + \delta_{p-p0})$$

Avec $\sigma_s^2 \delta_p$ représentant l'autocorrélation d'un signal blanc (puisque le signal s est supposé blanc, la DSP est constante). Ainsi, $rxx(p)$ est non nul pour $p = 0$, $p = p0$ et $p = -p0$.

Nous devrions donc observer trois pics sur le graphique de la fonction d'autocorrélation : le pic central correspondant à $p = 0$ et deux pics plus petits et symétriques correspondant à $p = p0$ et $p = -p0$. Sur le graphique de la fonction d'autocorrélation que nous avons tracé, ces pics sont bien présents.

Pour déterminer la valeur et les indices de ces pics, nous utilisons la fonction max sur des intervalles incluant les pics correspondants à $p = 0$ et $p = p0$ (ou $p = -p0$). Le pic le plus à droite étant à une distance $p0$ du pic central, la différence entre les deux abscisses nous donne $p0 = 3418 T_e = 155$ secondes.

Pour déterminer α, nous calculons le rapport entre la valeur du pic central et celle d'un des pics excentrés, noté r. Théoriquement, ce rapport est donné par :

$$r = \frac{1+\alpha^2}{\alpha}$$

Résolvons l'équation :

$$1 + \alpha^2 - r\alpha = 0$$

Après calcul, nous obtenons $\Delta > 0$, ce qui nous donne deux racines. Sachant que $\alpha < 1$, nous choisissons la seule racine qui vérifie cette condition :

$$\alpha = \frac{r - \sqrt{r^2 - 4}}{2} \approx 0,6472$$

l'autocorrélation.
Un filtre inverse est par la suite conçu pour éliminer l'écho du signal.
La réponse impulsionnelle du filtre est H=[1,zeros(1,p0-1),α].
Le signal filtré x_{new} est obtenu en appliquant ce filtre au signal original.
Théoriquement, les résultats sont calculés ainsi :

3.1.3. Résultats et Commentaires

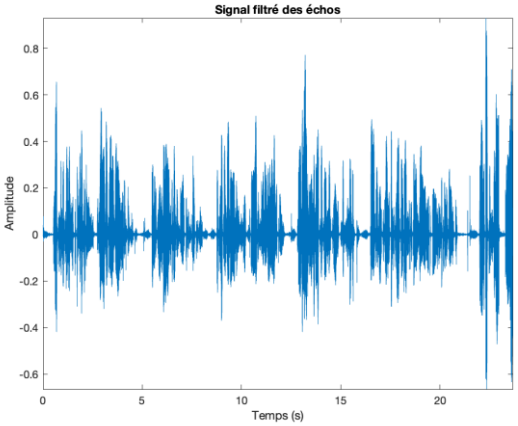

Figure 3-4 Signal filtré des échos

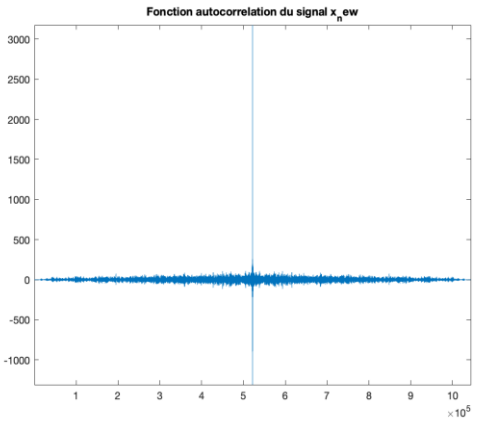

Figure 3-3 Spectre du signal filtré des échos

Le signal filtré est tracé pour visualiser l'effet du filtrage et vérifier la suppression de l'écho. (figure 3-3)
Ces figures montrent une réduction significative de l'écho.
La forme d'onde du signal après filtrage est plus claire et ne présente plus les répétitions caractéristiques de l'écho. (figure 3-4)
L'écoute du signal final confirme que l'écho a été efficacement supprimé.
Le son est plus net, ce qui indique que le filtrage a réussi à éliminer l'effet de l'écho sans distorsion notable.

3.1.4. Code

```
clear all; close all
%Chargement, lecture et √©chantillonnage du signal audio
filename = 'Pa11.wav';
[x, fe]=audioread(filename);
T = 1/fe ;
N=length(x);

% Affichage du signal original
t = (0:N-1)*T;
figure(1)
plot(t, x);
title('Signal original');
xlabel('Temps (s)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_1','Format','png','color','cmyk');

%Observation de l'autocorrelation du signal x
rxx=xcorr(x);
figure(2)
plot(rxx);
title("Fonction autocorrelation du signal x") ;
axis tight;
exportfig(gcf,'figure_2','Format','png','color','cmyk');


%Determination de to
[rx1,ind_1]=max(rxx);
rxx(ind_1-50:ind_1+100)=0;              % Elimination du pic principal pour
conserver que les seconds
[rx2,ind_2]=max(rxx);
p0=abs(ind_2-ind_1);

%Calcul du parametre alpha
alpha=0.5*(rx1/rx2-sqrt((rx1/rx2)^2-4));  % Formule vue en classe avec le chargv©
de TD

%Filtrage de l'echo dans le signal et affichage
H = [1 zeros(1,p0-1) alpha];
x_new=filter(1, H ,x);
figure(3)
plot(t, x_new);
title('Signal filtrv© des v©chos');
xlabel('Temps (s)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_3','Format','png','color','cmyk');


%Ecriture et lecture du signal final
audiowrite('Pa11_new.wav',x_new,fe);
sound(x_new, fe);
```

3.2. Sujet 2

Dans ce sujet, nous avons deux signaux audio qui sont des mélanges d'un signal utile et d'un bruit blanc.

Les signaux peuvent être modélisés par les équations suivantes :

$$x1(n)=a1\cdot s(n)+b1\cdot w(n). \qquad x2(n)=a2\cdot s(n)+b2\cdot w(n)$$

Les coefficients a1, a2, b1, et b2 sont tels que $b1 \gg a1$ et $b2 \gg a2$, indiquant que le bruit est prédominant dans les signaux.

L'objectif est d'utiliser une technique d'inversion de puissance pour combiner linéairement les deux signaux de manière à extraire le signal utile s(n).

3.2.1. Observation et analyse de la situation

Dans cette section, nous avons procédé à l'observation et à l'analyse initiale des signaux audio contenus dans les fichiers **x1.wav** et **x2.wav**
Les signaux audio originaux sont tracés en fonction du temps pour une première visualisation de ses caractéristiques temporelles. (figure 3-5 et figure 3-6)
Les variations rapides et irrégulières indiquent la présence significative de bruit dans les deux signaux.

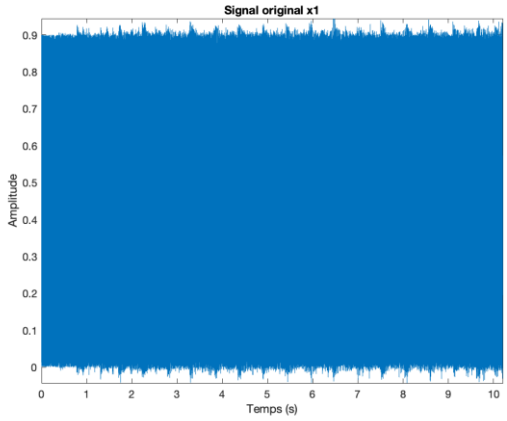

Figure 3-5 Signal x1

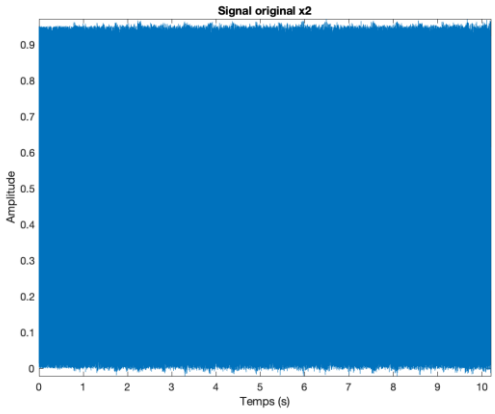

Figure 3-6 Signal x2

3.2.2. Méthodologie et Expérience

L'idée est de combiner linéairement les deux signaux au moyen d'un coefficient rho à travers une équation du type : epsilon(n) = x1(n) – rho*x2(n).
Il faut donc minimiser quadratiquement epsilon pour un rho optimal.
Le coefficient ρ est déterminé en utilisant la méthode de Gram-Schmidt : comme la moyenne du produit des deux signaux divisée par la moyenne du carré du deuxième signal.
Cela a permis de minimiser l'énergie du signal résultant ϵ(n).
On trace ensuite l'intercorrélation entre le signal résultant ϵ(n) et le signal x2(n) pour vérifier la décorrélation entre ϵ(n) et x2(n). (figure 3-7)

3.2.3. Résultats et Commentaires

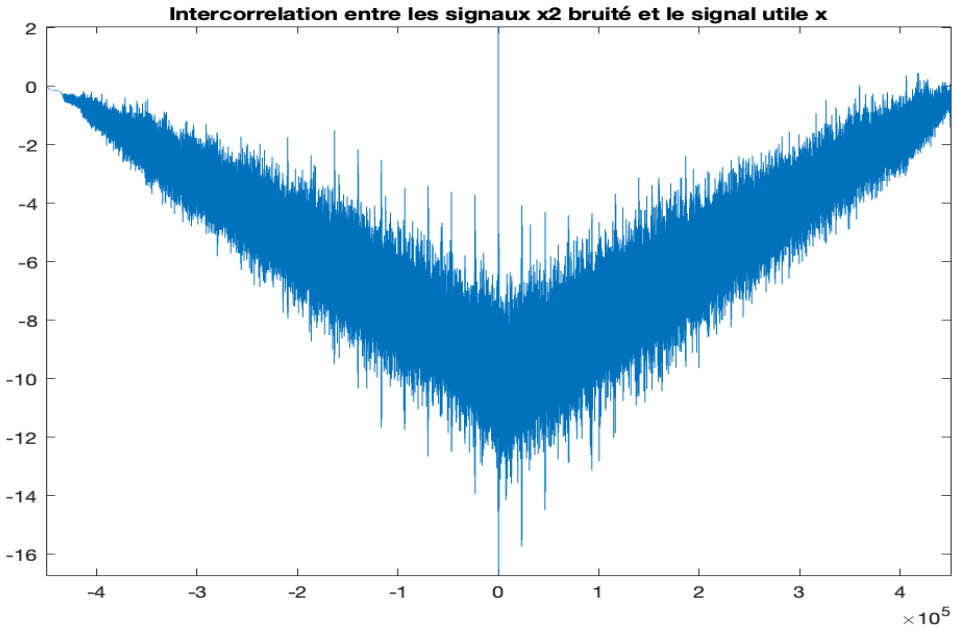

Figure 3-7 Intercorrélation des signaux x2 et le signal utile ϵ

Le signal ϵ(n) obtenu après la combinaison linéaire a montré une réduction significative du bruit par rapport aux signaux originaux.
L'analyse en plus de l'intercorrélation a confirmé que le signal ϵ(n) est bien décorrélé du signal x2(n). (figure 3-7). Cela signifie que le bruit a été efficacement supprimé, laissant principalement le signal utile.
Enfin, l'écoute du signal final ϵ(n) a montré une amélioration notable en termes de clarté et de réduction du bruit. Le signal utile est devenu plus discernable par rapport aux signaux d'origine. Ces démarches montrent l'efficacité de l'inversion de puissance pour extraire le signal utile.

3.2.4. Code

```
clear all; close all

%Chargement, lecture et √©chantillonnage des signaux audios
filename_1 = 'x1.wav';
[x1, fe_1]=audioread(filename_1);
T1 = 1/fe_1 ;
N1=length(x1);

filename_2 = 'x2.wav';
[x2, fe_2]=audioread(filename_2);
T2 = 1/fe_2 ;
N2=length(x1);



% Affichage des signaux originaux

t1 = (0:N1-1)*T1;    % Affichage x1
figure(1)
plot(t1, x1);
title('Signal original x1');
xlabel('Temps (s)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_1','Format','png','color','cmyk');

t2 = (0:N2-1)*T2;        % Affichage x2
figure(2)
plot(t2, x2);
title('Signal original x2');
xlabel('Temps (s)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_2','Format','png','color','cmyk');

%Determination de ro par Graam-Scmidt ( Sous impulsion du chargv© de TD )
ro=mean(x1.*x2)/mean(x2.^2);

%Le signal x recherche est une combinaison lineaire de x1 et x2 :

x=x1-ro*x2;

r_x_x1=xcorr(x,x2);
n=length(r_x_x1);
f=linspace(-n/2,n/2,n);
figure(3)
plot(f,r_x_x1) ;
title('Intercorrelation entre les signaux x2 bruitv© et le signal utile x');
axis tight;
exportfig(gcf,'figure_3','Format','png','color','cmyk');

%Ecriture et lecture du signal final
audiowrite('x_new.wav',x,fe_1);
sound(x,fe_1)
```

3.3. Sujet 3

Le but de cet exercice est de filtrer un signal audio contenant deux fréquences sinusoïdales en utilisant un filtre de prédiction linéaire (LPC).
Un signal sinusoïdal peut être considéré comme un signal autorégressif d'ordre 2.
Ainsi, un filtre autorégressif d'ordre 4 est donc capable de prédire et d'annuler deux fréquences sinusoïdales présentes dans le signal.

3.3.1. Observation et analyse de la situation

Dans cette section, nous avons procédé à l'observation et à l'analyse initiale du signal audio contenus dans le fichier **Mo11.wav**
Le signal audio original est tracé en fonction du temps pour une première visualisation de ses caractéristiques temporelles. (figure 3-8)

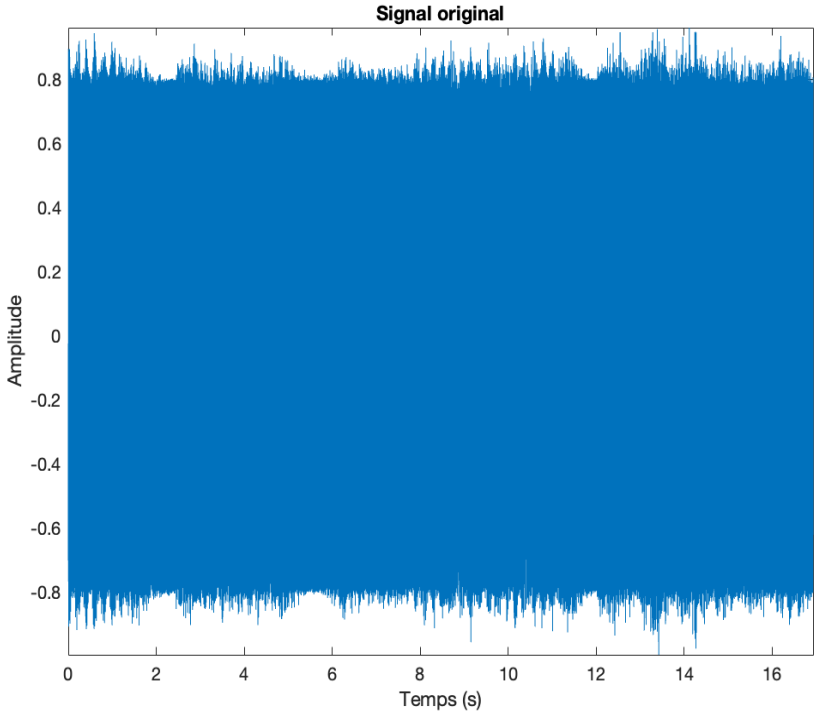

Figure 3-8 Signal original

3.3.2. Méthodologie et Expérience

Un filtre autorégressif d'ordre 4 est conçu pour prédire et annuler les deux fréquences sinusoïdales présentes dans le signal. Les coefficients du filtre sont obtenus en utilisant la fonction lpc de MATLAB.
Le signal est filtré à l'aide des coefficients du filtre pour supprimer les deux fréquences sinusoïdales.

La Transformée de Fourier (FFT) du signal filtré est calculée et tracée pour observer l'effet du filtrage sur le spectre fréquentiel. (figure 3-9)
Cette analyse permet de vérifier l'efficacité du filtre à éliminer les fréquences indésirables.

3.3.3. Résultats et Commentaires

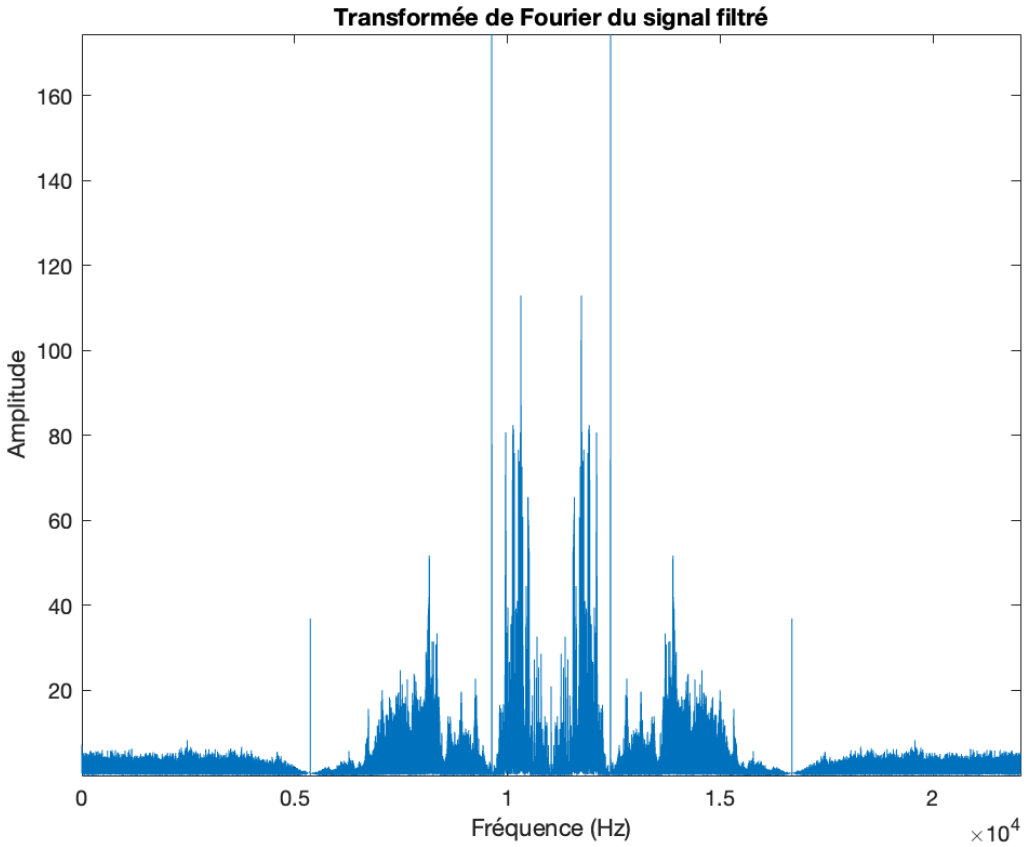

Figure 3-9 Spectre du signal filtré

Le spectre fréquentiel du signal filtré montre une réduction significative des amplitudes aux fréquences correspondant aux sinusoïdes initiales.
Cela confirme que le filtre a efficacement supprimé les fréquences indésirables.
Le tracé de la Transformée de Fourier du signal filtré valide visuellement l'efficacité du filtrage. (figure 3-9)
L'écoute du signal filtré confirme que les fréquences sinusoïdales ont été supprimées. Le signal résultant est nettement différent de l'original, avec une réduction des composantes sinusoïdales prédominantes.

3.3.4. Code

```
clear all; close all

%Chargement, lecture et √©chantillonnage du signal audio
filename = 'Mo11.wav';
[x, fe]=audioread(filename);
T = 1/fe ;
N=length(x);

% Affichage du signal original
t = (0:N-1)*T;
figure(1)
plot(t, x);
title('Signal original');
xlabel('Temps (s)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_1','Format','png','color','cmyk');

%Pour un signal contenant deux fr√©quences sinusoÝdales, un filtre autoregressif d'ordre 4  est n√©cessaire pour les pr√©dire et les annuler

coeffs = lpc(x, 4);            %%% FONCTION TROUVER EN LIGNE  PUIS SUR MATLAB AVEC HELP

% Filtrer le signal
x_new = filter(coeffs, 1, x);

% Transform√©e de Fourier du signal filtr√© et affichage
TFx_new = ifftshift(fft(x_new)) ;
f = linspace(0, (N-1)*fe/N, N );
figure(2)
plot(f, abs(TFx_new));
title('Transform√©e de Fourier du signal filtr√©');
xlabel('Fr√©quence (Hz)');
ylabel('Amplitude');
axis tight;
exportfig(gcf,'figure_2','Format','png','color','cmyk');

%Ecriture et lecture du signal final
audiowrite('Mo11_new.wav',x_new,fe);
sound(x_new, fe);
```

Conclusion

Au cours des différents projets réalisés, nous avons exploré plusieurs techniques de traitement du signal, notamment la transformation de Fourier, l'utilisation de filtres autorégressifs, et les manipulations temporelles et fréquentielles des signaux. Ces projets nous ont permis d'approfondir notre compréhension des concepts fondamentaux en traitement du signal et de les appliquer à des cas pratiques.

Nous avons aligné les lignes d'une image en utilisant la corrélation croisée pour détecter et corriger les décalages horizontaux, filtrer des signaux bruités au moyen des techniques d'inversion de puissance, de filtre RII et de prédiction linéaire. Ensuite nous avons créé un effet stéréo à partit d'un signal mono, filtré un son d'un écho et enfin réaliser l'inversion de fréquence sur un échantillon.

En somme, les compétences acquises et les connaissances approfondies lors de ces projets constituent une base solide pour des travaux futurs en traitement du signal, offrant une variété d'outils pour résoudre des problèmes complexes dans ce domaine.

Pour notre part, ce fut un projet passionnant où l'on a eu à mettre en pratique ces connaissances théoriques de façon significative.