Université Paul Sabatier
Toulouse

Compte Rendu

Traitement numérique du signal

M1 EEA

- Module KEAX7AH1 : Traitement numérique du signal
- Réalisé par :
  - KABOU Abdeldjalil
  - IAICHE Achour Mehdi Anis

2024 / 2025

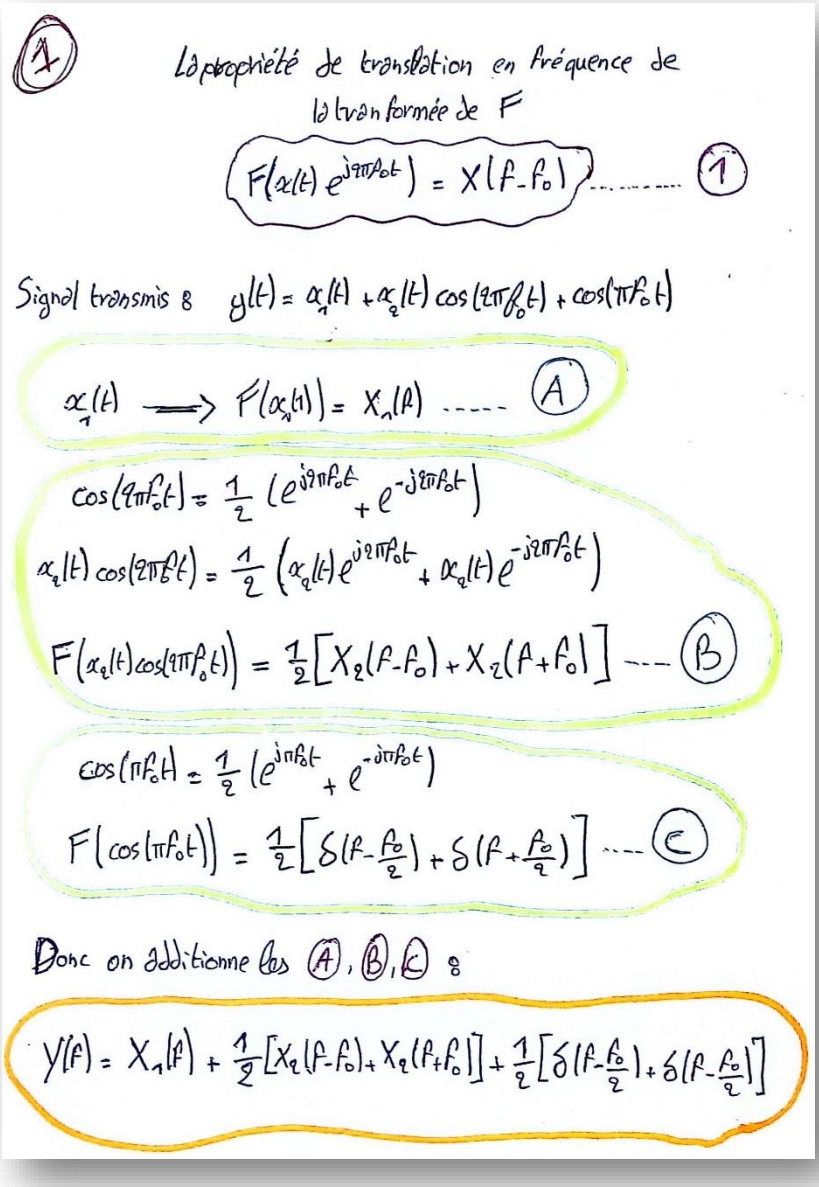
TP2 : Réalisation d'un démodulateur stéréo

I BUT DE LA MANIPULATION

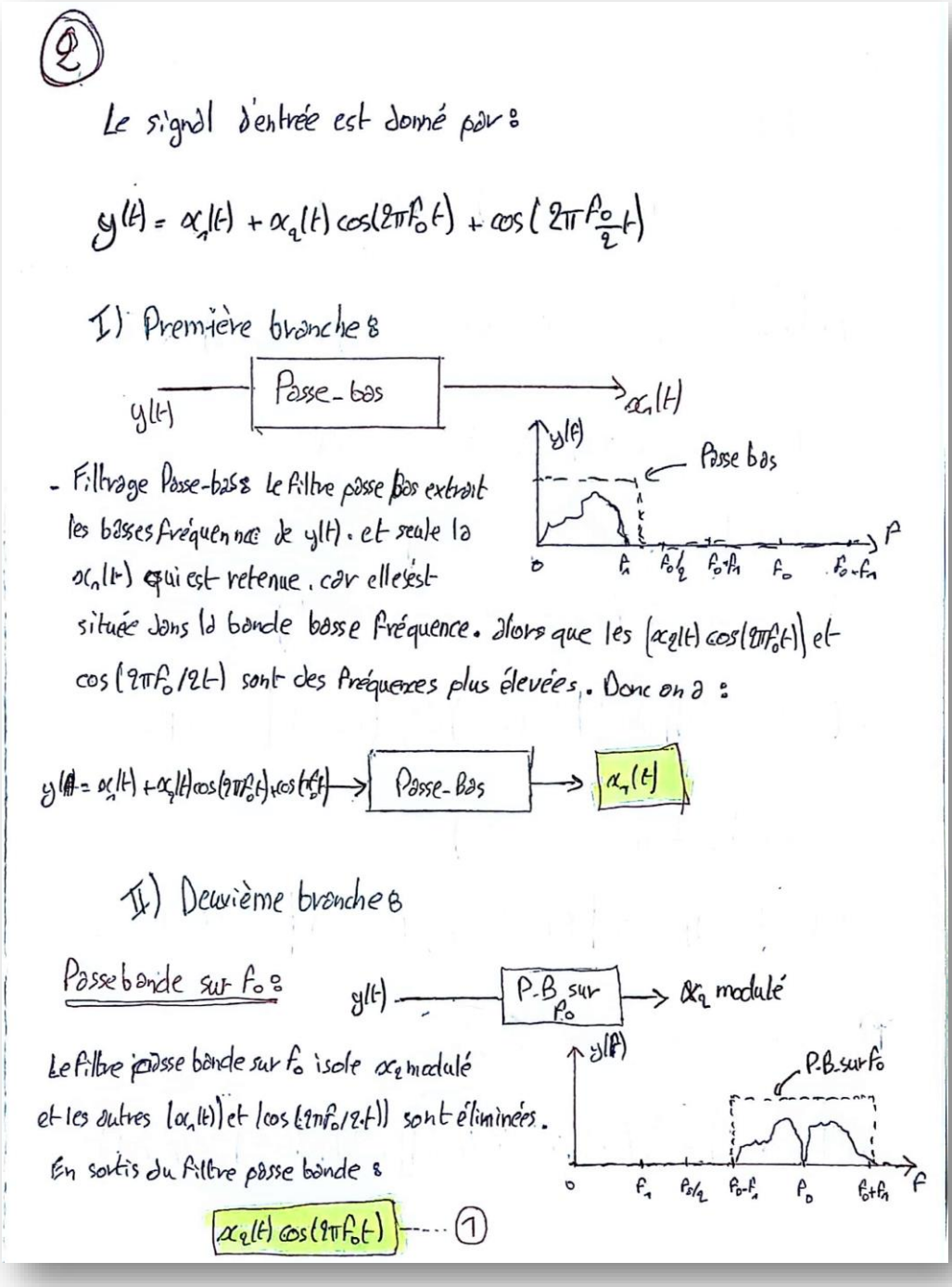
L'objectif de ce TP est de restaurer, dans les récepteurs, les signaux x1(t) et x2(t) à partir du signal transmis y(t). Une fois cette opération effectuée, un récepteur monophonique pourra alimenter son haut-parleur avec le signal x1(t)=g(t)+d(t), tandis qu'un récepteur stéréophonique pourra reproduire les signaux g(t) et d(t) à partir des signaux x1(t) et x2(t)..

II SCHEMA DU DEMODULATEUR

1. *La transformée de Fourier théorique du signal transmis y(t) en fonction des transformées de Fourier des signaux x1(t) et x2(t) :*



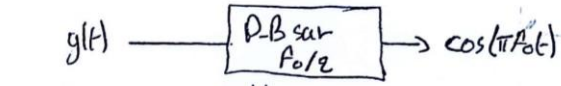
2. *Démonstration théorique que le système arrive à restaurer les signaux x1 et x2 en sortie :*



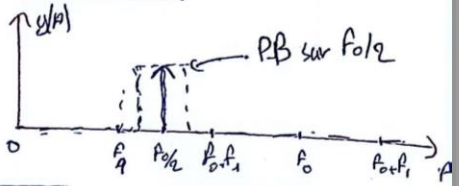
(3)

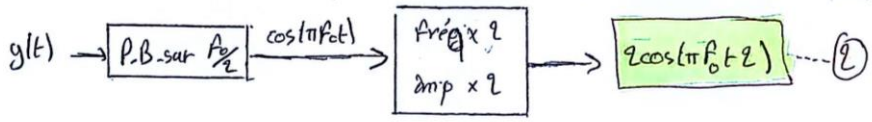
III) troisième branche :

Passe bande sur $f_0/2$:



le filtre passe bande isole uniquement la $\cos(\pi f_0 t)$ car elle est centrée sur la fréquence $f_0/2$, et les autres sont éliminées.





Maintenant on fait la multiplication de la deuxième partie et la troixième.

(1) x (2)

$V(t) = \alpha_2 \text{ modulé} \times 2\cos(2\pi f_0 t)$

$V(t) = \alpha_2(t)\cos(2\pi f_0 t) \times 2\cos(2\pi f_0 t)$

On a : $\cos(A) \times \cos(B) = \frac{1}{2}[\cos(A+B) + \cos(A-B)]$

Donc :

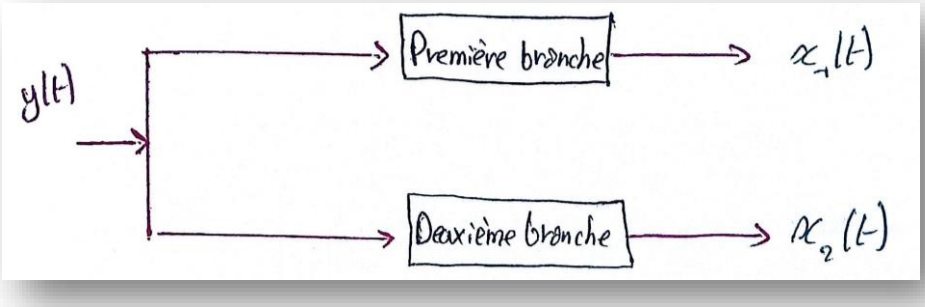
$$V(t) = \alpha_2(t)[\cos(4\pi f_0 t) + 1]$$

$$\boxed{V(t) = \alpha_2(t) + \alpha_2(t)\cos(4\pi f_0 t)}$$

Donc à la fin on applique le filtre passe-bas sur $V(t)$:

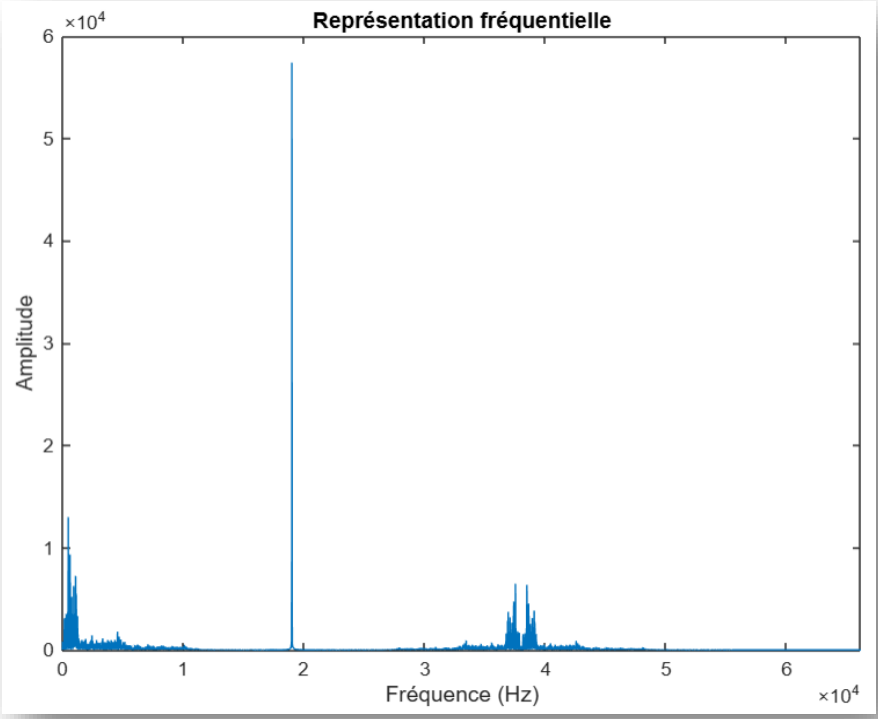
il élimine les hautes fréquence $(\alpha_2(t)\cos(4\pi f_0 t))$.

Donc :

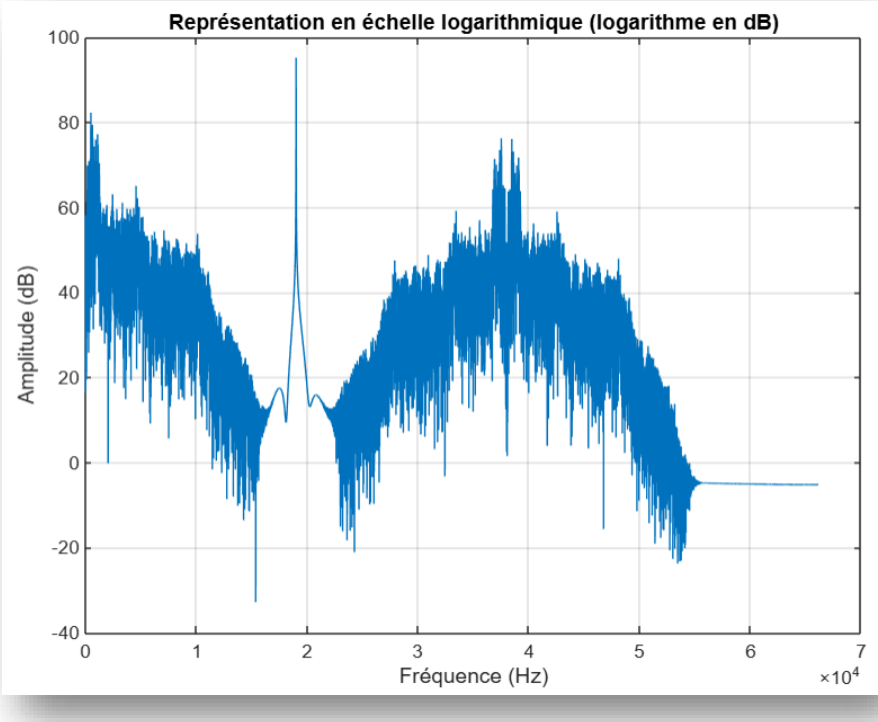


III ANALYSE DU SIGNAL

1. Traçage de la représentation fréquentielle du signal.



Pour une représentation en échelle logarithmique (logarithme en dB) :



*Le code est ci-dessous dans les annexes

On peut observer trois pics distincts correspondant aux composantes x1, x2 modulé, et la porteuse. Cela signifie que vous avez bien identifié les fréquences des composantes du signal :

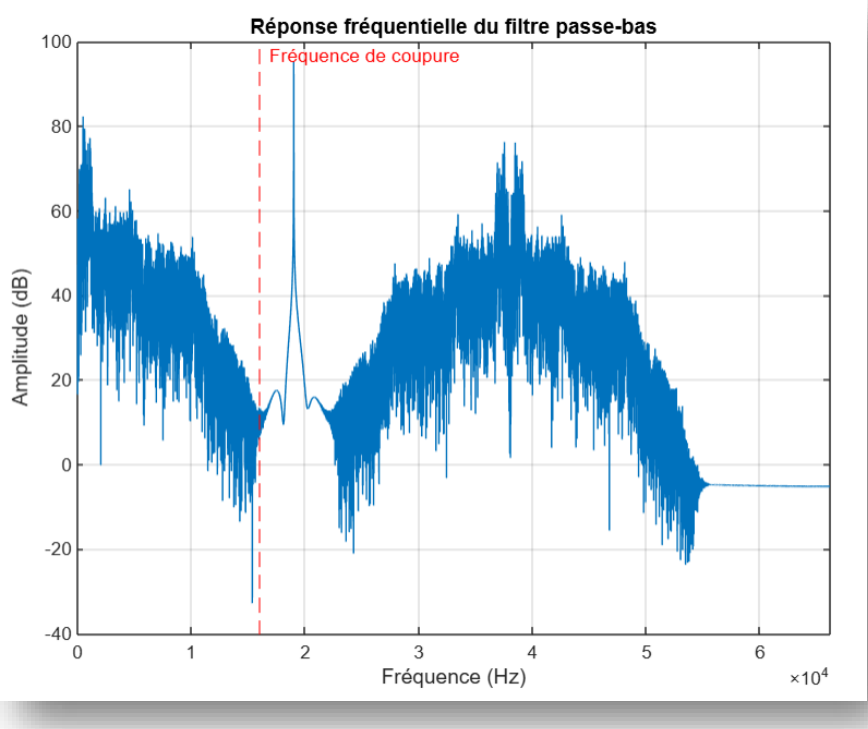
- f1= 14 000 Hz correspond à la composante basse fréquence (x1(t)).
- f0= 38 000 Hz est la fréquence de la porteuse utilisée pour moduler x2(t).

2. Les gabarits des 4 filtres utilisés dans le démodulateur.

a) Filtre passe-bas (première branche pour x1(t)) :

Gabarit :

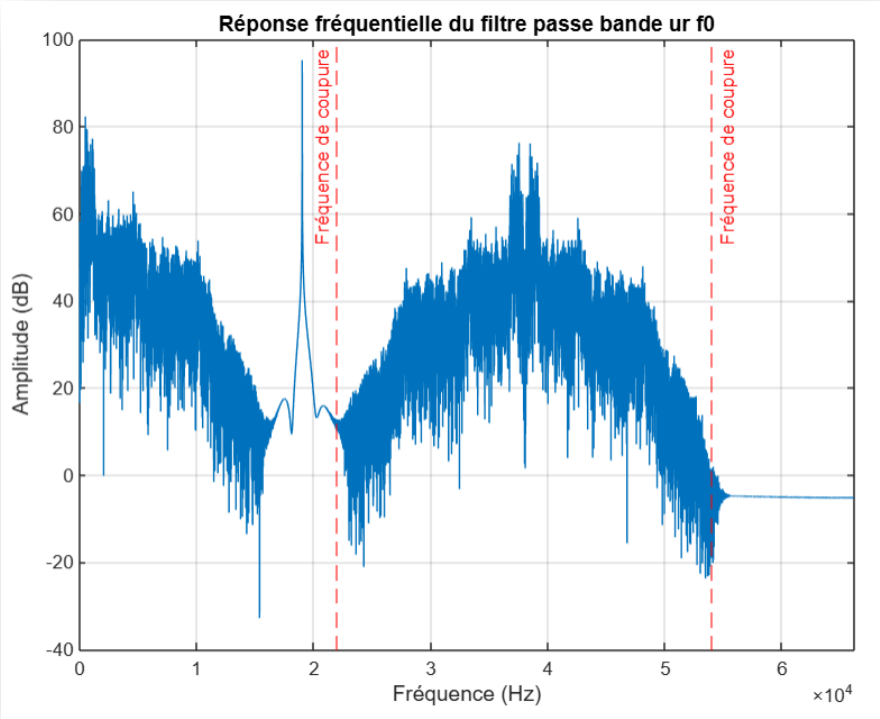
- Bande passante : [0, f1] donc [0 kHz à 14 kHz].
- Bande de transition : [f1, f1+2KHz] donc [14 kHz à 16 kHz]
- Bande coupée : > f1+2KHz donc 16 kHz.
- Amplitude : 82 dB



b) Filtre passe-bande sur f0 (deuxième branche : pour x2(t) modulé) :

Gabarit :

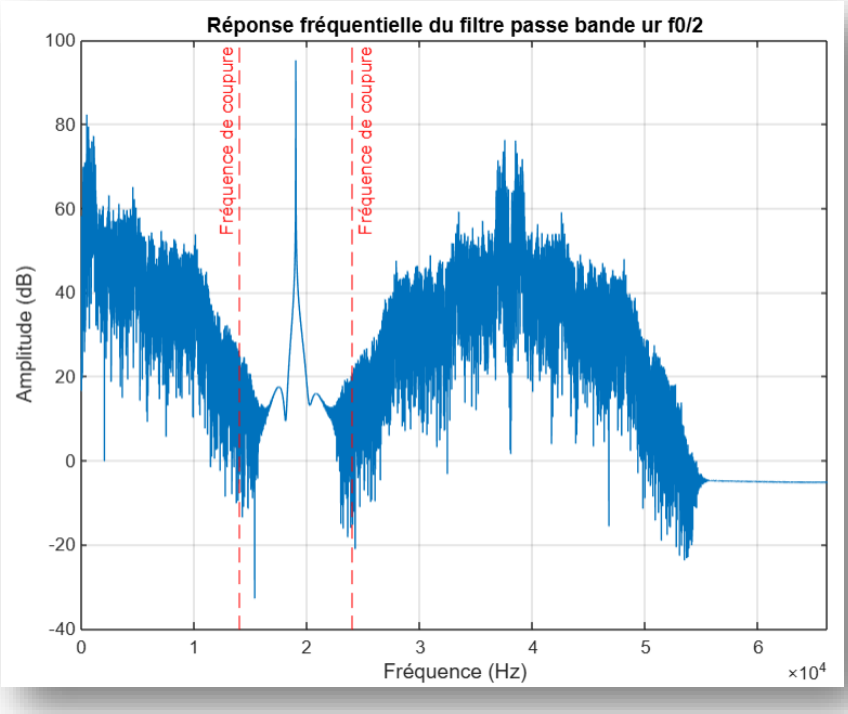
- Bande passante : [f0-f1, f0+f1] donc [24 kHz à 52 kHz].
- Bande de transition : [f0-f1-2KHz, f0-f1] et [f0+f1, f0+f1+2KHz] donc [22 kHz à 24 kHz] et [52 kHz à 54 kHz]
- Bande coupée : fc < [f0-f1-2Khz] et fc > [f0+f1+2Khz] donc fc < 22 kHz et fc > 54 kHz.



c) Filtre passe-bande sur f0/2 (troisième branche : pour cos (2πf0/2 t)) :

Gabarit :

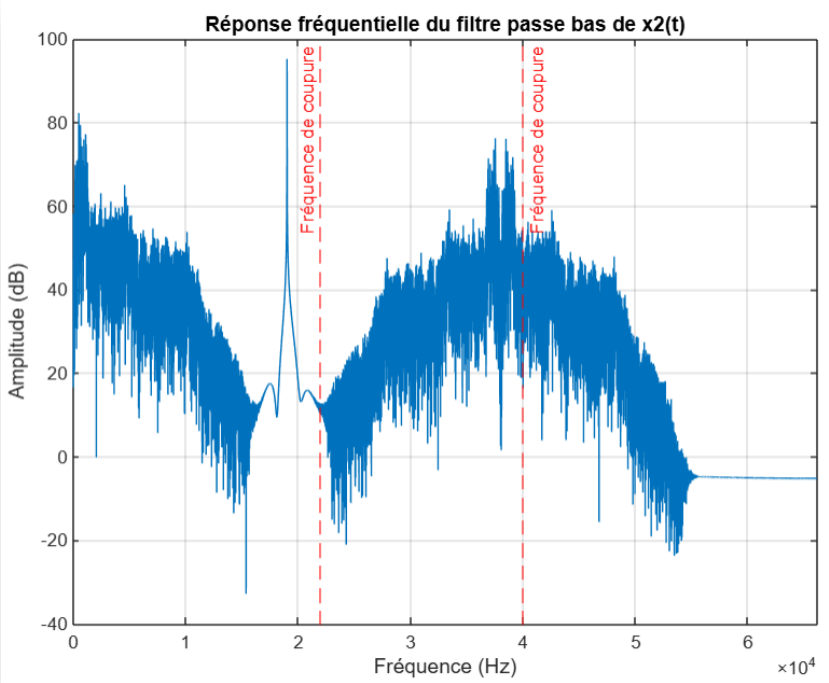
- Bande passante : [f1+2 kHz, f0-f1-2KHz] donc [16 kHz à 22 kHz].
- Bande de transition : [f1, f1+2 kHz] et [f0-f1-2KHz, f0-f1] donc [14 kHz à 16 kHz] et [22 kHz à 24 kHz]
- Bande coupée : fc < f1 et fc > f0-f1 donc fc < 14 kHz et fc > 24 kHz
- Amplitude : 76 dB



d) Filtre passe-bas (dernière branche : pour x2(t))

Gabarit :

- Bande passante : [f0-f1, f0] donc [24 kHz à 38 kHz].
- Bande de transition : [f0-f1-2KHz, f0-f1] et [f0, f0+2KHz] donc [22 kHz à 24 kHz] et [38 kHz à 40 kHz]
- Bande coupée : fc < [f0-f1-2Khz] et fc > [f0+2Khz] donc fc < 22 kHz et fc > 24 kHz.
- Amplitude : 75 dB



*Le code est ci-dessous dans les annexes

IV RECUPERATION DU SIGNAL X1(t)

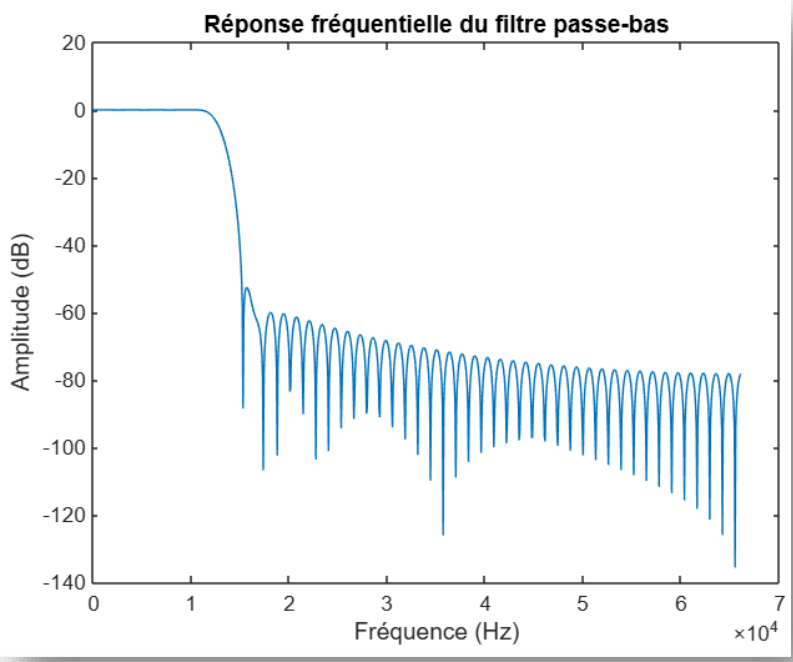
L'objectif est de récupérer x1(t), un signal contenu dans y(t), en le filtrant à l'aide d'un filtre passe-bas.

1. *Synthèse du filtre passe-bas avec fir1 :*

Le filtre doit respecter le gabarit donné :

- Fréquence de coupure fc.
- Atténuation en bande passante ≤1 dB.
- Atténuation en bande coupée ≥40 dB.

- Bande de transition large de 2 kHz.

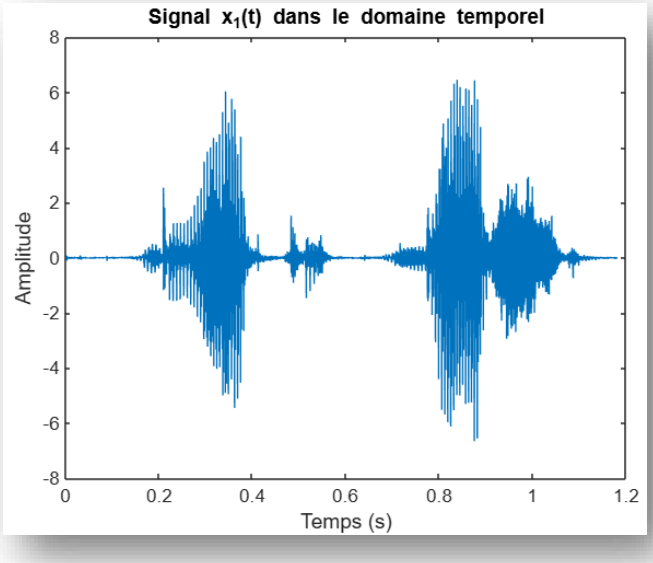


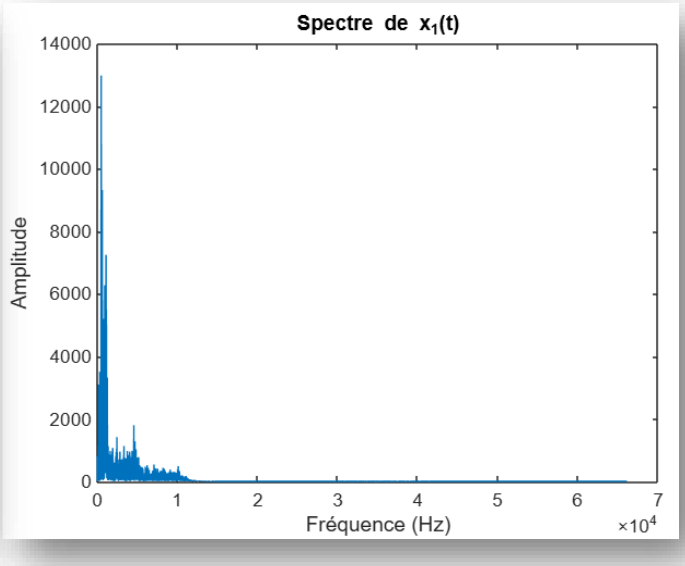
2. *Calcul du retard induit par le filtre*

   - Le retard d'un filtre RIF est donné par Retard= ordre / 2fe.
   - Par exemple, ici on a choisi n=100 et on a fe=132300 Hz :

     Le retard du filtre est *0.000378* secondes.

3. *Représentation temporelle et fréquentielle :*



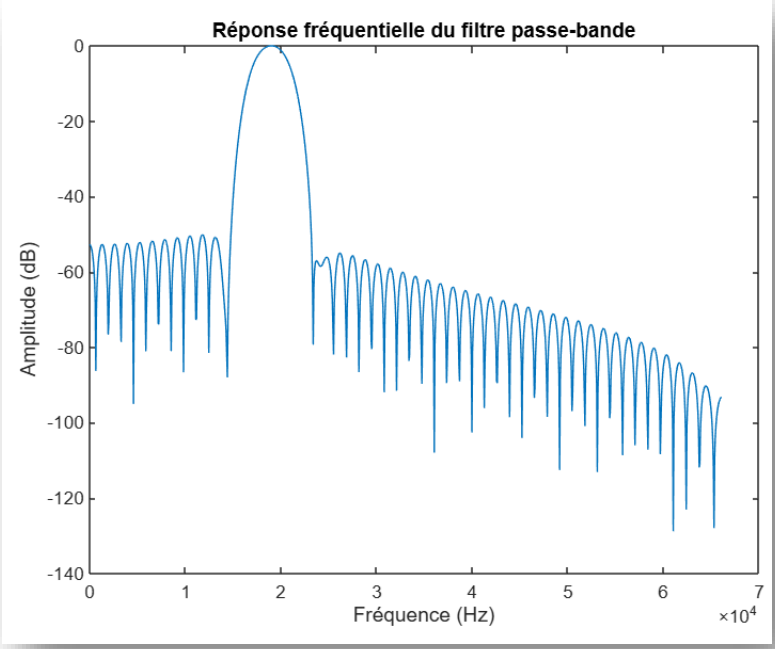


4. *Ecoute du signal filtré : nous entendons une phrase filtrée "droite gauche"*

*Le code est ci-dessous dans les annexes

V RECUPERATION DE LA PORTEUSE

1. *Synthèse du filtre passe bande :*

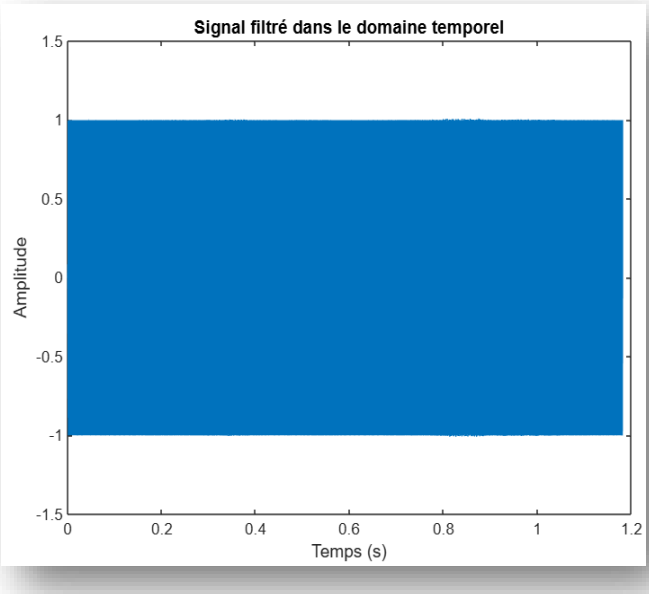


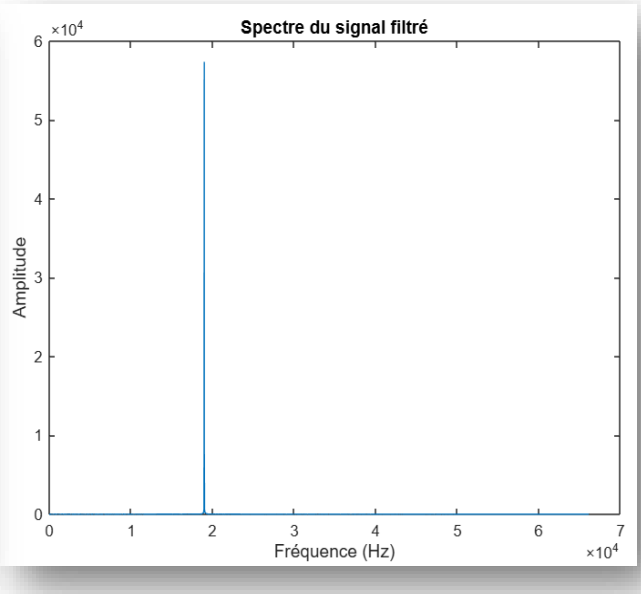
Calcul du retard induit par le filtre

- Le retard d'un filtre RIF est donné par Retard= ordre / 2fe.
- Par exemple, ici on a choisi n=100 et on a fe=132300 Hz :

  Le retard du filtre est *0.000378* secondes.

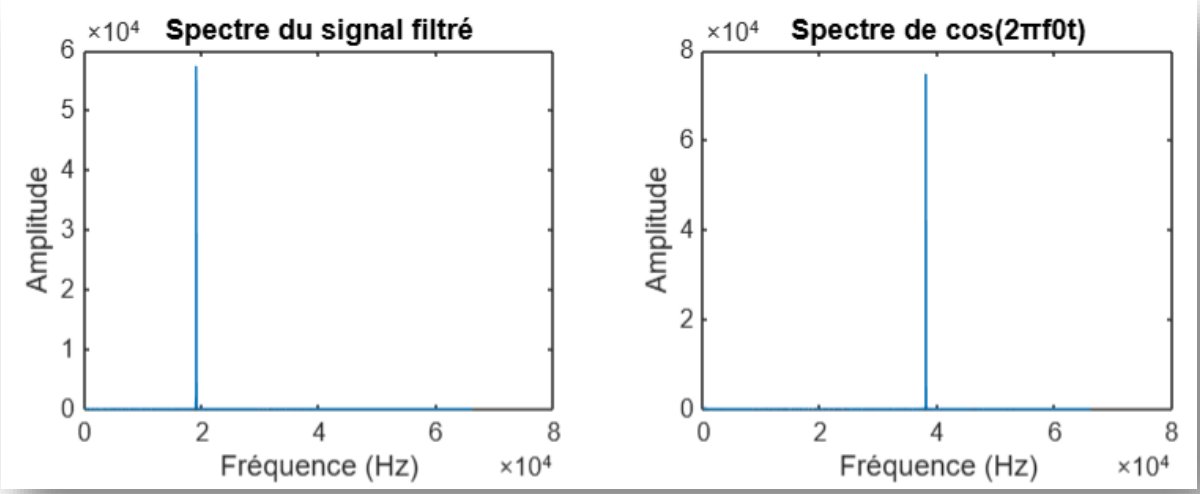
2. *Représentation temporelle et fréquentielle :*





3. *Proposition d'une méthode pour doubler sa fréquence et obtenir ainsi cos(2πf0t) :*

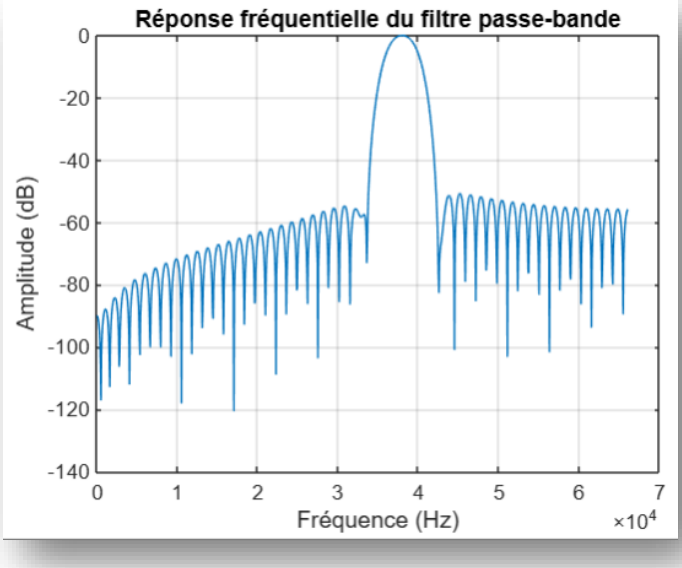
On a cos(2πf0t/2), donc on peut élever le signal au carré et on utilise un filtre passe-haut pour isoler la composante cos(2πf0t), on peut atteindre l'objectif



*Le code est ci-dessous dans les annexes

VI RECUPERATION DU SIGNAL X2(t)

1. *Synthèse du filtre passe bande centrée sur f0 :*

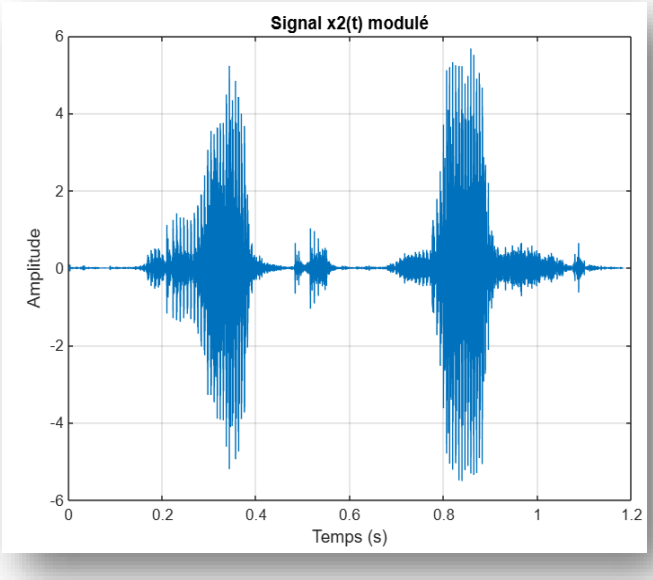


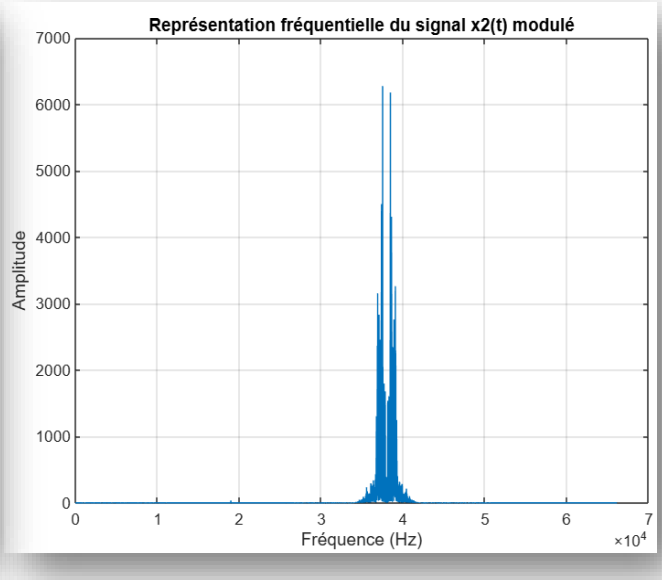
2. *Calcul du retard induit par le filtre*

- Le retard d'un filtre RIF est donné par Retard= ordre / 2fe.
- Par exemple, ici on a choisi n=100 et on a fe=132300 Hz :

Le retard du filtre est *0.000378* secondes.

3. *Représentation temporelle et fréquentielle du signal $X_2(t)$ modulé :*

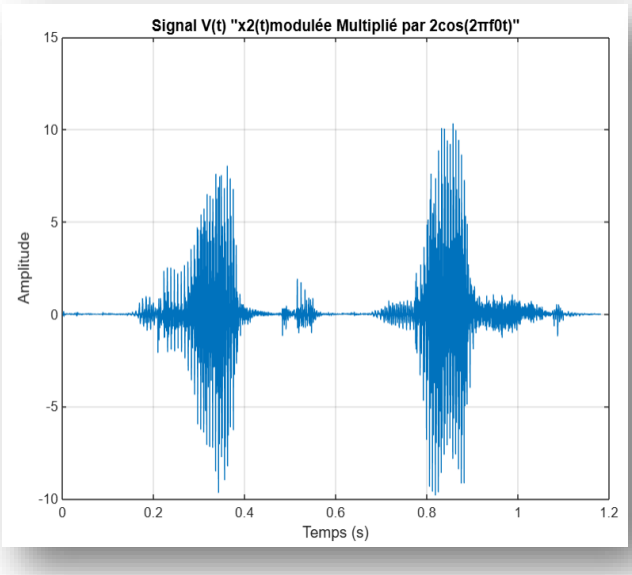


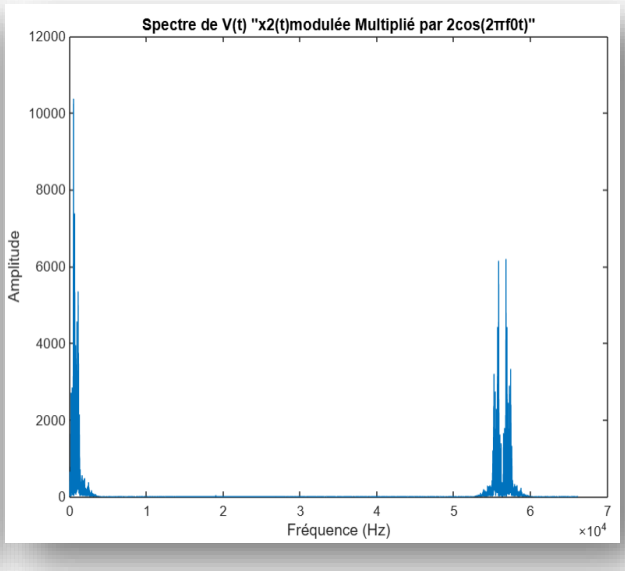


3. *Pour ramener le signal x2(t) :*

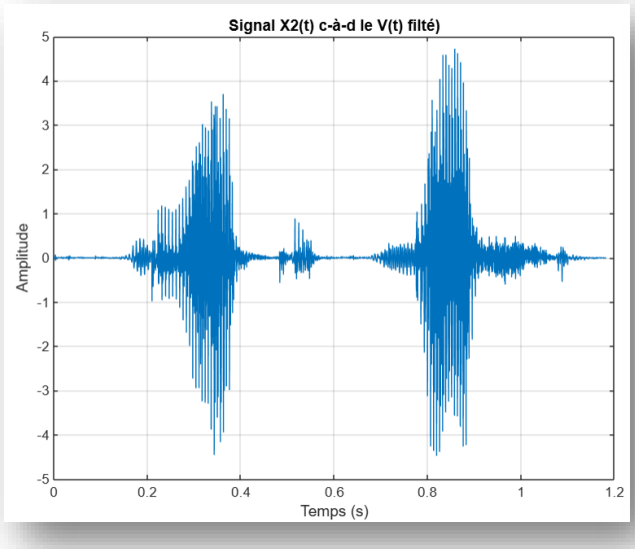
- Multiplication du signal filtré par 2cos(2πf0t) :
- *Calcul du retard induit par le filtre*
  - Le retard d'un filtre RIF est donné par Retard= ordre / 2fe.
  - Par exemple, ici on a choisi n=100 et on a fe=132300 Hz :

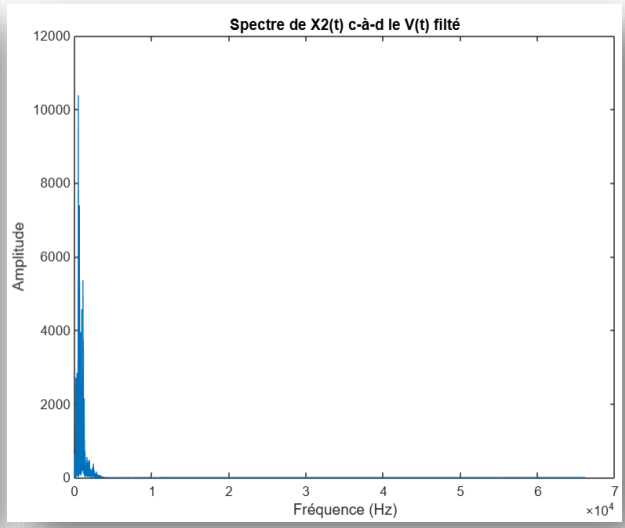
    Le retard du filtre est *0.000378* secondes.
- *Représentation temporelle et fréquentielle de Signal x2(t) modulée Multiplié par 2cos(2πf0t), le signal obtenu noté v(t) :*





- *Filtrage du résultat par un passe-bas pour obtenir le signal X2(t) (v(t) filtré) :*



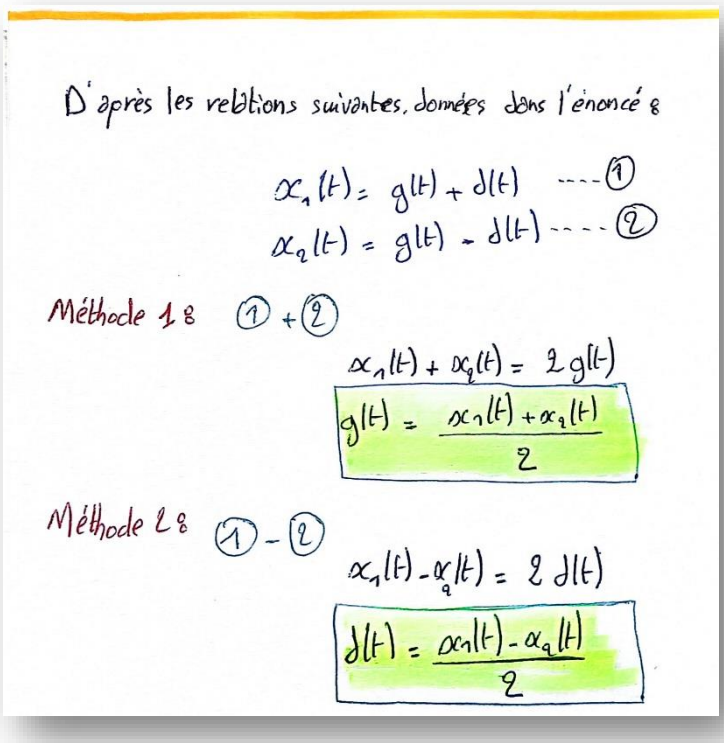


*En écoutant x2(t) : on attend l*a même phrase que x1(t), mais avec un rythme ou une clarté légèrement différente (*x2(t) est un peu claire que x1(t)*), en raison des transformations et filtrages appliqués.

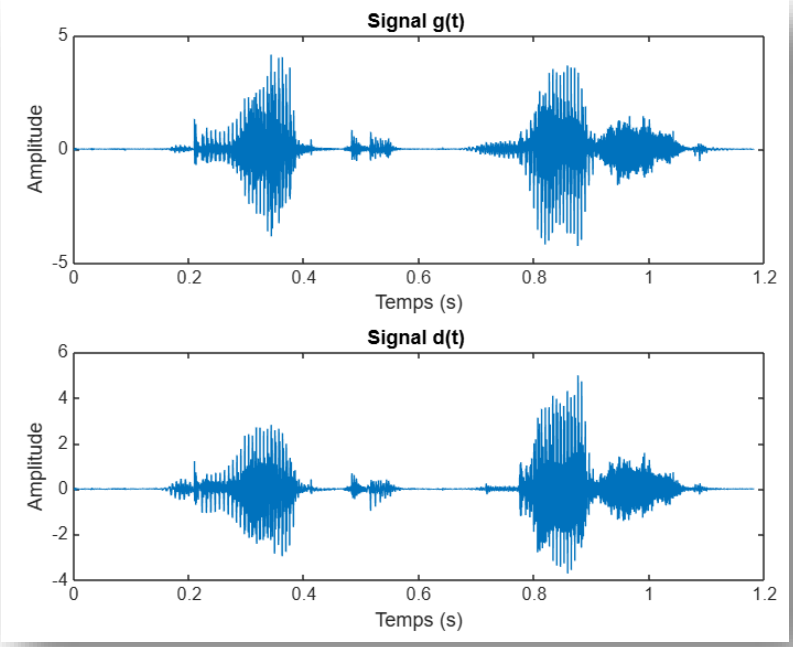
*Le code est ci-dessous dans les annexes

VII RECUPERATION DES SIGNAUX g(t) ET d(t)

1. Méthode pour retrouver g(t) et d(t) à partir de x1(t) et x2(t) :



2. *Reconstruire les signaux g(t) et d(t) avec notre méthode*



*Le code est ci-dessous dans les annexes

- *Résultats attendus :*

  En écoutant g(t) : on entend clairement "gauche".
  En écoutant d(t) : on entend clairement "droite".

3. *Retards induits par les filtres :*

Quand un filtre numérique traite un signal, il introduit un retard temporel. Ce retard provient de la nature du filtre, en particulier de son ordre.

Pour que x1(t) et x2(t) soient correctement alignés dans le temps, il faut compenser les retards induits par les filtres :

- On calcule le retard de chaque filtre
- On trouve le retard maximum (le filtre qui induit le plus grand retard)
- On synchronise les signaux

Annexes

I ANALYSE DU SIGNAL

Traçage de la représentation fréquentielle du signal.

```
load('Multi.mat');          % Charger le fichier
signal=importdata('Multi.mat');
Fe=132300;
N = length(signal);         % Taille du signal
f = (0:N-1)*(Fe/N);         % Axe des fréquences
Y = abs(fft(signal));       % la FFT
% Tracer la représentation fréquentielle
figure;
plot(f, Y);
xlabel('Fréquence (Hz)');
ylabel('Amplitude');
title('Représentation fréquentielle');
xlim([0 Fe/2]); % Restreindre à la moitié de la bande
            % pour une meilleure lisibilité
figure;
plot(f(1:N/2),20*log10(abs(Y(1:N/2))));
grid on;
xlabel('Fréquence (Hz)');
ylabel('Amplitude (dB)');
title('Représentation en échelle logarithmique (logarithme en dB)');
xlim([0 Fe/2]); % Restreindre à la moitié de la bande
            % pour une meilleure lisibilité
```

Les gabarits des 4 filtres utilisés dans le démodulateur

```
load('Multi.mat');          % Charger le fichier
signal=importdata('Multi.mat');
Fe=132300;
N = length(signal);         % Taille du signal
f = (0:N-1)*(Fe/N);         % Axe des fréquences
Y = abs(fft(signal));       % la FFT
% Tracer la représentation fréquentielle
figure;
plot(f, Y);
xlabel('Fréquence (Hz)');
ylabel('Amplitude');
title('Représentation fréquentielle');
xlim([0 Fe/2]); % Restreindre à la moitié de la bande
            % pour une meilleure lisibilité
figure;
plot(f(1:N/2),20*log10(abs(Y(1:N/2))));
```

```
grid on;
xlabel('Fréquence (Hz)');
ylabel('Amplitude (dB)');
title('Représentation en échelle logarithmique (logarithme en dB)');
xlim([0 Fe/2]); % Restreindre à la moitié de la bande
        % pour une meilleure lisibilité

%filtre passe-bas
figure;
plot(f(1:N/2),20*log10(abs(Y(1:N/2))));
grid on;
xlabel('Fréquence (Hz)');
ylabel('Amplitude (dB)');
%title('Représentation en échelle logarithmique (logarithme en dB)');
xlim([0 Fe/2]); % Restreindre à la moitié de la bande
        % pour une meilleure lisibilité
title('Réponse fréquentielle du filtre passe-bas');
% Annotations pour les bandes
hold on;
xline(16000, '--r', 'Fréquence de coupure','LabelOrientation','horizontal'); % Fréquence de coupure

%filtre passe bande ur f0
figure;
plot(f(1:N/2),20*log10(abs(Y(1:N/2))));
grid on;
xlabel('Fréquence (Hz)');
ylabel('Amplitude (dB)');
%title('Représentation en échelle logarithmique (logarithme en dB)');
xlim([0 Fe/2]); % Restreindre à la moitié de la bande
        % pour une meilleure lisibilité
title('Réponse fréquentielle du filtre passe bande ur f0');
% Annotations pour les bandes
hold on;
xline(22000, '--r', 'Fréquence de coupure','LabelHorizontalAlignment', 'left'); % Fréquence de coupure
hold on;
xline(54000,'--r','Fréquence de coupure','LabelHorizontalAlignment', 'right'); % Fréquence de coupure

%filtre passe bande ur f0/2
figure;
plot(f(1:N/2),20*log10(abs(Y(1:N/2))));
grid on;
xlabel('Fréquence (Hz)');
ylabel('Amplitude (dB)');
%title('Représentation en échelle logarithmique (logarithme en dB)');
xlim([0 Fe/2]); % Restreindre à la moitié de la bande
        % pour une meilleure lisibilité
title('Réponse fréquentielle du filtre passe bande ur f0/2');
% Annotations pour les bandes
hold on;
```

```
xline(14000, '--r', 'Fréquence de coupure','LabelHorizontalAlignment', 'left'); % Fréquence de coupure
hold on;
xline(24000,'--r','Fréquence de coupure','LabelHorizontalAlignment', 'right'); % Fréquence de coupure

%filtre passe bas de x2(t)
figure;
plot(f(1:N/2),20*log10(abs(Y(1:N/2))));
grid on;
xlabel('Fréquence (Hz)');
ylabel('Amplitude (dB)');
%title('Représentation en échelle logarithmique (logarithme en dB)');
xlim([0 Fe/2]); % Restreindre à la moitié de la bande
        % pour une meilleure lisibilité
title('Réponse fréquentielle du filtre passe bas de x2(t)');
% Annotations pour les bandes
hold on;
xline(22000, '--r', 'Fréquence de coupure','LabelHorizontalAlignment', 'left'); % Fréquence de coupure
hold on;
xline(40000,'--r','Fréquence de coupure','LabelHorizontalAlignment', 'right'); % Fréquence de coupure
```

II RECUPERATION DU SIGNAL X1(T)

```
function wavplay(signal, sampleRate)
% WAVPLAY  Plays an audio signal using the audioplayer object.
%   WAVPLAY(x, Fs) plays the audio signal x at the specified sample rate Fs.

   % Instantiate object to play audio data.
   player = audioplayer(signal, sampleRate);

   % Play signal from beginning to end (no overlap with other code possible).
   playblocking(player);
end

load('Multi.mat');          % Charger le fichier
signal=importdata('Multi.mat');
fe = 132300;          % Fréquence d'échantillonnage
fc = 13000;          % Fréquence de coupure (Hz)
Wc = fc / (fe / 2);   % Fréquence de coupure normalisée
n = 100;             % Ordre initial du filtre (ajustez par essais)
h = fir1(n, Wc, 'low', hamming(n+1)); % Filtre passe-bas
[H, freq] = freqz(h, 1, 1024, fe);
amplitude_dB = 20 * log10(abs(H));
figure;
plot(freq, amplitude_dB);
```

```
xlabel('Fréquence (Hz)'); ylabel('Amplitude (dB)');
title('Réponse fréquentielle du filtre passe-bas');
delay = n / (2 * fe);
fprintf('Le retard du filtre est %.6f secondes.\n', delay);
x1 = filter(h, 1, signal);
t = (0:length(x1)-1) / fe;
figure;
plot(t, x1);
xlabel('Temps (s)'); ylabel('Amplitude');
title('Signal x_1(t) dans le domaine temporel');
N = length(x1);
X1 = fft(x1);
f = (0:N-1) * (fe / N);
figure;
plot(f(1:N/2), abs(X1(1:N/2)));
xlabel('Fréquence (Hz)'); ylabel('Amplitude');
title('Spectre de x_1(t)');
wavplay(x1, fe)
```

III RECUPERATION DE LA PORTEUSE

```
function wavplay(signal, sampleRate)
% WAVPLAY  Plays an audio signal using the audioplayer object.
%   WAVPLAY(x, Fs) plays the audio signal x at the specified sample rate Fs.
   % Instantiate object to play audio data.
   player = audioplayer(signal, sampleRate);
   % Play signal from beginning to end (no overlap with other code possible).
   playblocking(player);
end
load('Multi.mat');          % Charger le fichier
signal=importdata('Multi.mat');
%Paramètres
f0 = 38000;           % Fréquence porteuse (Hz)
fe = 132300;          % Fréquence d'échantillonnage (Hz)
fc = f0 / 2;          % Fréquence centrale du passe-bande (Hz)
delta_f = 2000;       % Demi-largeur de bande passante (Hz)
n = 100;              % Ordre initial du filtre (à ajuster)
%Fréquences limites normalisées
W1 = (fc - delta_f) / (fe / 2); % Limite inférieure
W2 = (fc + delta_f) / (fe / 2); % Limite supérieure
%Synthèse du filtre passe-bande
h = fir1(n, [W1 W2], 'bandpass', hamming(n+1)); % Fenêtre Hamming
[H, freq] = freqz(h, 1, 1024, fe); % Calcul de la réponse en fréquence
amplitude_dB = 20 * log10(abs(H));
figure;
plot(freq, amplitude_dB);
xlabel('Fréquence (Hz)'); ylabel('Amplitude (dB)');
```

```
title('Réponse fréquentielle du filtre passe-bande');
grid on;
v = filter(h, 1, signal); % Filtrer le signal y(t)
t = (0:length(v)-1) / fe;
figure;
plot(t, v);
xlabel('Temps (s)');
ylabel('Amplitude');
title('Signal filtré dans le domaine temporel');
N = length(v);
V = fft(v);
f = (0:N-1) * (fe / N);
figure;
plot(f(1:N/2), abs(V(1:N/2)));
xlabel('Fréquence (Hz)');
ylabel('Amplitude');
title('Spectre du signal filtré');
wavplay(x1, fe)
```

Proposition d'une méthode pour doubler sa fréquence et obtenir ainsi cos(2πf0t)

```
function wavplay(signal, sampleRate)
% WAVPLAY  Plays an audio signal using the audioplayer object.
%   WAVPLAY(x, Fs) plays the audio signal x at the specified sample rate Fs.
   % Instantiate object to play audio data.
   player = audioplayer(signal, sampleRate);
   % Play signal from beginning to end (no overlap with other code possible).
   playblocking(player);
end

load('Multi.mat');          % Charger le fichier
signal=importdata('Multi.mat');
f0 = 38000;           % Fréquence porteuse (Hz)
fe = 132300;          % Fréquence d'échantillonnage (Hz)
fc = f0 / 2;          % Fréquence centrale du passe-bande (Hz)
delta_f = 2000;       % Demi-largeur de bande passante (Hz)
n = 100;              % Ordre initial du filtre (à ajuster)
%Fréquences limites normalisées
W1 = (fc - delta_f) / (fe / 2); % Limite inférieure
W2 = (fc + delta_f) / (fe / 2); % Limite supérieure
%Synthèse du filtre passe-bande
h = fir1(n, [W1 W2], 'bandpass', hamming(n+1)); % Fenêtre Hamming
[H, freq] = freqz(h, 1, 1024, fe); % Calcul de la réponse en fréquence
amplitude_dB = 20 * log10(abs(H));
delay = n / (2 * fe); % Retard en secondes
fprintf('Le retard induit par le filtre est %.6f secondes.\n', delay);
v = filter(h, 1, signal); % Filtrer le signal y(t)
N = length(v);
```

```
V = fft(v);
f = (0:N-1) * (fe / N);
subplot(2,2,1)
plot(f(1:N/2), abs(V(1:N/2)));
xlabel('Fréquence (Hz)');
ylabel('Amplitude');
title('Spectre du signal filtré');
wavplay(v, fe)

%Signal cos(2πf0t)
v_double = 2*v.^2 - 1;
N_double = length(v);
V = fft(v_double);
f = (0:N_double-1) * (fe / N_double);
subplot(2,2,2)
plot(f(1:N_double/2), abs(V(1:N_double/2)));
xlabel('Fréquence (Hz)');
ylabel('Amplitude');
title('Spectre de cos(2πf0t)');
wavplay(v_double, fe)
```

IV RECUPERATION DU SIGNAL X2(T)

```
load('Multi.mat');          % Charger le fichier
signal=importdata('Multi.mat');
f0 = 38000;           % Fréquence porteuse (Hz)
fe = 132300;          % Fréquence d'échantillonnage (Hz)
% Paramètres du filtre passe-bande
fc1 = f0 - 2000; % Bord inférieur (f0 - bande de transition)
fc2 = f0 + 2000; % Bord supérieur (f0 + bande de transition)
n = 100;      % Ordre du filtre passe-bande
% Conception du filtre passe-bande
h = fir1(n, [fc1, fc2] / (fe / 2), 'bandpass'); % Normalisation des fréquences
% Représentation fréquentielle du filtre
[H, freq] = freqz(h, 1, 1024, fe); % Réponse en fréquence du filtre
% Affichage du filtre
figure;
plot(freq, 20*log10(abs(H)));
xlabel('Fréquence (Hz)'); ylabel('Amplitude (dB)');
title('Réponse fréquentielle du filtre passe-bande');
grid on;
% Filtrage du signal y(t)
x2 = filter(h, 1, signal);
% Calcul du retard induit par le filtre
retard = n / 2 / fe; % Retard en secondes
% Affichage du signal filtré dans le domaine temporel
```

```
t = (0:length(x2)-1) / fe; % Vecteur temps
figure;
plot(t, x2);
xlabel('Temps (s)'); ylabel('Amplitude');
title('Signal x2(t) modulé');
grid on;
% Calcul de la FFT de x2(t)
X2 = fft(x2);
f = linspace(0, fe, length(X2));
% Affichage fréquentiel
figure;
plot(f(1:end/2), abs(X2(1:end/2)));
xlabel('Fréquence (Hz)');
ylabel('Amplitude');
title('Représentation fréquentielle du signal x2(t) modulé');
grid on;
```

- *Représentation temporelle et fréquentielle de Signal x2(t) modulée Multiplié par 2cos(2πf0t), le signal obtenu noté v(t) et Filtrage du résultat par un passe-bas pour obtenir le signal $X_2(t)$ (v(t) filtré) :*

```
function wavplay(signal, sampleRate)
% WAVPLAY  Plays an audio signal using the audioplayer object.
%   WAVPLAY(x, Fs) plays the audio signal x at the specified sample rate Fs.
   % Instantiate object to play audio data.
   player = audioplayer(signal, sampleRate);
   % Play signal from beginning to end (no overlap with other code possible).
   playblocking(player);
end

load('Multi.mat');         % Charger le fichier
signal=importdata('Multi.mat');
f0 = 38000;          % Fréquence porteuse (Hz)
fe = 132300;         % Fréquence d'échantillonnage (Hz)
% Paramètres du filtre passe-bande
fc1 = f0 - 2000;
fc2 = f0 + 2000;
n = 100;     % Ordre du filtre passe-bande
% Conception du filtre passe-bande
h = fir1(n, [fc1, fc2] / (fe / 2), 'bandpass'); % Normalisation des fréquences
% Représentation fréquentielle du filtre
[H, freq] = freqz(h, 1, 1024, fe); % Réponse en fréquence du filtre
% Filtrage du signal y(t)
x2 = filter(h, 1, signal);
% Génération du signal 2cos(2πf0t)
t = (0:length(x2)-1) / fe; % Vecteur temps correspondant au signal x2
Signal_cos = 2 * cos(2 * pi * f0 * t); % Signal porteur
```

```
% Multiplication pour ramener à la bande de base
Signal_resultat = x2 .* Signal_cos;
% % Affichage du signal temporel résultant
figure;
plot(t, Signal_resultat);
xlabel('Temps (s)');
ylabel('Amplitude');
title('Signal V(t) "x2(t)modulée Multiplié par 2cos(2πf0t)"');
grid on;
% Affichage du signal fréquantiel résultant
N = length(Signal_resultat);
X1 = fft(Signal_resultat);
f = (0:N-1) * (fe / N);
figure;
plot(f(1:N/2), abs(X1(1:N/2)));
xlabel('Fréquence (Hz)');
ylabel('Amplitude');
title('Spectre de V(t) "x2(t)modulée Multiplié par 2cos(2πf0t)"');

% Réutilisation du filtre passe-bas (h_pb défini dans la partie précédente)
fe = 132300;        % Fréquence d'échantillonnage
fc = 13000;         % Fréquence de coupure (Hz)
Wc = fc / (fe / 2);   % Fréquence de coupure normalisée
n = 100;            % Ordre initial du filtre (ajustez par essais)
h_pb = fir1(n, Wc, 'low', hamming(n+1)); % Filtre passe-bas
x2_baseband = filter(h_pb, 1, Signal_resultat);
% Affichage du signal fréquantiel résultant
N = length(x2_baseband);
X1 = fft(x2_baseband);
f = (0:N-1) * (fe / N);
% Affichage du signal filtré dans le domaine temporel
figure;
plot(t, x2_baseband);
xlabel('Temps (s)');
ylabel('Amplitude');
title('Signal X2(t) c-à-d le V(t) filté)');
grid on;
figure;
plot(f(1:N/2), abs(X1(1:N/2)));
xlabel('Fréquence (Hz)');
ylabel('Amplitude');
title('Spectre de X2(t) c-à-d le V(t) filté')
wavplay(x2_baseband, fe)
```

:

V RECUPERATION DES SIGNAUX g(t) ET d(t)

Reconstruire les signaux g(t) et d(t) avec notre méthode

```
load('Multi.mat');          % Charger le fichier
signal=importdata('Multi.mat');
fe = 132300;          % Fréquence d'échantillonnage
fc = 13000;           % Fréquence de coupure (Hz)
Wc = fc / (fe / 2);   % Fréquence de coupure normalisée
n = 100;              % Ordre initial du filtre (ajustez par essais)
h = fir1(n, Wc, 'low', hamming(n+1)); % Filtre passe-bas
x1 = filter(h, 1, signal); % Remplacez 'y' par votre signal réel

% Paramètres du filtre passe-bande
fc1 = f0 - 2000;
fc2 = f0 + 2000;
% Conception du filtre passe-bande
h1 = fir1(n, [fc1, fc2] / (fe / 2), 'bandpass'); % Normalisation des fréquences
% Filtrage du signal y(t)
x2 = filter(h1, 1, signal);
% Génération du signal 2cos(2πf0t)
t = (0:length(x2)-1) / fe; % Vecteur temps correspondant au signal x2
Signal_cos = 2 * cos(2 * pi * f0 * t); % Signal porteur
% Multiplication pour ramener à la bande de base
Signal_resultat = x2 .* Signal_cos;
% Réutilisation du filtre passe-bas (h_pb défini dans la partie précédente)
h2 = fir1(n, Wc, 'low', hamming(n+1)); % Filtre passe-bas
x2_baseband = filter(h2, 1, Signal_resultat);

% Tracer les signaux g(t) et d(t)
t = (0:length(g_t)-1) / fe; % Axe temporel
figure;
subplot(2,1,1);
plot(t, g_t);
title('Signal g(t) ');
xlabel('Temps (s)');
ylabel('Amplitude');
subplot(2,1,2);
plot(t, d_t);
title('Signal d(t) ');
xlabel('Temps (s)');
ylabel('Amplitude');
% Reconstruction des signaux g(t) et d(t)
g_t = (x1 + x2_baseband) / 2; % Signal gauche
d_t = (x1 - x2_baseband) / 2; % Signal droite
% Écoute des signaux reconstruits
sound(g_t, fe); % Écouter g(t) (signal gauche)
pause(length(g_t) / fe + 1); % Pause avant de jouer le deuxième signal
sound(d_t, fe); % Écouter d(t) (signal droite)
```