

	<i>Nom</i> MARQUET <i>Prenom</i> Félix
--	---

## Expérience 1

Charger le signal depuis '*TP4cipaQ1.mat*'

Les échantillons y contenus résultent d'un échantillonnage à 96 KHz

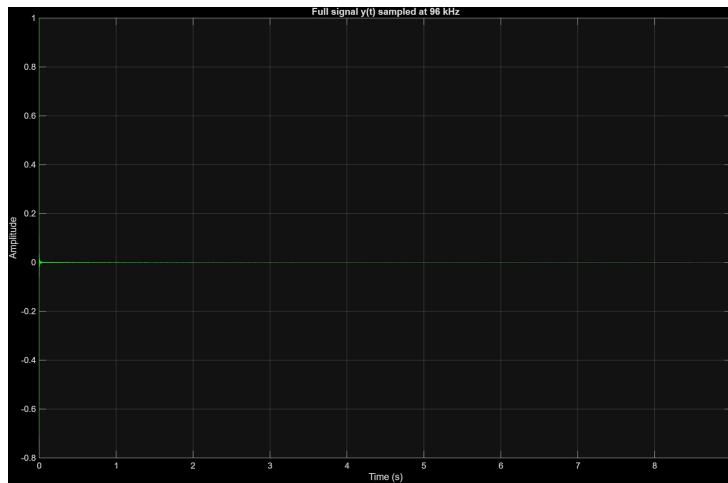
- i/ Afficher le signal en fonction du temps en secondes
- ii/ Afficher la portion du signal correspondant aux 10000 premiers échantillons
- iii/ Afficher le spectre d'amplitude du signal, les fréquences en Hz

Constater une portion d'énergie en basse fréquence et une autre en hautes fréquences  
Concevoir et appliquer un filtrage pour récupérer le signal en bande de base

- iv/ Afficher le spectre d'amplitude du signal filtré, les fréquences en KHz
- v/ Afficher le signal filtré en fonction du temps en secondes
- vi/ Afficher les 10000 premiers échantillons de ce dernier

Insérer les figures ici

i/

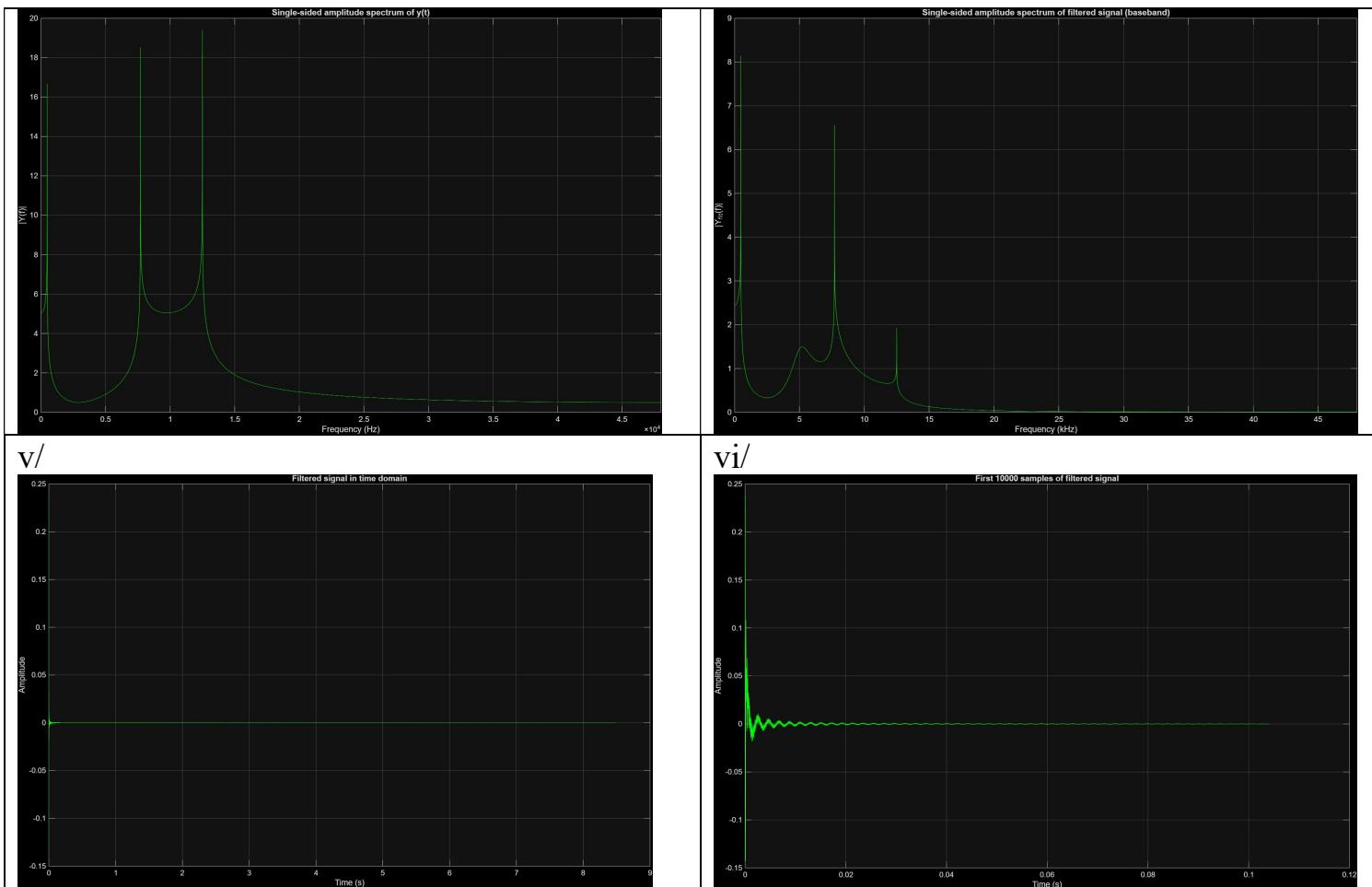


ii/



iii/

iv/



Insérer ici le code Matlab permettant de générer ces figures

`% Experiment 1 : Baseband recovery from TP4cipaQ1.mat`

```

clc;
clear;
close all;

%% 0) Load data

data = load('TP4cipaQ1.mat');
fields = fieldnames(data);           % Get field names
sig = data.(fields{1});
sig = sig(:);

Fs = 96000;                         % Sampling frequency (Hz)
Ts = 1/Fs;                          % Sampling period (s)

N = length(sig);                   % Number of samples
t = (0:N-1) * Ts;                 % Time axis (s)

%% i) Plot full signal y(t)

figure('Name','i) Full signal y(t)');
plot(t, sig, 'g');
xlabel('Time (s)');
ylabel('Amplitude');
title('Full signal y(t) sampled at 96 kHz');
grid on;

%% ii) Plot first 10000 samples of y(t)

Nseg = min(10000, N);             % Limit to 10000 samples
t_seg = t(1:Nseg);
sig_seg = sig(1:Nseg);

figure('Name','ii) First 10000 samples of y(t)');
plot(t_seg, sig_seg, 'g');

```

```

xlabel('Time (s)');
ylabel('Amplitude');
title('First 10000 samples of y(t)');
grid on;

%% iii) Amplitude spectrum of y(t) (frequencies in Hz)

Y      = fft(sig);                      % FFT of signal
magY  = abs(Y);                         % Magnitude spectrum

halfN   = floor(N/2) + 1;                 % Positive-frequency part
magY_half = magY(1:halfN);
f       = (0:halfN-1) * (Fs/N);          % Frequency axis (Hz)

figure('Name','iii) Amplitude spectrum of y(t)');
plot(f, magY_half, 'g');
xlabel('Frequency (Hz)');
ylabel('|Y(f)|');
title('Single-sided amplitude spectrum of y(t)');
grid on;
xlim([0 Fs/2]);

%% Low-pass filter design with Butterworth (baseband extraction)

Fc     = 5000;                          % Cutoff frequency (Hz)
Wn     = Fc / (Fs/2);                   % Normalized cutoff (0..1) relative to Nyquist
order  = 4;                            % Filter order

[b, a] = butter(order, Wn, 'low'); % Low-pass Butterworth filter

%% Apply the low-pass filter

y_filt = filter(b, a, sig);

%% iv) Amplitude spectrum of filtered signal (frequencies in kHz)

YF    = fft(y_filt);
magYF = abs(YF);

magYF_half = magYF(1:halfN);
f_kHz      = f / 1000;                  % Frequency axis in kHz

figure('Name','iv) Spectrum of filtered signal');
plot(f_kHz, magYF_half, 'g');
xlabel('Frequency (kHz)');
ylabel('|Y_{filt}(f)|');
title('Single-sided amplitude spectrum of filtered signal (baseband)');
grid on;
xlim([0 (Fs/2)/1000]);

%% v) Filtered signal y_filt(t) versus time (s)

figure('Name','v) Filtered signal y\_\_filt(t)');
plot(t, y_filt, 'g');
xlabel('Time (s)');
ylabel('Amplitude');
title('Filtered signal in time domain');
grid on;

%% vi) First 10000 samples of filtered signal

y_filt_seg = y_filt(1:Nseg);

figure('Name','vi) First 10000 samples of filtered signal');
plot(t_seg, y_filt_seg, 'g');
xlabel('Time (s)');
ylabel('Amplitude');
title('First 10000 samples of filtered signal');

```

```
grid on;
```

## Expérience 2

Charger le signal depuis '*TP4cipaQ2.wav*'

Ecouter le signal et constater la présence d'un bruit

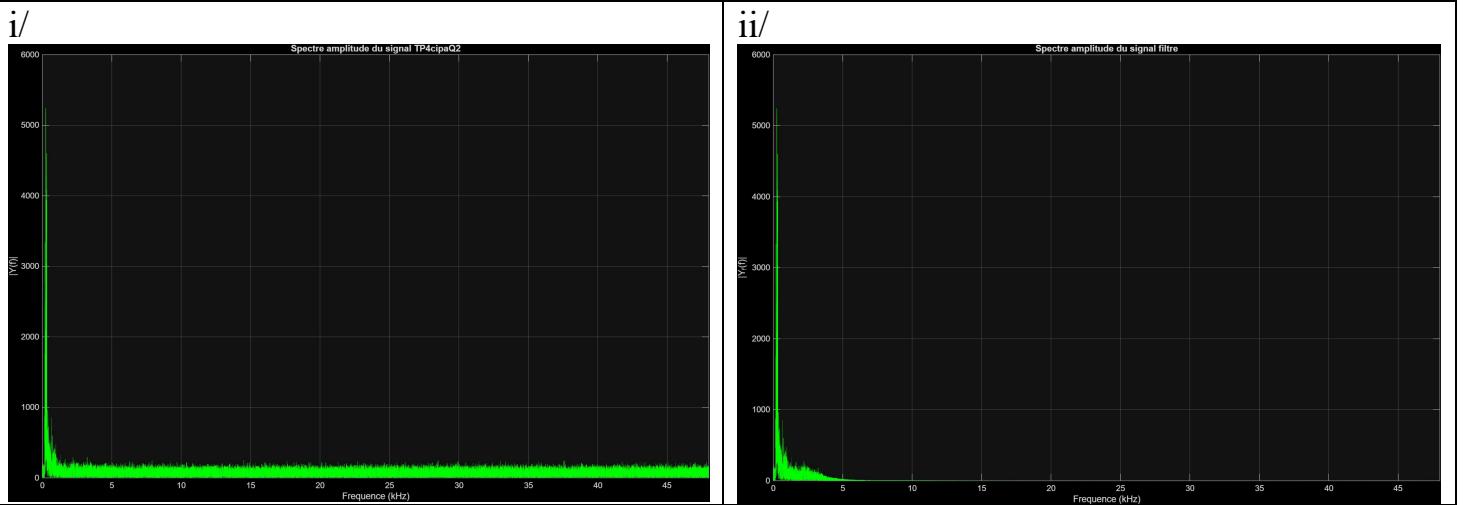
i/ Afficher le spectre d'amplitude du signal avec les fréquences en KHz

Constater une portion d'énergie en basse fréquence noyée dans un bruit blanc

Concevoir et appliquer un filtrage adapté pour récupérer le signal en bande de base

ii/ Afficher le spectre du signal après filtrage

Insérer les figures ici



Insérer ici le code Matlab permettant de générer ces figures

```
% Experiment 2 : Baseband recovery from TP4cipaQ2.wav
```

```
clc;
clear;
close all;

% Load the audio
[y, Fs] = audioread('TP4cipaQ2.wav');
if size(y,2) > 1
    y = mean(y,2); % Switch to mono
end

N = length(y);

%% i) Spectre of amplitude in kHz
Y = fft(y);
magY = abs(Y);
halfN = floor(N/2)+1;
magY_half = magY(1:halfN);
f = (0:halfN-1)*(Fs/N); % Hz
f_kHz = f/1000; % kHz

figure;
plot(f_kHz, magY_half, 'g');
xlabel('Fréquence (kHz)');
ylabel('|Y(f)|');
title('Spectre amplitude du signal TP4cipaQ2');
grid on;
xlim([0 Fs/2000]);

%% Butterworth filter
```

```

Fc = 3000;
Wn = Fc/(Fs/2);
ordre = 4;

[b,a] = butter(ordre, Wn, 'low');

y_filt = filter(b,a,y);

%% ii) Spectre of filtered signal
Yf = fft(y_filt);
magYf = abs(Yf);
magYf_half = magYf(1:halfN);

figure;
plot(f_kHz, magYf_half, 'g');
xlabel('Frequence (kHz)');
ylabel('|Y_f(f)|');
title('Spectre amplitude du signal filtre');
grid on;
xlim([0 Fs/2000]);

sound(y_filt, Fs);

```

iii/ Ecouter le signal filtré

iv/ Expliquer la persistance d'un bruit

Le filtrage passe-bas supprime les composantes de bruit hors de la bande de base, mais le bruit blanc contient de l'énergie à toutes les fréquences (incluant la bande utile). Le filtre laisse donc passer à la fois le signal utile et une partie du bruit blanc (la partie présente sur la même bande de fréquence).

### Expérience 3

Charger le signal depuis '*TP4cipaQ3.mat*'

Les échantillons y contenus résultent d'un échantillonnage à 96 KHz

Recuperer les deux vecteurs  $y_1$  et  $y_2$  sensés contenir, respectivement, les échantillons en provenance des signaux suivants  $a(t)\cos(2\pi 30000t+A)$  et  $a(t)\sin(2\pi 30000t+A)$

On souhaite récupérer le signal d'information  $a(t)$  malgré une phase A inconnue

HINT :  $\cos^2 + \sin^2 = 1$

i/ Expliquer comment récupérer  $a(t)$

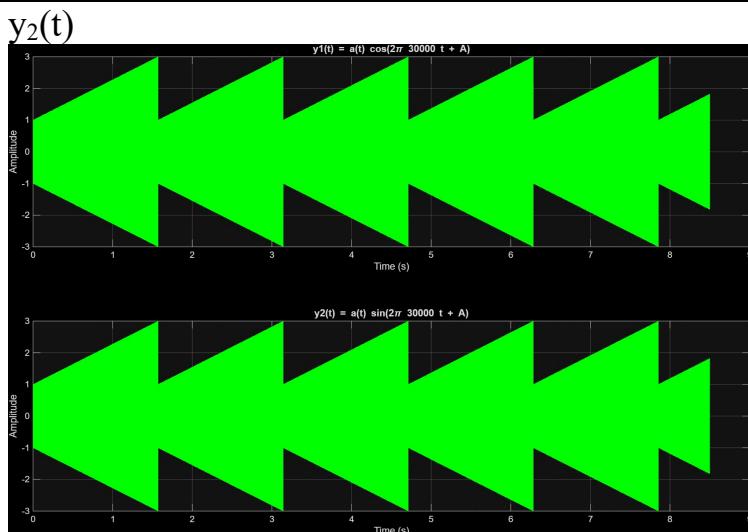
On a  $y_1(t) = a(t)\cos(2\pi 30000t+A)$  et  $y_2(t) = a(t)\sin(2\pi 30000t+A)$ . En éllevant au carré et en sommant :

$$y_1^2(t) + y_2^2(t) = a^2(t)(\cos^2(t) + \sin^2(t)) = a^2(t)$$

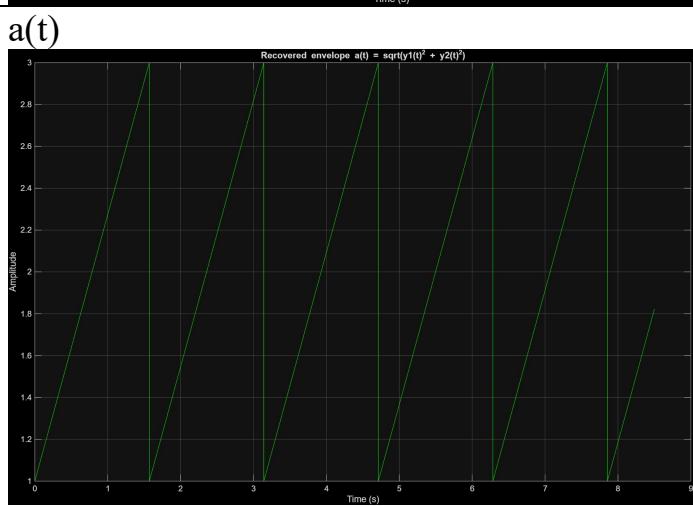
Donc,  $a(t) = \sqrt{y_1^2(t) + y_2^2(t)}$ .

ii/ Implémenter ce traitement avec Matlab et afficher les signaux  $y_1(t)$ ,  $y_2(t)$ ,  $a(t)$ , ainsi que le spectre d'amplitude de  $a(t)$  avec l'axe horizontal en KHz, et l'axe vertical en dB

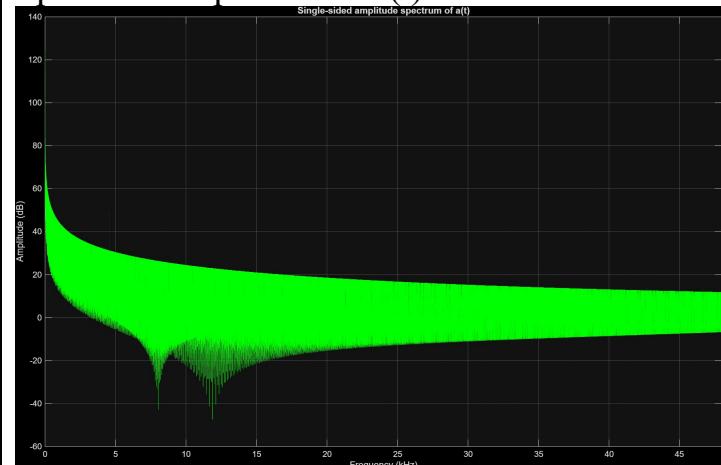
Insérer les figures ici



$y_1(t)$



Spectre d'amplitude de  $a(t)$



Insérer ici le code Matlab permettant de générer ces figures

```

%% Experiment 3 : Recovery of a(t) from y1(t) and y2(t)

clc;
clear;
close all;

%% Load data from MAT-file

data = load('TP4cipaQ3.mat');
y1 = data.y1;
y2 = data.y2;
y1 = y1(:);
y2 = y2(:);

Fs = 96000; % Sampling frequency (Hz)
Ts = 1/Fs; % Sampling period (s)
N = length(y1); % Number of samples
t = (0:N-1)*Ts; % Time axis (s)

%% Recover a(t) using the hint cos^2 + sin^2 = 1
% a(t) is the envelope common to y1 and y2.

a = sqrt( y1.^2 + y2.^2 );

%% Plot y1(t) and y2(t) in time domain

figure('Name','y1(t) and y2(t)');
subplot(2,1,1);
plot(t, y1, 'g');
xlabel('Time (s)');
ylabel('Amplitude');
title('y1(t) = a(t) cos(2\pi 30000 t + A)');
grid on;

subplot(2,1,2);
plot(t, y2, 'g');
xlabel('Time (s)');
ylabel('Amplitude');
title('y2(t) = a(t) sin(2\pi 30000 t + A)');
grid on;

%% Plot a(t) in time domain

figure('Name','a(t)');
plot(t, a, 'g');
xlabel('Time (s)');
ylabel('Amplitude');
title('Recovered envelope a(t) = sqrt(y1(t)^2 + y2(t)^2)');
grid on;

%% Amplitude spectrum of a(t) in kHz and in dB

A_fft = fft(a);
magA = abs(A_fft);

halfN = floor(N/2) + 1; % Positive frequencies only
magA_half = magA(1:halfN);

f_Hz = (0:halfN-1)*(Fs/N); % Frequency axis (Hz)
f_kHz = f_Hz / 1000; % Convert to kHz

% Convert magnitude to dB scale (20*log10), avoid log(0) using eps
magA_dB = 20*log10(magA_half + eps);

figure('Name','Amplitude spectrum of a(t)');
plot(f_kHz, magA_dB, 'g');
xlabel('Frequency (kHz)');
ylabel('Amplitude (dB)');

```

```
title('Single-sided amplitude spectrum of a(t)');
grid on;
xlim([0 Fs/2000]); % From 0 to Nyquist frequency in kHz
```