BARRACO Thomas
CECCONI Marie

GENESE 1
14/01/16

Traitement du Signal
TD5 MATLAB

OBJECTIF : Ce TP a pour but de nous permettre de voir quelques aspects pratiques du cours de traitement du signal grâce au logiciel Matlab. Nous commencerons par échantillonner la fonction porte et calculer et tracer sa transformée de Fourrier, souvent utilisée en traitement du signal, pour ensuite filtrer un signal afin d'en recueillir l'information utile.

A. Echantillonnage d'un signal porte

Le but de l'exercice est de calculer la transformée de Fourier théorique de la fonction porte, puis de comparer ce résultat avec la transformée de Fourier numérique donnée par le logiciel de calcul. Nous imposerons au préalable trois paramètres qui sont la largeur T de la porte, la durée D du signal et la période Te = 1/Fe d'échantillonnage.
On prend par exemple :
Fe = 8 Hz et une résolution fréquentielle Rf = 1/D = 1/8 Hz

Le logiciel Matlab ne manipule pas de fonctions continues, seulement des signaux discrets. Pour définir et traiter la fonction porte nous allons donc l'échantillonner : on multiplie le signal par une série d'impulsions de Dirac espacées d'une durée Te.
Avec nos paramètres fixés précédemment, on choisit un nombre de points total égal à D qui doit être un multiple de 2. On placera alors 28 points à 0 à gauche, 28 points à 0 à droite et 8 points à l'amplitude 1 au centre.

```
% Tracé discret de la fonction porte
close all
clear all

D=64;        %nombre de points total dans le signal
Te=1;
T=8;
Np=T/Te;     %nombre de points à 1
N=D/Te;

p=ones(1,Np);
p=[zeros(1,(N-Np)/2),p,zeros(1,(N-Np)/2)];

t=-D/2:Te:D/2-Te;    %on enlève le point en 0 pour avoir une fonction paire
(symétrie par rapport à 0)
stem(t,p)
title('Porte échantillonnée')
```

Un signal pair a une transformée de Fourier réelle, c'est pourquoi on s'arrange pour centrer nos signaux.

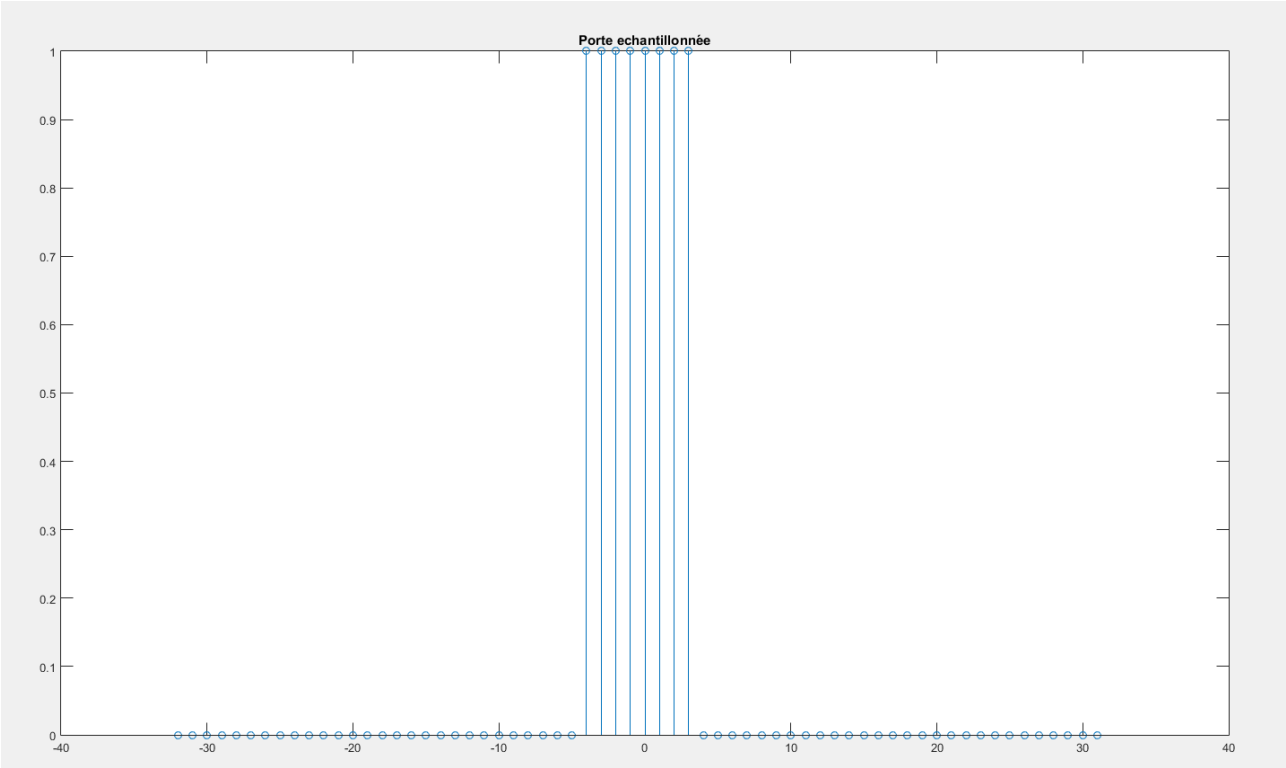


Illustration 1 : Fonction porte discrétisée (D=64, T=8s, Te=1s)

Pour tracer le spectre du signal, on utilise les fonctions fft puis fftshift pour réorganiser les valeurs :

```
% Tracé du spectre de la FFT

y=fft(p,D);
y1=fftshift(y);
Fe=1/Te;
Rf=1/D;      %résolution fréquentielle : pas de la FTT
f=-Fe/2:Rf:Fe/2-Rf;

figure(2)
%subplot(1,2,1)
plot(f,abs(y1))
title('Spectre FFT ')
%subplot(1,2,2)
hold on
stem(f,abs(y1))
%title('Spectre FFT discret')
```

Remarque : ici, le théorème de Shannon qui stipule que Fe ≥ 2Fmax ne peut être respecté, étant donné que la fréquence maximale du sinus cardinal est infinie à cause de ses oscillations.

On sait également qu'avec un signal périodique en temporel, on obtiendra un signal fréquentiel discret du même nombre de points.

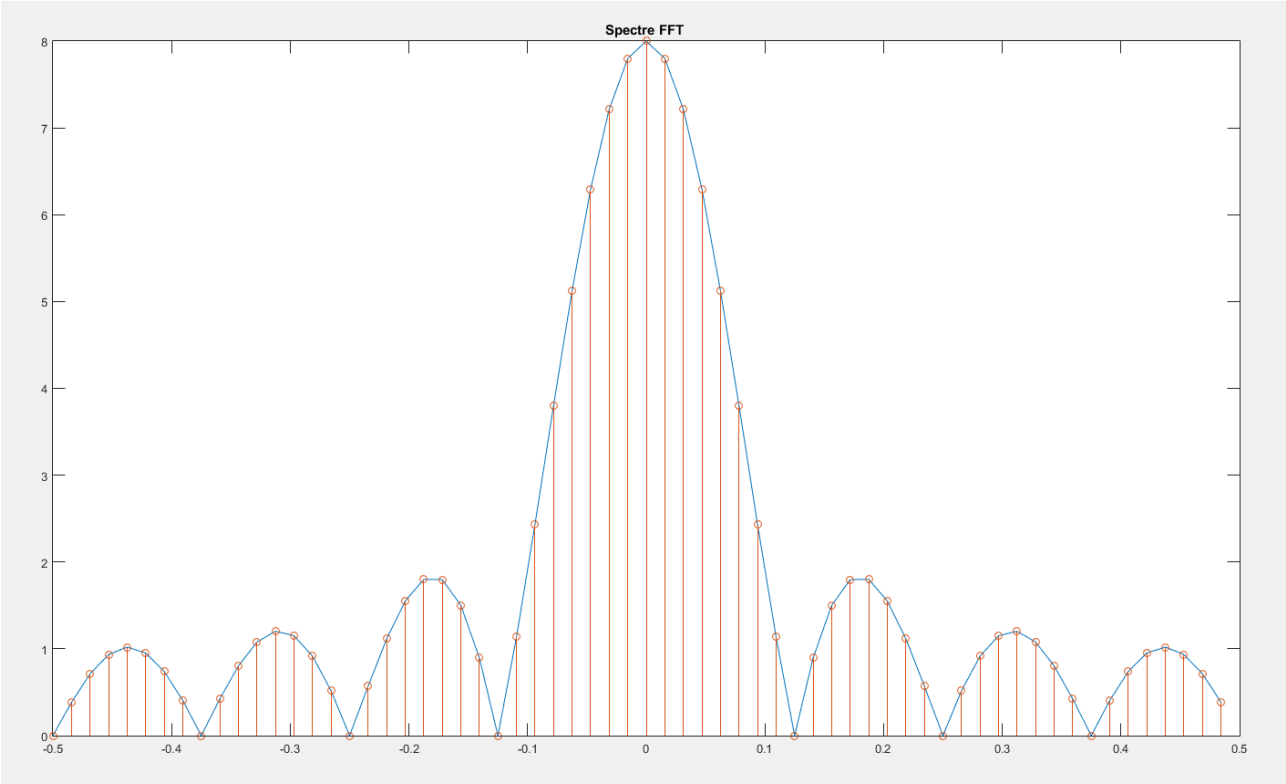


Illustration 2 : Superposition de la FFT discrétisée et continue (D=64, T=8s, Te=1s)

On obtient bien un sinus cardinal pour transformée de Fourrier de la fonction porte. Maintenant, si on diminue la période d'échantillonnage, on augmente la précision du spectre de la FFT *(illustration 3)*.

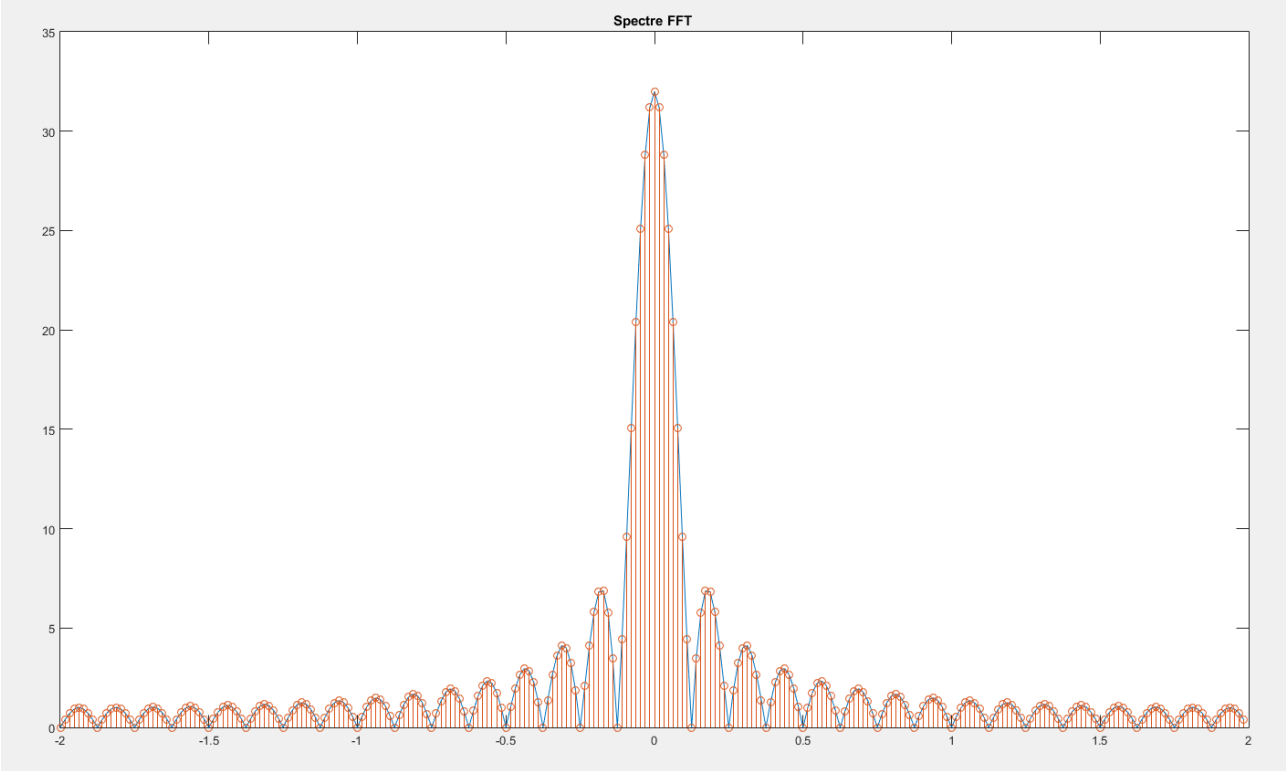


Illustration 3 : Superposition de la FFT discrétisée et continue (D=64, T=8s, Te=0,25s)

Cependant, si l'on n'échantillonne pas assez, on a une perte d'informations et on n'obtient seulement une faible partie du signal *(illustration 4)*.

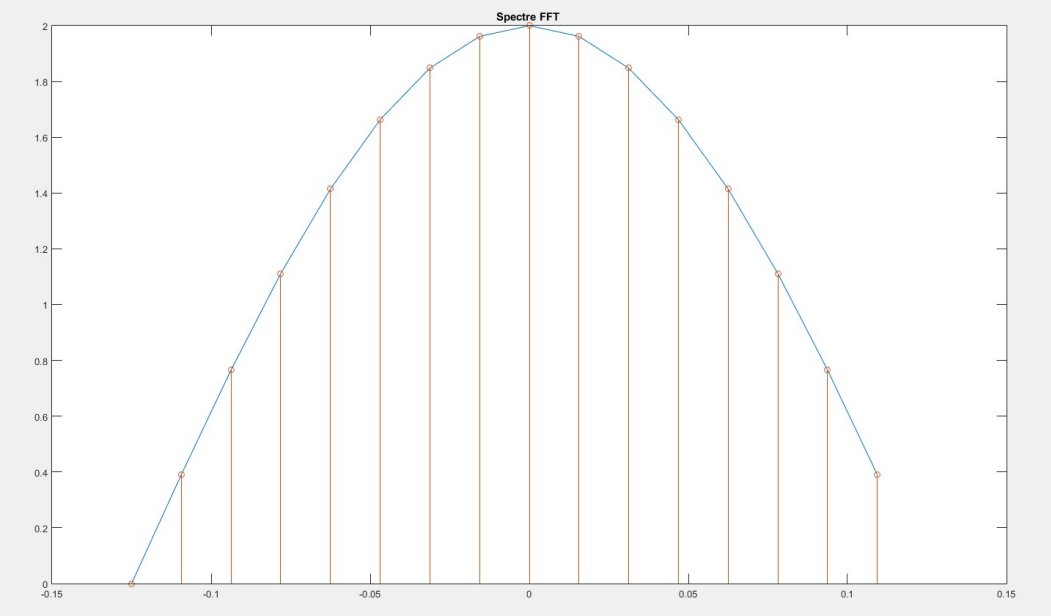


Illustration 4 : Superposition de la FFT discrétisée et continue (D=64, T=8s, Te=4s)

A présent, en laissant Te = 1s et en augmentant le nombre de points sur lequel on définit notre signal à D = 128, on observe l'effet du zéro-padding qui améliore la précision de l'échantillonnage *(illustration 5)*.

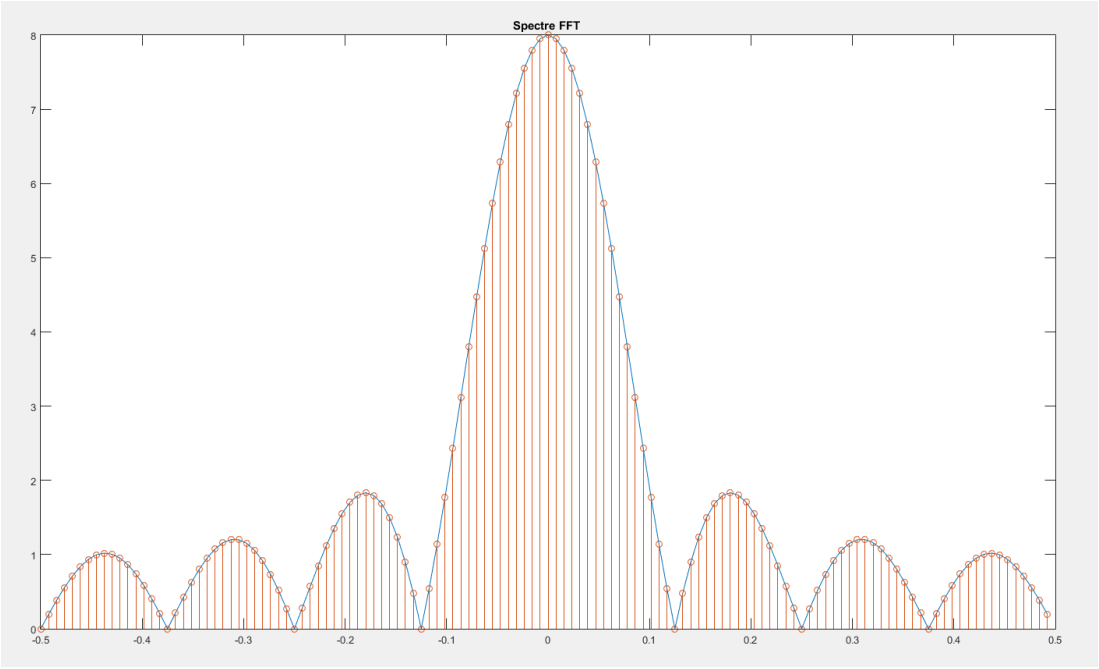


Illustration 5 : Superposition de la FFT discrétisée et continue (D=128, T=8s, Te=1s)

B. Traitement numérique d'un signal par filtrage RII et RIF

Le but de cette partie est d'analyser un signal par FFT reconstruit à partir de données numériques contenues dans un fichier « mesure.dat ». On fera apparaître les différentes composantes en fréquence et en amplitude et nous caractériserons le bruit.
Ensuite, comme nous l'avons annoncé, nous allons filtrer le signal afin d'extraire la composante de plus haute fréquence, par deux méthodes : filtrage RII et RIF.

Les filtres RII sont des filtres dont la fonction de transfert est une fraction rationnelle en z, tandis que les filtres RIF ont une fonction de transfert en polynôme en z.

On trace le signal de sortie ainsi que son spectre :

```
clear all
close all

load ('mesure.dat')

t = mesure(1,:);                    %vecteur temps
x = mesure(2,:);                    %signal de sortie
n = length(t);                      %nombre de points
Te = mesure(1,2)-mesure(1,1);       %période d'échantillonnage
Fe = 1/Te;                          %fréquence d'échantillonnage

duree = mesure(1,1024);             %durée du signal
figure(1)
plot(t,x)
title('Signal reçu')

%Tracé de la FFT

f=-Fe/2:1/duree:Fe/2;
y=fft(x);
y1=fftshift(y)*Te/duree;
figure(2)
plot(f,abs(y1))
title('Spectre FFT')
```

On observe le signal sur l'illustration 6.

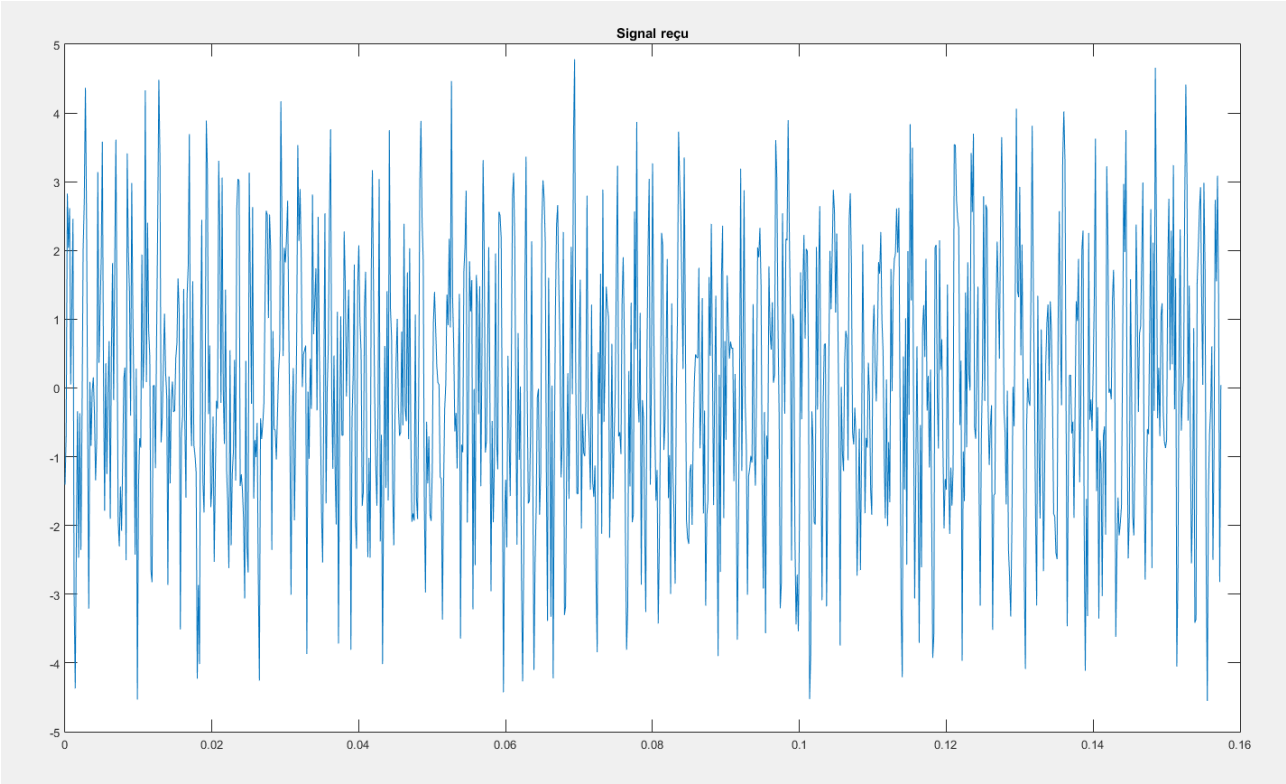


Illustration 6 : Signal de sortie récupéré à partir du fichier mesure.dat

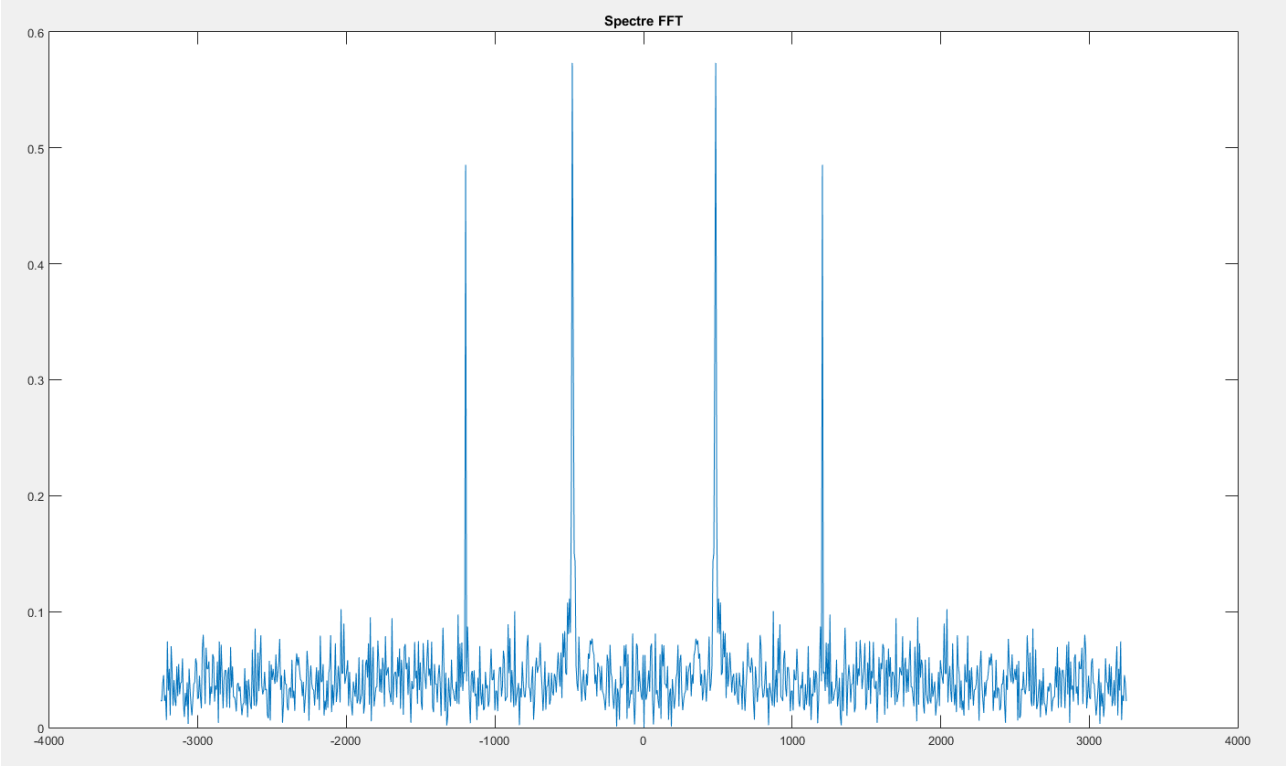


Illustration 7 : Spectre de la FFT du signal

Dans cette analyse spectrale on remarque qu'il y a 4 pics qui correspondent à des cosinus à 500Hz et 1200Hz, ainsi qu'un bruit sur toute la durée du signal.

On veut donc faire un filtre passe-bande à 1200Hz.

- **Filtre RII**

Pour le filtre RII, le raisonnement est de type analogique. On se donne un polynôme (Butterworth par exemple) afin de générer une fonction de transfert continue que l'on va discrétiser. Pour cela, il suffit de remplacer le p de Laplace par $1-z^{-1}$ pour trouver la fraction en z du type :

$$H(z) = \frac{B(z^{-1})}{A(z^{-1})} = \frac{b(0) + b(1) * z^{-1} + \ldots}{a(0) + a(1) * z^{-1} + \ldots}$$

On raisonne tout d'abord en continu et ensuite on discrétise. Il faut régler la bande passante et l'ordre du filtre pour positionner les zéros.

On commence par régler l'ordre du filtre à 1 puis à 2 avec une bande passante BP=[1100Hz,1300Hz].

```
%Création du filtre de Butterworth

ordre = 1;                  %ordre du filtre de Butterworth

BP = [1100 1300]/Fe/2;      %bande passante du filtre normalisée pour se
ramener entre 0 et 1
[b,a] = butter(ordre,BP);
butt=filter(b,a,x);         %filtrage du signal de sortie
figure(3)
plot(t,butt)
title('Signal filtré')

figure(4)
freqz(b,a)                  %diagramme de Bode de la réponse fréquentielle
title('Digramme de Bode du filtre RII')
```

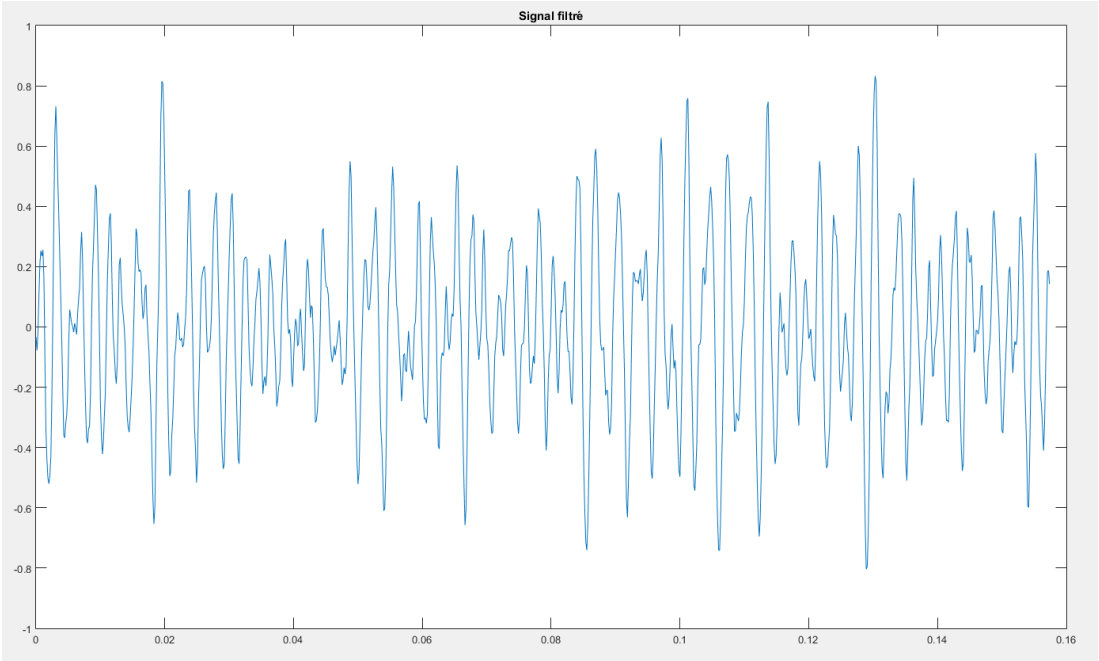


Illustration 8 : Allure du signal filtré par Butterworth RII (ordre 1)

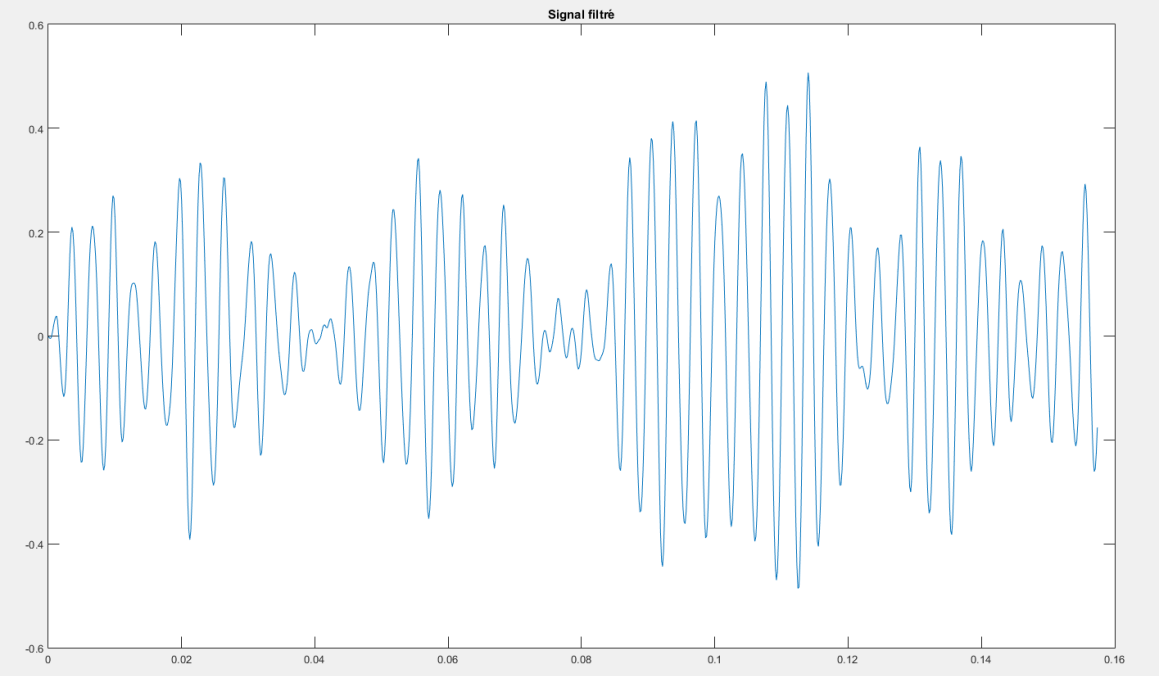


Illustration 9 : Allure du signal filtré par Butterworth RII (ordre 2)

On remarque qu'avec un polynôme d'ordre faible, le temps de réponse n'est pas très élevé *(illustration 10)*. Mais cela n'influe pas énormément sur l'allure du signal.

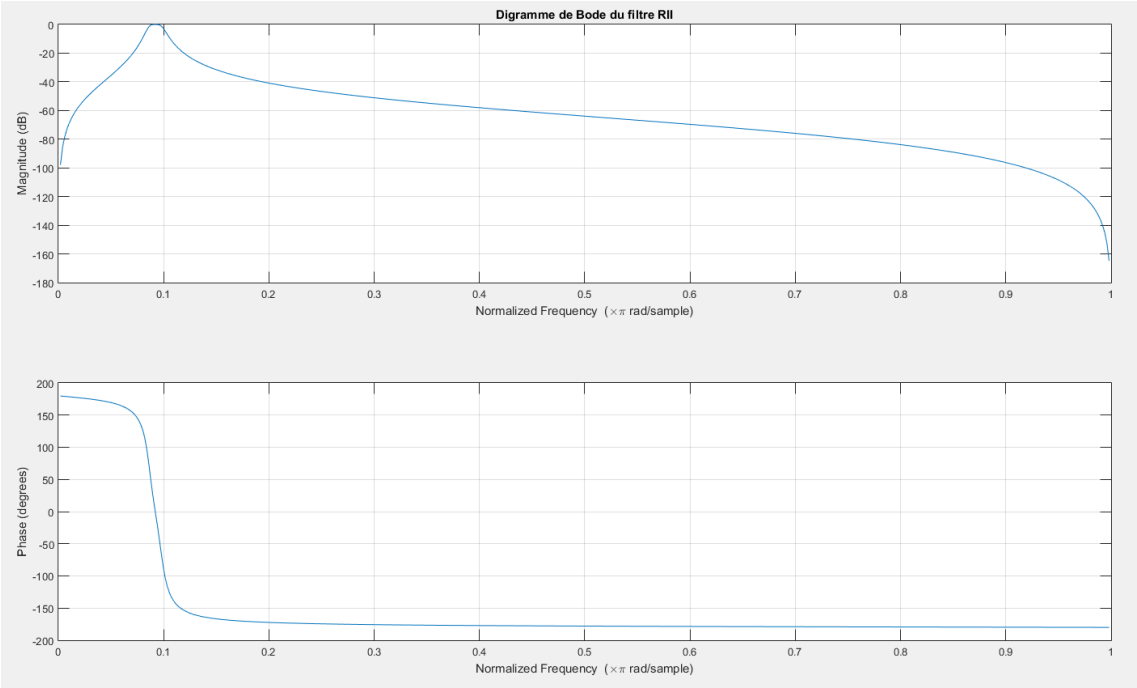


Illustration 10 : Diagramme de Bode du filtre Butterworth RII

Alors que si l'on prend des ordres plus grands, 4 par exemple, on aura une meilleure qualité de filtrage mais un retard important *(illustration 11)*.

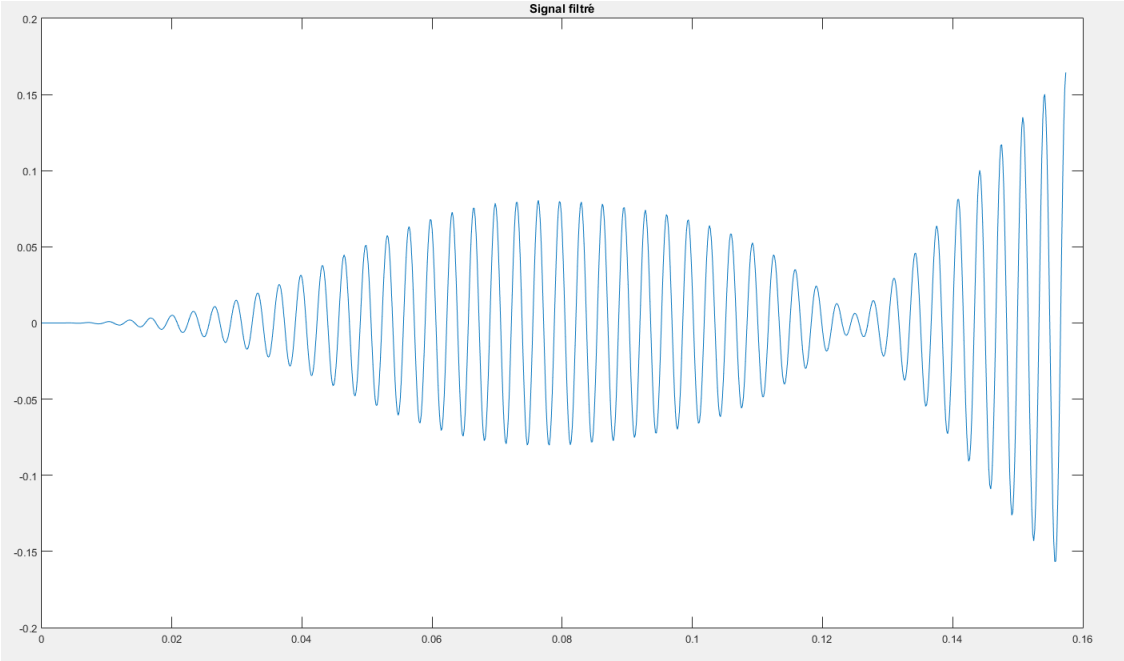


Illustration 11 : Allure du signal filtré par Butterworth RII (ordre 4)

De plus, si l'on réduit la largeur de la bande passante, on observera une meilleure qualité de filtrage. Il faut alors trouvé la bande passante adaptée pour garder toute l'information du signal.

Ainsi pour les filtres RII, le signal de sortie sera d'autant meilleur que la bande passante sera étroite autour de la fréquence à conserver et l'ordre du filtre élevé. Cependant le temps de réponse sera d'autant plus long.

- **Filtre RIF**

Pour le filtre RIF, on a donc un polynôme en z de la forme :

$$H(z^{-1}) = a(0) + a(1) * z^{-1} + \ldots$$

A la place de régler l'ordre, on paramètre dans ce cas la longueur de la réponse impulsionnelle en plus de la bande passante.

```
%Création du filtre RIF

long = 500;                 %longueur de la réponse impulsionnelle
BP1 = [1100 1300]/Fe/2;     %bande passante
b = fir1(long,BP1);
rif=filter(b,1,x);            %filtrage du signal de sortie avec a=1

figure (5)
plot(t,rif)
title('Signal filtré par le filtre RIF')

f1=abs(fftshift(fft(rif)))*Te/duree;  %transformée de Fourier du signal filtré
```

```
%Allure de la TFF

figure(6)
plot(f,f1)
title('Transformée de Fourier du signal filtré par RIF')

figure(7)
freqz(b,a)                %diagramme de Bode
```

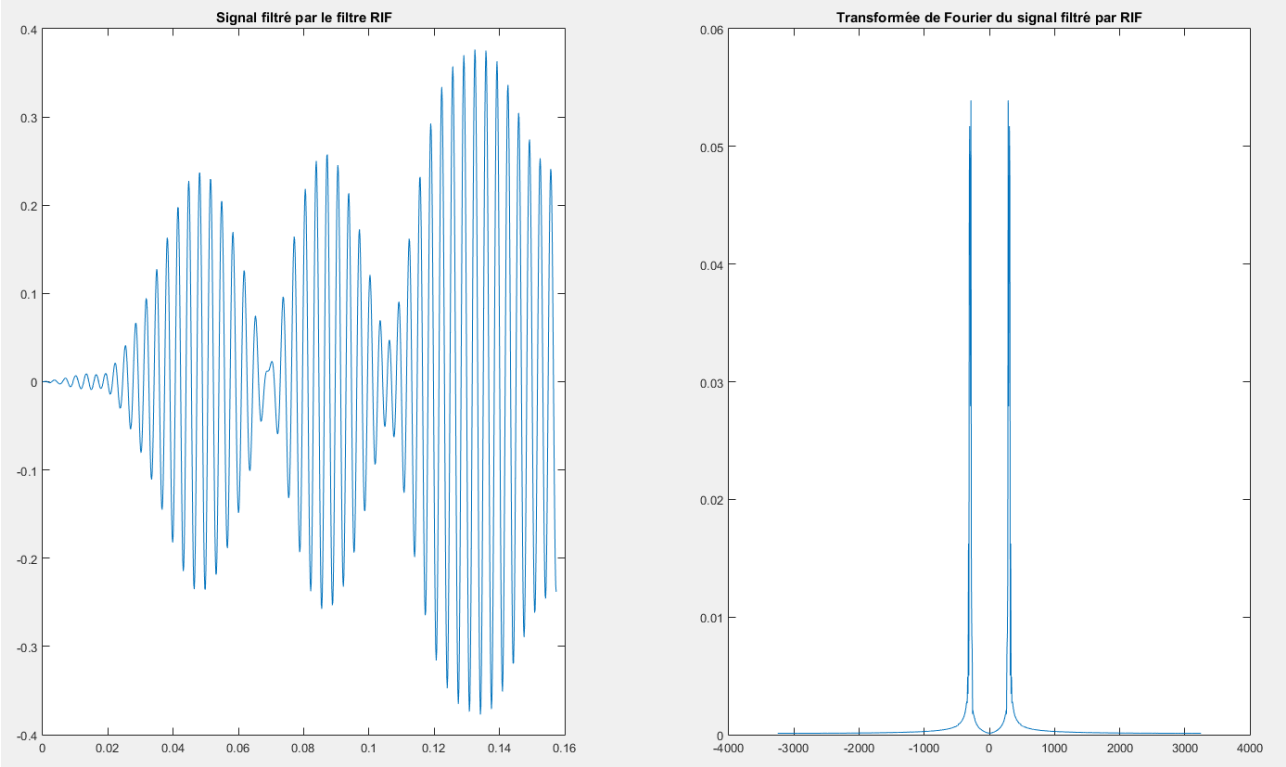


Illustration 12 : Signal et filtrage par RIF (long=500)

En modifiant la bande passante, comme précédemment, plus étroite elle est, meilleur le signal filtré est.

En ce qui concerne la longueur de la réponse impulsionnelle, si elle est trop faible, le signal n'est pas correctement filtré et le résultat est imprécis *(illustration 13)*.

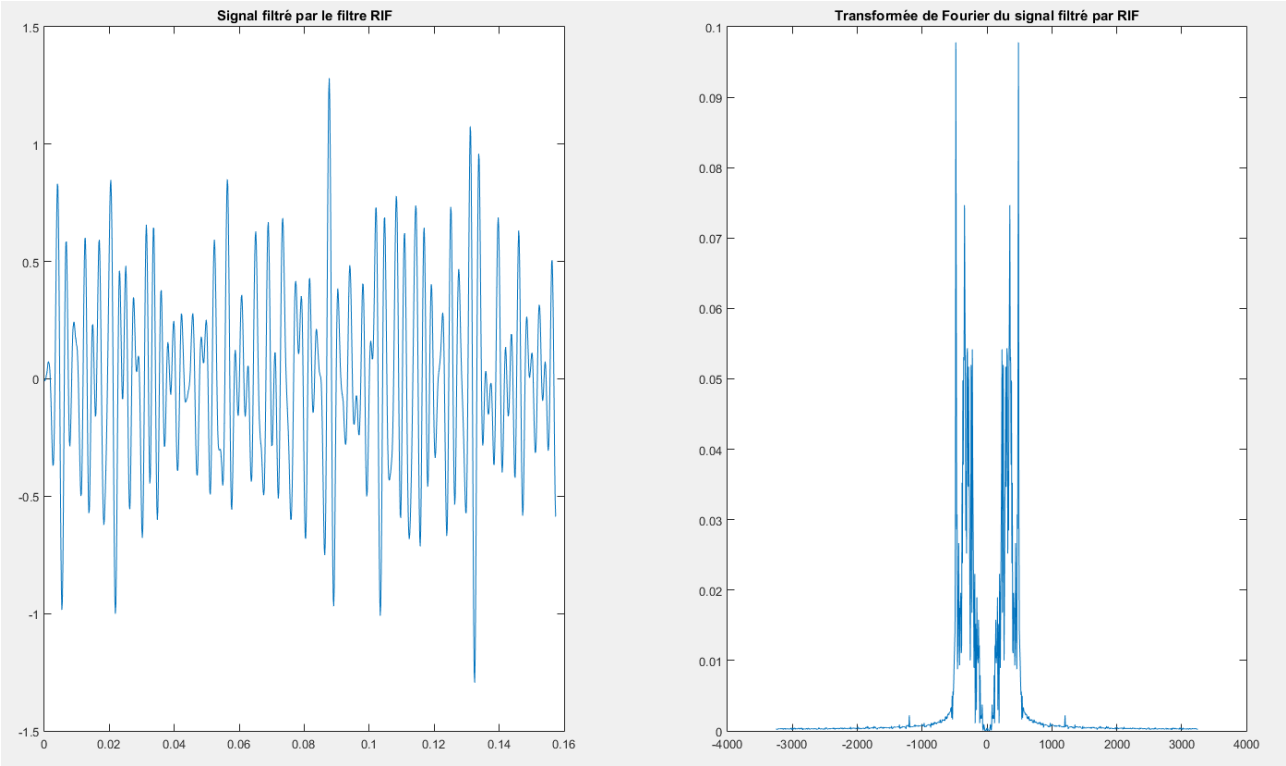
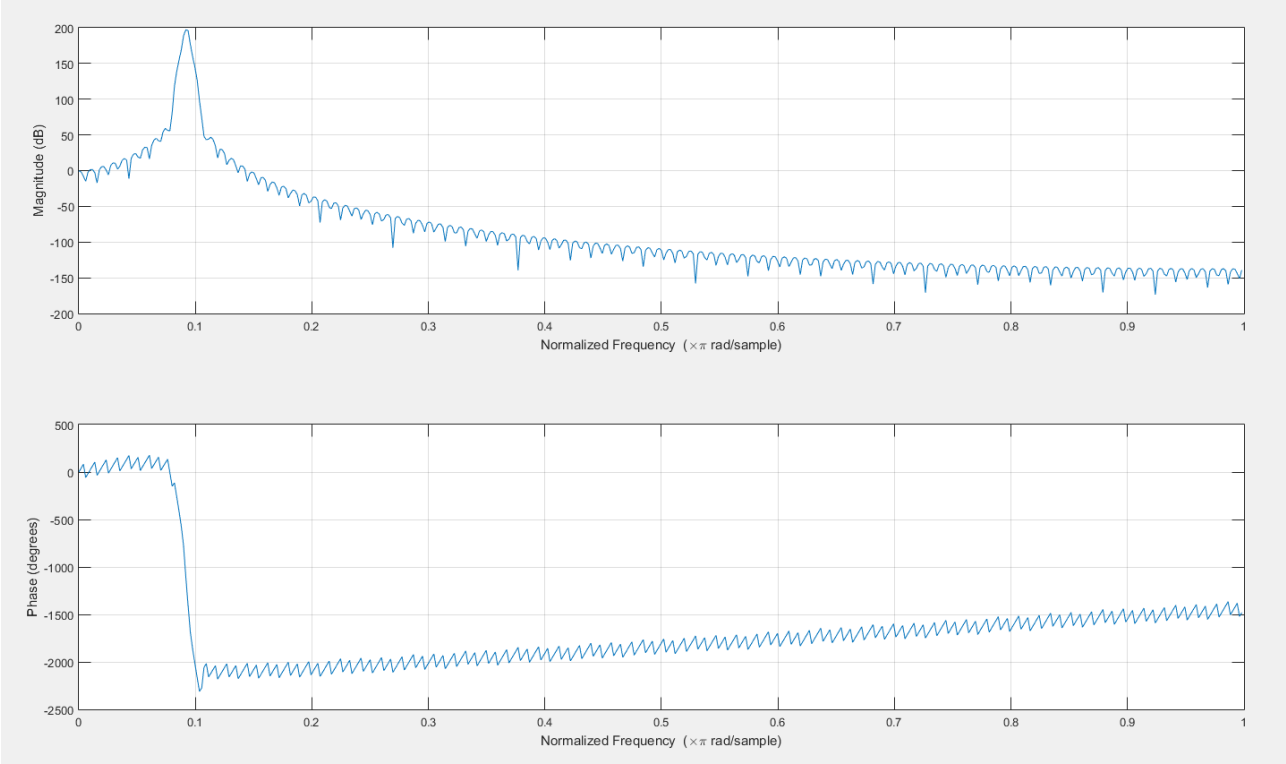


Illustration 13 : Signal et filtrage par RIF (long=50)

Avec une longueur de réponse impulsionnelle de 800, on obtient le diagramme de Bode ci-dessous.



Le temps de réponse est relativement court.

Donc pour les filtres RIF, plus la bande passante est petite, meilleur sera le signal de sortie, mais le retard sera d'autant plus élevé.

Le problème reste de trouver un bon compromis entre la longueur de la réponse impulsionnelle et la largeur de la bande passante de manière à obtenir un bon filtrage avec un temps de retard raisonnable.

CONCLUSION :

Ainsi, dans ce TP, nous avons pu mettre en évidence l'importance du choix de grande fréquence d'échantillonnage par rapport à la fréquence la plus élevée du signal pour obtenir des signaux corrects en sortie. De plus, nous avons pu constater que, quelque soit le type de filtre (RIF ou RII), le fait d'avoir une bande passante resserrée autour de la fréquence à garder est un facteur de bonne qualité du filtre, car le signal est meilleur en sortie.