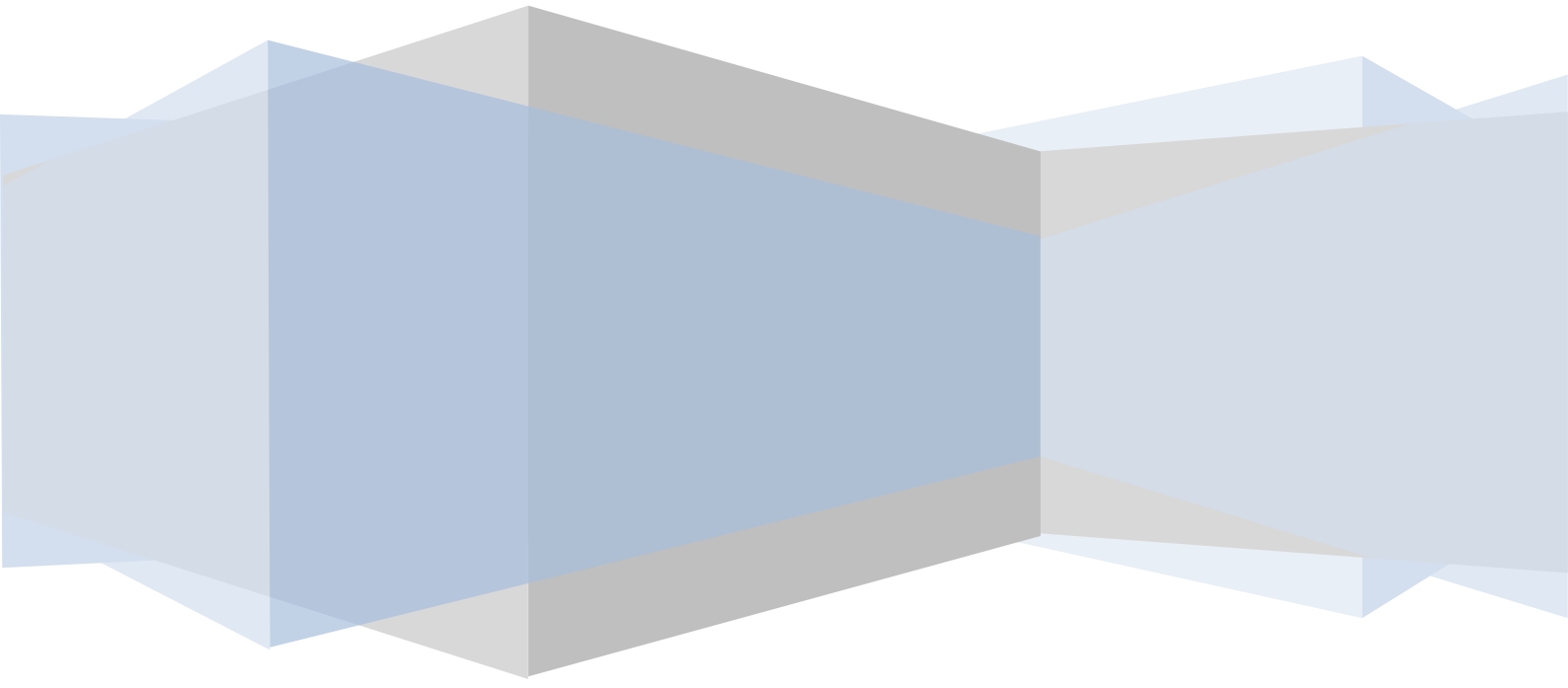
EFREI Promo 2015

Projet Système à la Fonction

Synthèse d'un banc de filtres numériques

FABRE Maxime – LEPOT Florian



Introduction

Ce projet de système à la fonction a pour objectif la synthétisation de banc de filtres numériques. Plus particulièrement la programmation de filtres passe-bas, passe-haut et passe-bande, dans le but d'extraire d'une onde ses fréquences basses et hautes par exemple. Pour venir à bout de notre objectif nous avons appris et utilisé Matlab un langage de programmation mais aussi un outil de développement puissant dans le domaine de l'électronique.

Notre rapport sera divisé en deux parties la première concernera les techniques de programmation et les synthétisations des bancs de filtres numériques. La deuxième portera sur les calculs et relevés graphiques liés aux différents synthétisations et tests fait précédemment.

Sommaire

Introduction 1
Sommaire 2
I. Réalisation du Projet : Programmation sous Matlab 3
A. Technique de programmation sous Matlab 3
B. Les différents filtres 3
C. Programmation des filtres 4
II. Analyse des résultats 7
A. Les différents spectres 7
B. Calculs liés aux filtres 7
C. Relevé des résultats et représentations graphiques 8
Conclusion 12

I. Réalisation du Projet : Programmation sous Matlab

A. Technique de programmation sous Matlab

Matlab est un est à la fois un langage de programmation et un environnement de développement. Ce langage de programmation haut niveau est assimilable très facilement. Voici quelques techniques de programmation sous Matlab qui nous ont été utiles :

Pour x = [1 2 3] et y = [4 5 6], on trouve pour les opérations suivantes :

- x.*y = [4 10 18],
- x*y' = 32,
- x * y donne une erreur.

On en déduit que la première opération est un produit terme par terme, que la deuxième est un produit scalaire et que le dernier donne une erreur.

Avec l'opération z = x + y*i, on obtient :
z = [(1.0000 + 4.0000i) (2.0000 + 5.0000i) (3.0000 + 6.0000i)]

Avec l'opération Conj(z), on obtient :
Conj(z) = [(1.0000 - 4.0000i) (2.0000 - 5.0000i) (3.0000 - 6.0000i)]

On a appliqué le conjugué car on remarque sur toutes les valeurs imaginaires ont été multipliées par – 1.

Pour calculer le module de tous les éléments d'un vecteur à valeurs complexes comme z, on utilise la fonction abs.
Cela nous donne : abs(z) = [4.1231 5.3852 6.7082]

Dans un de nos codes, lorsqu'on supprime le dernier point virgule de la ligne 2, cela affiche l'instruction (t = 0), mais le résultat final obtenu est le même.

La suppression du point virgule n'entraine pas une erreur de compilation comme les langages C et C++ par exemple, cependant la ligne où le point virgule a été supprimer est indiqué.

B. Les différents filtres

e(t) représente ici l'entrée

- **Le filtre passe bas se situe avant le dérivateur. Sa fonction de sortie est S2(t)**

On part des formules connues du TP 3 : $S2(t) = e(t) - S1(t)$

On cherche donc S2 pour le filtre passe bas, on obtient : $S2(t) = e(t) - \frac{k*(d\ S2(t))}{dt}$ car on sait que $S1(t) = \frac{k*(d\ S2(t))}{dt}$.

On dérive alors S2(t) et on obtient :

$S2(t) = e(t) - k * (S2(t) - S2(t-1)$

$(k+1)S2(t) = e(t) - k * S2(t-1)$

On en déduit que $\alpha = \frac{1}{k+1}$ et $\beta = \frac{k}{k+1}$

- **Le filtre passe haut se situe devant le dérivateur, sa fonction de sortie est S1(t).**

On part des formules connues du TP 3 : $S1(t) = k * \frac{dS2(t)}{dt}$

$S1(t) = k * \frac{d(e(t) - s1(t)}{dt}$

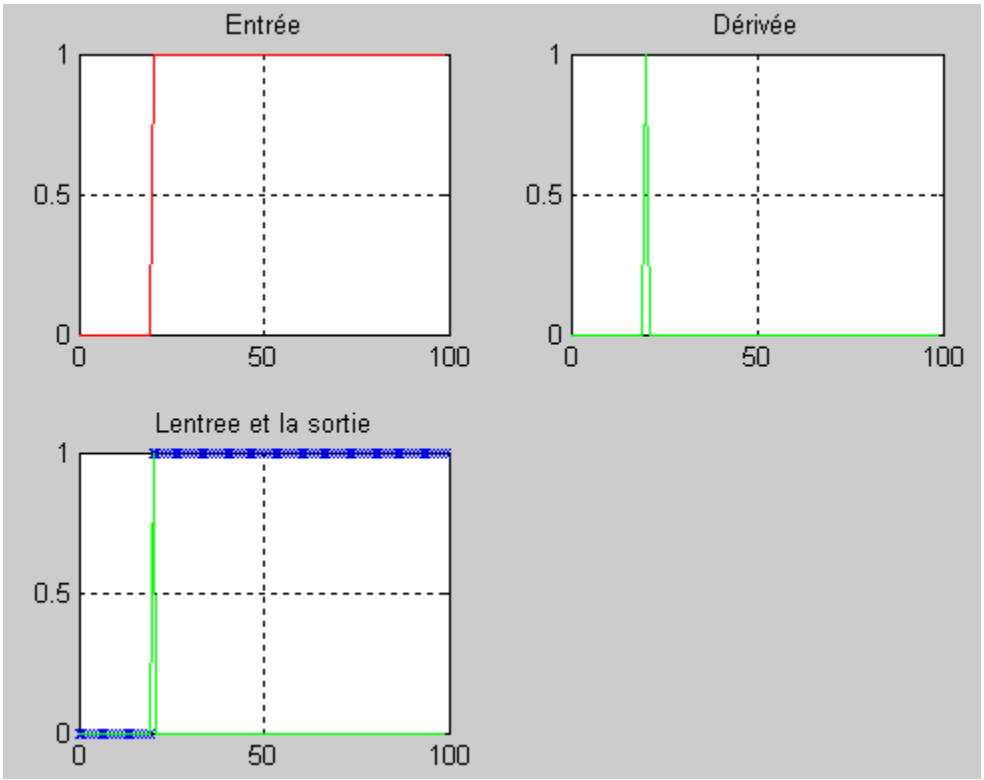
$S1(t) = k * \frac{d(e(t)}{dt} - k * \frac{S1(t)}{dt}$

$S1(t) = k\big(e(t) - e(t-1)\big) - k(-S1(t) - S1(t-1))$

$(k+1)S1(t) = k\big(e(t) - e(t-1)\big) + k(S1(t-1)$

$S1(t) = \frac{k\big(e(t) - e(t-1)\big)}{k+1} + \frac{k(S1(t-1)}{k+1}$

C. Programmation des filtres



On peut voir que l'entrée est un signal carré, et la sortie est un pic. On en conclue qu'on a programmé un filtre passe haut, et donc le calcul d'une dérivée.

Quand on programme l'entrée e de la façon suivante : `e=[zeros(1,t0) ones(1,N-t0)];` on a le même résultat qu'avant.

On passe maintenant à la programmation des différents filtres.

- **Programmation d'un passe-bas d'ordre I**

```
for t=2:N    s(t)=(1/(k+1))*e(t)+(k/(k+1))*s(t-1);   end
```

- **Programmation d'un passe-haut d'ordre I**

```
for t=2:N    z(t)=(k/(k+1))*(e(t) - e (t-1))+(k/(k+1))*z(t-1);   end
```

k représente comme vu précédemment une constante. La réponse indicielle est la réponse d'un système dynamique à une fonction marche de Heaviside sigma (t) communément appelée échelon.

Réponse indicielle selon les valeurs de k pour le filtre passe-haut :

- Pour k = 10, la réponse indicielle est 31,
- Pour k = 5, la réponse indicielle est 26,
- Pour k = 1, la réponse indicielle est 23.

Plus k est petit, plus la réponse indicielle sera basse.

Réponse indicielle selon les valeurs de k pour un filtre passe-bas:

- Pour k = 10, la réponse indicielle est 30,
- Pour k = 5, la réponse indicielle est 25,
- Pour k = 1, la réponse indicielle est 21.

Plus k est petit, plus la réponse indicielle sera basse.

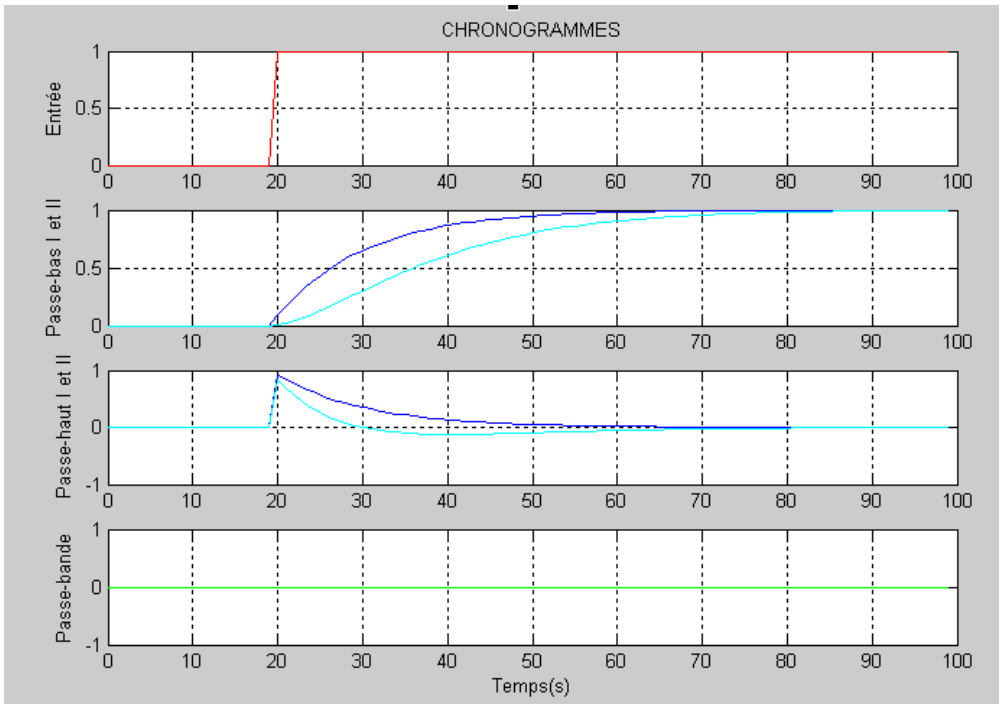
On a synthétisé un filtre passe bas 2eme ordre en prenant pour modèle la mise en cascade des 2 filtres de premier ordre. Voici la modélisation des sorties de ce filtre :

- **Programmation d'un passe-bas d'ordre II**

```
 for t=2:N    ss(t)=(1/(k+1))*s(t)+(k/(k+1))*ss(t-1);   end
```

- **Programmation d'un passe-haut d'ordre II**

```
for t=2:N    zz(t)=(k/(k+1))*(z(t) - z (t-1))+(k/(k+1))*zz(t-1);   end
```



- Pour k = 10, on calcule la réponse indicielle du filtre **passe haut** et on obtient 33,
- Pour k = 10, on calcule la réponse indicielle du filtre **passe bas** et on obtient 51.

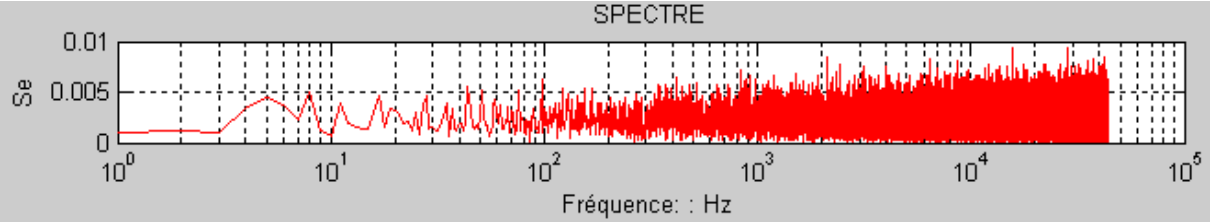
II. Analyse des résultats

A. Les différents spectres

Représentation d'un bruit blanc et d'une impulsion :

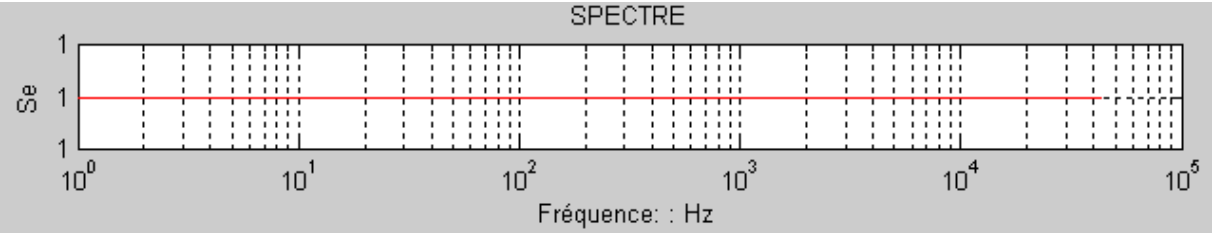
- **Pour le bruit blanc**

Son spectre présente de nombreux pics qui s'intensifient au fur et à mesure. En ce qui concerne la perception auditive de ce signal, nous entendons un grésillement qui s'étend sur le temps.



- **Pour une impulsion**

Son spectre est constant au fil du temps. En ce qui concerne la perception auditive du signal, on entend 2 « bips » séparés par un « vide ».



B. Calculs liés aux filtres

On sait que $f = \frac{1}{T}$. Donc $T = \frac{1}{f}$. Ainsi, pour une seconde, on obtient 1 Hz.

Donc pour une fréquence de 44,1 kHz, il nous faudra 44100 échantillons pour représenter le signal sur une fenêtre d'une seconde.

La fréquence maximale des signaux que nous pourrons produire ou traiter sera donc, d'après le théorème de Shannon : $fmax = \frac{fe}{2} = \frac{44100}{2} = 22050\ Hz$.

Concernant les filtres plus particulièrement :

Chaque équation d'un filtre est composée de différentes constantes de temps.

On sait que :

$k = \frac{1}{\omega c}$ $k1 = \frac{1}{q*\omega 0}$ $k2 = \frac{q}{\omega 0}$ $q = \frac{\omega 0}{\Delta\omega}$

On remplace alors ω par $2\pi\Delta f$:

$$q = \frac{\omega 0}{2\pi\Delta f}$$

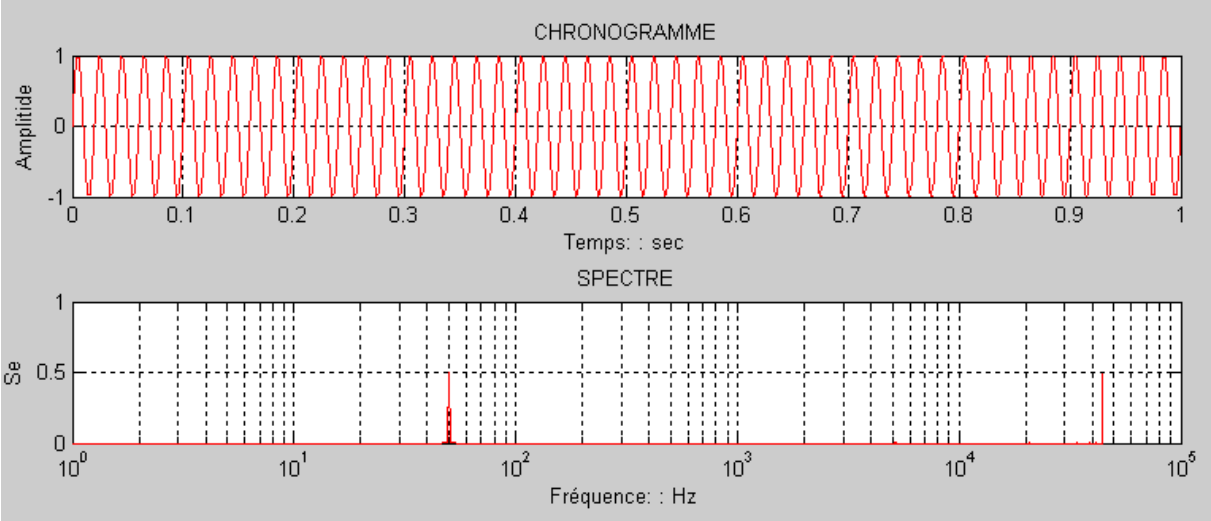
$$k1 = \frac{2\pi\Delta f}{\omega 0^2} = \frac{1}{\frac{\omega 0^2}{\Delta\omega}} = \frac{1}{\frac{\omega 0^2}{2\pi\Delta f}} = 1$$

$$k2 = \frac{1}{2\pi\Delta f} = \frac{fc}{\pi\omega} * \frac{1}{\omega 0} = \frac{1}{2\pi fc} = 0{,}0079$$

$$k = \frac{1}{2\pi\Delta f} = 1{,}5915 * 10^{-4}$$

C. Relevé des résultats et représentations graphiques

On normalise la fréquence car elle a pour formule $sin(2pif * Te)$. Te Correspond à l'intervalle de temps des points qu'on veut normaliser. On divise donc la fréquence réelle par le nombre d'échantillons.



Cette courbe représente le sinus à une fréquence réelle de 20Hz.

f(Hz)	20	50	100	500	1000	2000	5000	15000	20000	40000	50000
fB(Hz)	20	50	100	499	1000	2000	5000	15000	20000	40000	38200
fH(Hz) ($x10^3$)	44.08	44.05	44	43.6	43.1	42.1	39.1	29.1	24.1	4	5.9
Audible	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y

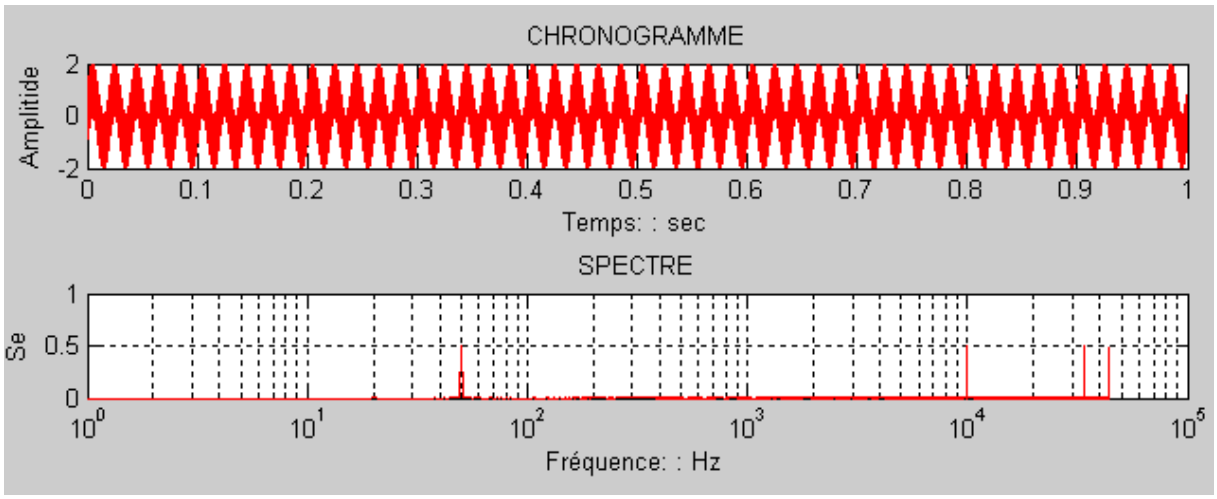
- fB représente la fréquence basse

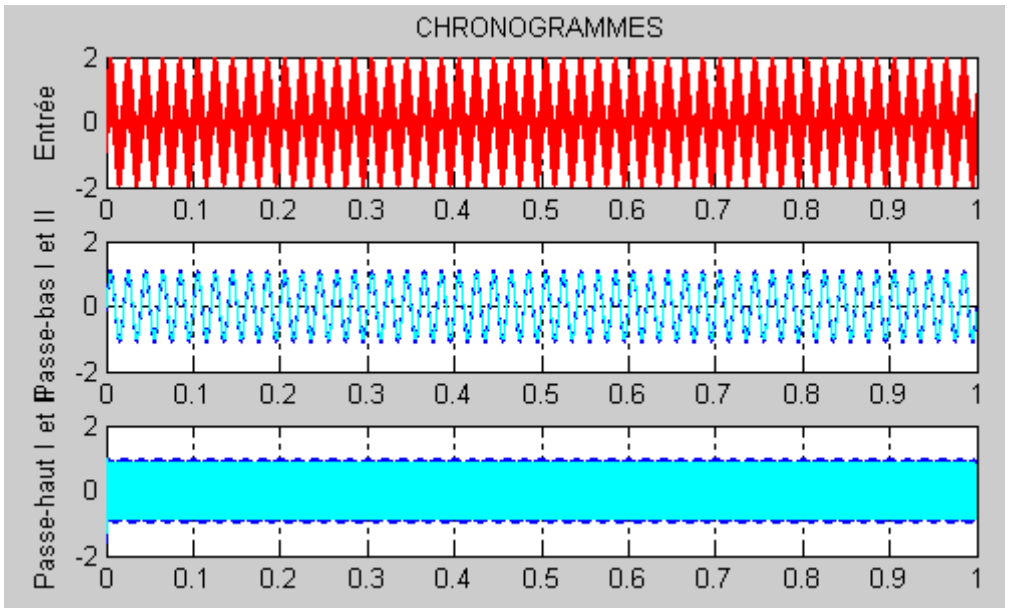
- fH représente la fréquence hausse
- Si on additionne fB et fH, on obtient la fréquence additionnelle, donc le nombre d'échantillons.

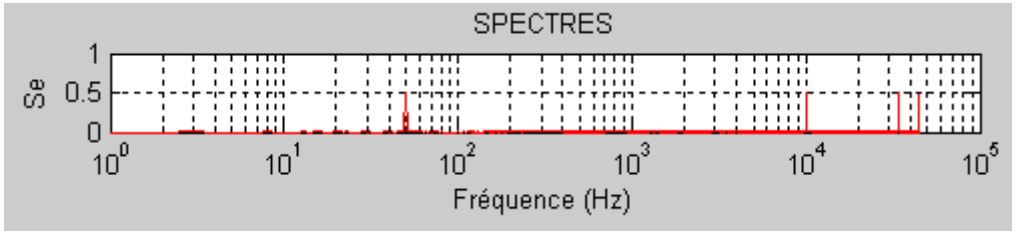
On attaque les filtres passe-bas et passe-haut (ordre 1 et 2) avec un signal composite de la forme :

e(t)=sin(2pf1t)+ sin(2pf2t) avec f1=50Hz et f2=10kHz.

On obtient les courbes et spectres suivant.







On programme un filtre passe-bas de 1er ordre de Butterworth, de fréquence de coupure 400Hz. Avec une normalisation des fréquences à N/2.)

Le tableau est : `[B,A] = butter(n,fn,'low');`

Tableau consignant les coefficients du filtre :

B	0.0277	0.0277
A	1.0000	-0.9446

La relation de récurrence entre l'entrée et la sortie est : `s = filter(B,A,e);`

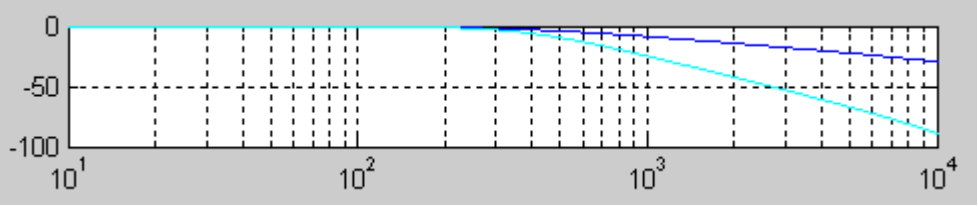
On procède à un filtrage du 2ème ordre.

Tableau consignant les coefficients du filtre :

A	1.0000	-0.9446	
B	0.0277	0.0277	
D	0.0008	0.0016	0.0008
C	1.0000	-1.9194	0.9226

```
clc; clf;
N=44100; e=zeros(1,N); e(5)=N;
%--------------------------------Synthèse passe-bas 1
fcoupure=400;
fn=2*fcoupure/N; %---Fréquence normalisée
n=1; [B,A] = butter(n,fn,'low')
s = filter(B,A,e);
Fs=fft(s); Ss=sqrt(Fs.*conj(Fs))/N; freq=0:(N-1);

%--------------------------------Synthèse passe-bas 2
n=2;
[D,C] = butter(n,fn,'low')
ss = filter(D, C, s);
Fss=fft(ss); Sss=sqrt(Fss.*conj(Fss))/N;
subplot(4,1,1);
semilogx(freq,20*log10(Ss),'b',freq,20*log10(Sss),'c');
grid; ylabel ('Passe-bas I et II'); xlim([10 10000]);
```

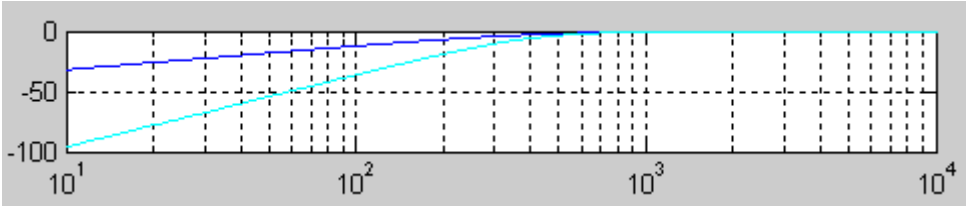


- N = 2 car nous sommes dans le second ordre.
- La sortie du passe-bas 1 (s) devient l'entrée du passe-bas 2 de sortie (ss).

Pour la synthèse des filtres passe-haut, il suffit de modifier dans la définition des tableaux :

- `[B,A] = butter(n,fn,'low');`
- `[D,C] = butter(n,fn,'low');`

En remplaçant `low` par `high`, on obtient alors le graphique suivant :



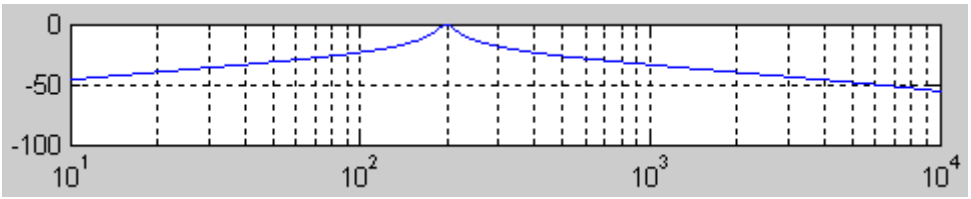
Les filtres passe-bande (ou coupe-bande) présentent deux fréquences de coupures.

```
%--------------------------------Synthèse passe-bande
n=1;
LB=[190 210]; LBn=2*LB/N;
[V,U] = butter(n,LBn,'bandpass');
z = filter(V, U, e);
Fz=fft(z); Sz=sqrt(Fz.*conj(Fz))/N;
subplot(4,1,3); semilogx(freq,20*log10(Sz),'b');
grid; ylabel ('Passe-bande'); xlim([10 10000]);
%-------------------------------------------------------Fin ;-)
```

Tableau consignant les coefficients du filtre :

V	0.0014	0	-0.0014
U	1.0000	-1.9963	0.9972

- La relation de récurrence entre l'entrée et la sortie est : `z = filter(V, U, e);`



Conclusion

Grâce à ce projet, nous avons pu découvrir et prendre en main le logiciel Matlab qui est un environnement de développement et programmer avec celui-ci les 3 filtres : Passe-haut, Passe-bas et passe bande.

Nous nous sommes ensuite intéressés à des signaux échantillonnés pour nous permettre d'établir un lien entre réponse impulsionnelle et réponse fréquentielle des filtres.

Nous avons fini par déterminer les coefficients la réponse du système temporel et nous avons trouvé les relations de récurrence entre l'entrée et la sortie dans les différents pour les différentes sortes de filtre.