SPILERS Lise
BLANC Monica-Pauline

EFREI

L1-INT2

01/06/2020

System to Function : Project

INTRODUCTION

Nowadays, a signal is composed of a lot of spurious waves that is not necessary if we want to use this signal. Moreover, when we want to transfer an information in a long distance, the most complaining problem is the loss of a parts of it. How can we solve those problems?

In order to correct these problems, we can use amplifier or filters, and this is what we are going to focus on: fitters. Low, high or band pass, filters are electronics components used for treatment of a signal. They can attenuate several components as interfering waves of low or high frequencies (=filter) and pass others in order to have just the desired information. It's really useful in our society

However, before these labsession, we studied them in Multisim (in a theoretical manner) in order to understand how they function using electronic components but now we are going to see the practice with mathematical expressions.

So, so as to do it we will use a new software: MatLab. It will help us to test the effect of our different filters.

PS: During all the report, you can see numbers at the top of the words as the same if it was a footnote, but it isn't, it's just the number corresponding to the picture in the Annexe. Moreover, we put some Matlab file with this pdf because sometimes it was important to show the program, but we know that our report is very long, so we didn't put it in it.

I. Introduction to MATLAB

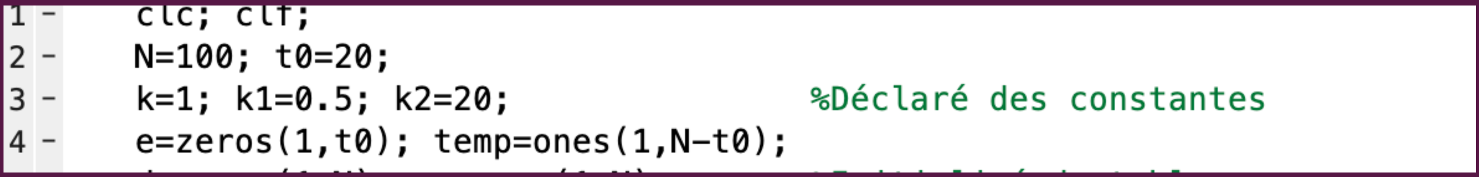
Before programming the different filters, we had to be more comfortable with MATLAB, to do so we started to open a file already operational which corresponds to the program of the derivative[1] :

See on the annex 2

Figure 1: we see the input which is equal to 1 and the output which corresponds to the computation of the derivative and it's equal to 0. We already know that the derivative of a constant is 0 so the program has worked well.

Figure 2: It's just a graph of the input, just to see it in time.

Then we changed the e value with the following input:



```
clc; clf;
N=100; t0=20;
k=1; k1=0.5; k2=20;                    %Déclaré des constantes
e=zeros(1,t0); temp=ones(1,N-t0);
```

What we oberve ?

Changing the e value change anything, it's the same as the line 8.

Finaly in order to understand well how we code in MATLAB, we eliminate the semi -colon of line 2 to see what it changes. Our hypothesis is that MATLAB will not work because of the importance of it (example in C, where semi-colons are very important).

Our hypothesis was wrong. MATLAB worked and it shows something else, the console shows instructions and results: in the command, we see the value of t_0. So what we understand is that in MATLAB, ";" is useful for showing a result or instructions according to a variable.

II. Programmation of different filters

a) Low Pass filter

- FIRST ORDER

Analyzing this circuit[3], we can obtain the equation of the output s(t) :

$$s(t) = \frac{e(t)}{1+k} + \frac{k \times s(t-1)}{1+k}$$

$$\Leftrightarrow s(t) = \frac{1}{1+k} e(t) + \frac{k}{1+k} s(t-1)$$

Thus, we find α and β, we obtain: $s(t) = \alpha e(t) + \beta s(t-1)$

We type the equation above in MATLAB and we apply different k chose randomly (1,10,50) in order to see how does the low pass filter work. *See obtained grpahs[5]*

We measure the slope drawing a line (the pink one):

k	**1**	**10**	**50**
Slope	1/2s	1/10s	1/40s

The relationship between k and the cut-off frequency of the filter is that when k increases, smaller is the slope: the high frequency are filtered that corresponds to big variations helps us to conclude that when the cut-off frequency decreases, k increases.

- <u>SECOND ORDER</u>

A first order low pass filter can be "defined" as the first derivative. Consequently, in order to have a second order, we have to compute the second derivative. To do so we need two low pass filter of first order, choosing the input for the second order to be the output of the first order: we replace e(t) by s(t) and s(t) by ss(t) (we keep the same alpha and beta).

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

We obtain: $ss(t) = \alpha s(t) + \beta ss(t-1)$ (We can see this equation in Matlab[5])

What we can see in the graph[6] that the low pass filter of the second order attenuate less the lows frequencies than the first order but the main difference is in the high frequency: in the second order, high frequencies are more filtered than the first order.

b) High Pass filter

- <u>FIRST ORDER</u>

Some calculation to find the output of the high filter z(t) :

$z(t) = k * \frac{ds(t)}{dt} \Leftrightarrow z(t) = k * \frac{d\left(e(t)-z(t)\right)}{dt} \Leftrightarrow z(t) = k * [\frac{de(t)}{dt} - \frac{dz(t)}{dt}] \Leftrightarrow z(t) = k * [e(t) - e(t-1-z(t) + z(t-1)]$

$\Leftrightarrow z(t) * (k+1) = k * [e(t) - e(t-1) + z(t-1)$

$\Leftrightarrow \mathbf{z(t) = \frac{k}{k+1} * [e(t) - e(t-1) + z(t-1)]}$ (see the equation in Matlab[8])

We type the equation above in MATLAB and we apply different k chose randomly (1,10,50) in order to see how does the high pass filter work. *See obtained graphs*[9]

We measure the slope drawing a line (the pink one):

k	**1**	**10**	**50**
Slope	-1/2s	-1/13s	-1/30s

The output signal is taken from across the resistor. The relationship between k and the cut-off frequency of the filter is that when k increases, smaller is the cut-off frequency and bigger is the response time (which correspond to the time it takes to reach the finale value which happens to 0).

- SECOND ORDER

A first order high pass filter can be "defined" as the first derivative. Consequently, in order to have a second order, we have to compute the second derivative. To do so we need two high pass filter of first order, choosing the input for the second order to be the output of the first order : we replace e(t) by z(t) and z(t) by zz(t) (we keep the same alpha and beta).

$$\boldsymbol{z(t) = \frac{k}{k+1} * [e(t) - e(t-1) + z(t-1)]}$$

We obtain: $zz(t) = \frac{\boldsymbol{k}}{\boldsymbol{k+1}} * [\boldsymbol{z(t) - z(t-1) + zz(t-1)}]$ **(see the equation in Matlab[10])**

What we can see in the graph[11] that the high pass filter of the second order attenuate more the lows frequencies than the first order but the main difference is in the lows frequencies: in the second order, lows frequencies are more filtered than the first order.

c) Bandpass filter

Analyzing this circuit, we can deduce two equations one for the input of the first differentiator y'(t) and one for the output y(t):

$$y(t) = k1([y'(t) - y'(t-1)]$$
$$y'(t) = \mathrm{e(t)– y(t) - k2[y(t) - y(t-1)]}$$

We replace the y'(t) in y(t):

$$y(t) = k1\{[e(t) - y(t) - k2(y(t) - y(t-1)] - [e(t-1) - y(t-1) - k2(y(t-1) - y(t-2)]\}$$

$$y(t) = k1e(t) - k1y(t) - k1k2y(t) + k1k2y(t-1) - k1e(t-1) + k1y(t-1) + k1k2y(t-1) - k1k2y(t-2)$$

We manage to have something for y(t) and we find:

$$\mathbf{y(t)} = \frac{\mathbf{k1}}{\mathbf{1 + k1 + k1k2}}[\mathbf{e(t) - e(t-1)}] + \frac{\mathbf{k1 + 2k1k2}}{\mathbf{1 + k1 + k1k2}}\mathbf{y(t-1)} - \frac{\mathbf{k1k2}}{\mathbf{1 + k1 + k1k2}}$$

(See the equation on Matlab[11])

We are going to focus to the response of the filter such as k1=0,1 and k2=20. We already know that: k1=$\frac{1}{q\omega 0}$ and k2=$\frac{q}{\omega 0}$.
So if search q we have to look at the graph and see the center, and the initial and final value:

We already know that k1=$\frac{1}{q\omega 0}$ and k2=$\frac{q}{\omega 0}$

⇔ $q\omega 0 = \frac{1}{k1}$ (1) and q = k2 × $\omega 0$ (2), we can replace q (1) that we find in (2)

⇔ k2 × $\omega 0^2 = \frac{1}{k1}$ ⇔ $\omega 0^2 = \frac{1}{k1.k2}$ ⇔ $\omega 0 = \sqrt{\frac{1}{k1.k2}}$ ⇔ $\omega 0 = \sqrt{\frac{1}{0{,}1\times 20}} = \frac{1}{\sqrt{2}}$

In order to have q, we replace $\omega 0$ in it
(2) q = k2 × $\omega 0$ ⇔ q = 20 × $\frac{1}{\sqrt{2}} = \frac{20}{\sqrt{2}}$

Moreover, what we can see in this graph[14] is that the signal is a attenuated sinusoidal signal progressively converging towards 0. It's pseudo-period seems to be 20 seconds. Testing different q, we see that q has an impact in the response: increasing q it makes the response decrease.

III. Digital filtering

a) Signal observation and spectral analysis

We start by looking how vectors work in Matlab, we have to vecotrs x=[1,2,3] and y=[4,5,6]

- x.*y = [4 10 18] → product (term by term)
- x*y' = 32 → scalar product
- x*y = error

If z=x+yi, we have z= [1+4i 2+5i 3+6i] and we have in Matlab with conj(z) = [1-4i 2-5i 3-6i]. In Matlab the conj(z) is multiplied by -1.
If we want the magnitude of all vector elements of a complex, we can use abs(z) or norm(z). The main difference is that norm is the magnitude of all vectors and abs is the magnitude for each vector:

```
abs(z) = 4.1231    5.3852    6.7082
norm(z) = 9.5394
```

Since the frequency is 44,1 Hz, we have 44 100 samples per second. Thus, N = 44 100.
We can process without loss, according to Shannon criterium, a f_{MAX} of 22 050 Hz
This is bevause of the shannon's condition that says that: $f_{max} = \frac{Fe}{2}$ (here f_e is 44 100 Hz)
We normalize the frequencies because the frequency of a sampled wave depends not just on the frequency of the original analog wave, but also on how frequently you sample it. Normalized frequency is actually unit of measurement of frequency equivalent to cycles/sample. We know that continuous time variable is t in second and with normalized, this time variable is replaced by the discrete integer variable, N, with units of samples. The computer processes two signals (the signal and the sampling frequency are proportional)with the same coefficient. More specifically, what matters is not how fast the original analog signal varies, but how fast it varies with respect to the sampling period.

f(Hz)	**20**	**200**	**1000**	**10 000**	**15 000**	**16 000**	**17 000**	**18 000**	**20 000**	**40 000**	**50 000**
Fl(Hz)	20	200	1000	10 000	15 000	16 000	17 000	18 000	20 000	4103	5910
Fh(Hz)	44,08	43,08	43,02	34,04	29,02	28,08	27,10	26,11	24,03	39,91	38,10
Audible	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N

We notice that when we are in the Shannon's conditions, fl = f and fh=f_e-f.

For f = 40 000, f_e is approximatively equal to f + fl and for 50 000, f= f_e-f. (all in Hz).

We had to program a white noise[15] and an impulsion[16]

b) Filter synthesis

We know that $\omega = 2\pi \vartriangle f$, So :

- $k = \frac{1}{\omega_c}$ ⇔ $k = \frac{1}{2\pi f_c}$
- $k1 = \frac{1}{q \times \omega_0}$ ⇔ but q=$\frac{\omega_0}{\Delta\omega}$ so $k1 = \frac{\Delta\omega}{\omega 0^2}$ ⇔ $k1 = \frac{2\pi f}{(2\pi f_0)^2}$
- $k2 = \frac{q}{\omega 0}$ ⇔ but q=$\frac{\omega_0}{\Delta\omega}$ so $k2 = \frac{\omega 0}{\omega 0 \times \Delta\omega}$ ⇔ $k2 = \frac{1}{2\pi \Delta f}$
- $q = \frac{\omega_0}{\Delta\omega}$ ⇔ $q = \frac{2\pi f_0}{2\pi \Delta f}$

The advantage of this method (eliminating the intermediate outputs and express as a function of the input e(t),e(t-1)… and its previous values s(t),s(t-1) …) allows us to reduce errors when its coding at hand. Moreover, this is advantageous because we have to just modify the input e(t) if we want to change the signal, without caring to the outputs (as they don't exist).

c)Application to audio-frequencies signals

Using the composite signal e(t)=sin($2\pi f1t$) + sin($2\pi f1t$) with f1=50hz and f2=10kHz, we have four chronogram[16] and spectrum[18]

We got a peak of response around 50Hz at the input and the output because one of the signal frequencies is at 50 Hz but we can also see that there is no filter that filter well the frequencies. We can conclude with the graph that the low pass filter is more efficient in attenuating the 10kHz frequencies and the high pass more in the 50hz. For the band pass, it isn't optimal either and it's center is around 2kHz. Moreover, when we heard the output of the second order low pass, we heard a really low-pitched sound but when we heard the output of the first order high frequency we heard really high pitched sound
With those graphs we can see that the input (in red) of 50Hz is composed of alternated patterns of 10kHz. The High pass is more with the pattern of 10kHz but the low pass more with the 50Hz. Moreover, the bandpass is conserving one parts of this pattern.

Then we replace the input with an uniform white noise signal (command in Matlab : e=rand(1,N)-0,5)*2))[19]

We know that the white noise is composed of all frequencies and looking at the graph we can see that the filter of second order attenuates more than before. Moreover, we can see that low pass attenuates more high frequencies and high pass more the low.

d) Impulse response

We generate an impulse signal using (e=zeros(1,N) and e(10)=N) [20]

When we look at these graphs we that the high pass attenuates more the impulsion than low pass. We can see that the pseudo oscillations of the bandpass is approximatively 0,0005.
Moreover, we see that the input signal is constant so the spectrum of these different filters are really close to their bode plot.
The cut-off frequency is at + or - close to 3dB (for the low pass is $\approx 700 - 800Hz$ and for the high pass it's $\approx 900 - 1000Hz$). The center frequency of the bandpass is 2kHz which respects the value we chose at the beginning.

IV. Digital filtering synthesis

a) Low pass filtering

The coefficient are :

- B= 0,0277 0,0277
- A= 1,000 -0,9446

The arrays is [B,A] = butter (n,fn, 'low') and the recurrence equation between the input and the output of this filter (here a low pass) is s(t) of the low pass we uses in Lab1.

Adding the following lines, we have :

- C= 0,008 0,0016 0,008
- D= 1,000 -1,9194 0,9226

The recurring equation is ss(t) (saw also in Lab2).

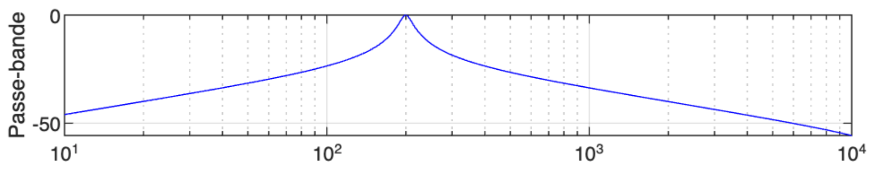
In order to have High pass we just have to change the s(t) in z(t), the ss(t) in zz(t) so as to use the recurrence equation that we used in Lab1 and change the 'low' in 'high'.[21]

For the recurrence equation is what we find in lab1 with y(t) .Completing the program we have :

In both graph dark blue is for the first order and cyan for the second. The first is for the low pass and the second graph is for the high pass. As we said in lab2, theses graphs are correspond to their bode plot. For the order, we saw that bigger is the degree (n=1,n=2, …) bigger is the slope.

b) Bandpass filtering

A bandpass is a filter who pass frequency within certain range and reject the other frequencies (those which are outside of this range) : this is what we can see in the following graph. Also his bode plot is very similar at the ideal one.



Subsituying the imulse singla by white noise and run the program[22]

The white noise spectrum that appears has a similar graph as the impulsion after it went through the bandpass

CONCLUSION

To conclude, we see a new software : MatLab. It allows us to show how to realize concretely different type of filters, that we saw in course, of different orders and not with electronic components as we did in Multisim but with mathematical expressions.

Moreover, we could try our filter with real filtered sound and not with an oscillator: thanks to it, it allows us to understand more, their functioning but also the usefulness of it (for example that the low pass saved our ears filtering high frequencies)

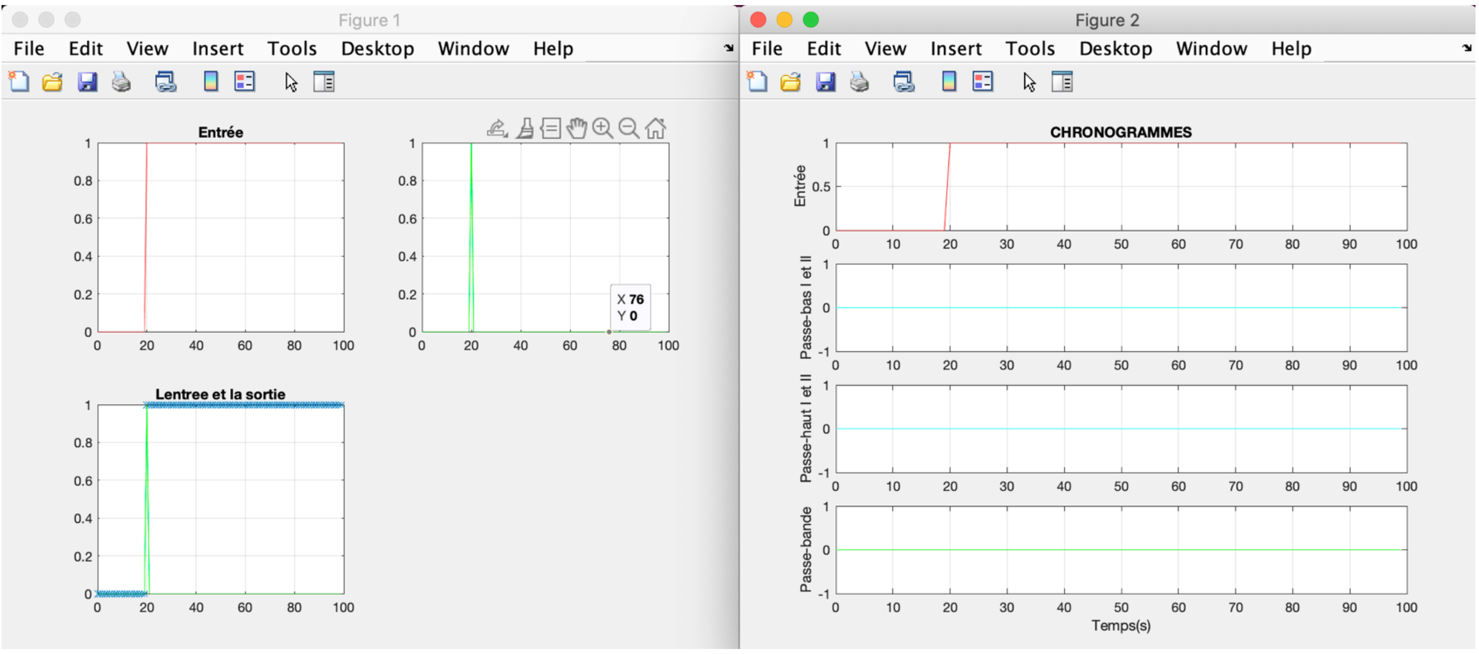
Lise and I hope you will have as much pleasure to read this report as we had to do the TP and write it. ☺

Annexes

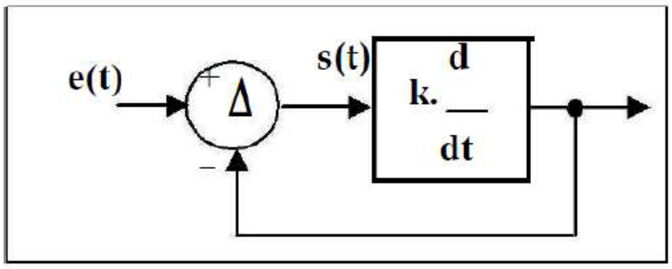
```
%---------------Programe d'une dérivée----------------

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

1. **The program in Matlab for the “dérivée”**



2. **What we obtain with this program**



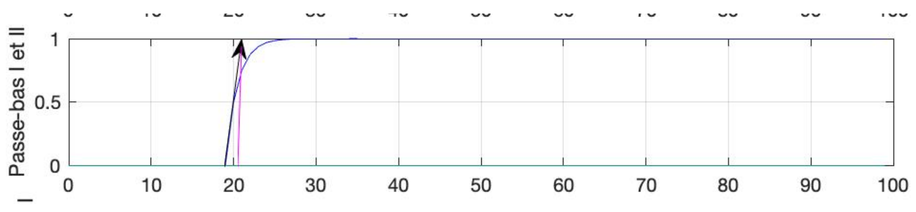
3. **Scheme of a low pass filter of the first order**

4. **The program of the low pass of first order in Matlab**

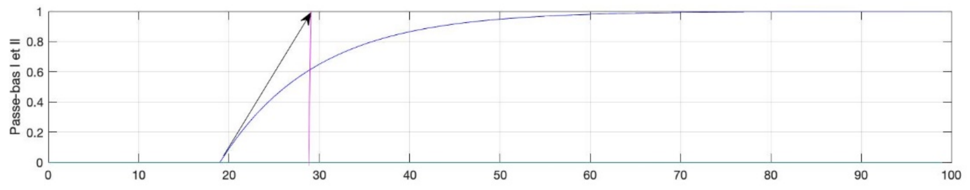
```
---------------Programmation d'un passe-bas d'ordre I------------------------

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

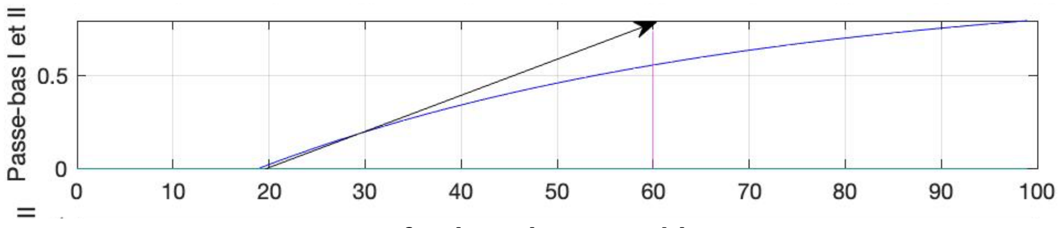
5. Graphs of lowpass filter of first order



K =1



K =10



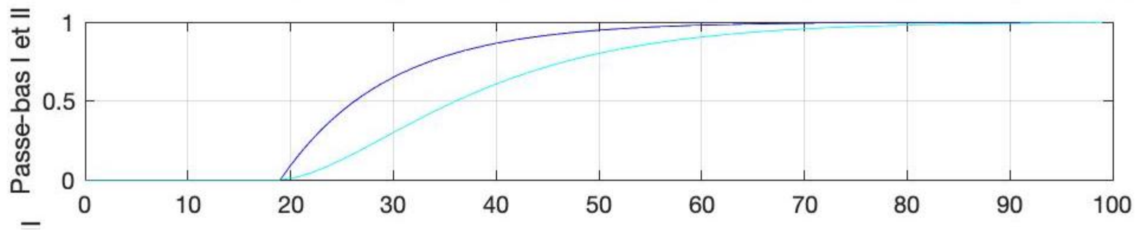
K =50

for k=1, k=10 and k=50

6. The program in Matlab of the low pass of the second order

```
----------------Programmation d'un passe-bas d'ordre II-------------------------
                for t=2:N   ss(t)=[1/(k+1)]*s(t) + [k/(1+k)]*ss(t-1); end
```

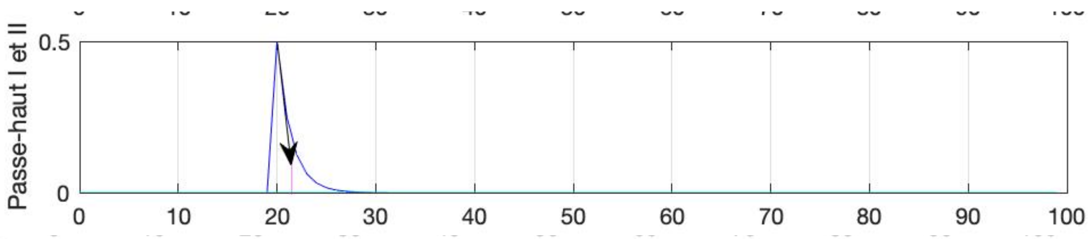
7. Graphs of lowpass 2nd order (first order and second order)



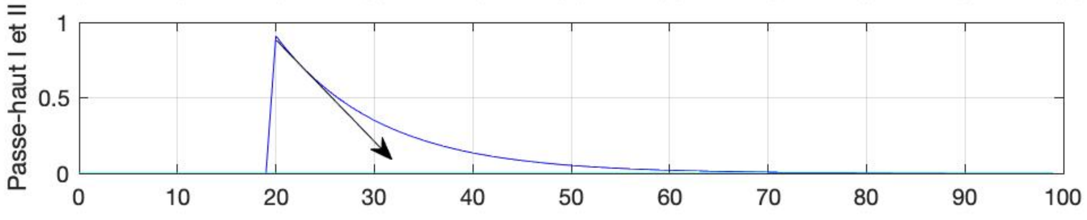
8. Equation in Matlab the high pass of the first order

```
-----------------Programmation d'un passe-haut d'ordre I------------------------
                 for t=2:N    z(t)=(k/(k+1))*[e(t)-e(t-1)+z(t-1)]; end
```

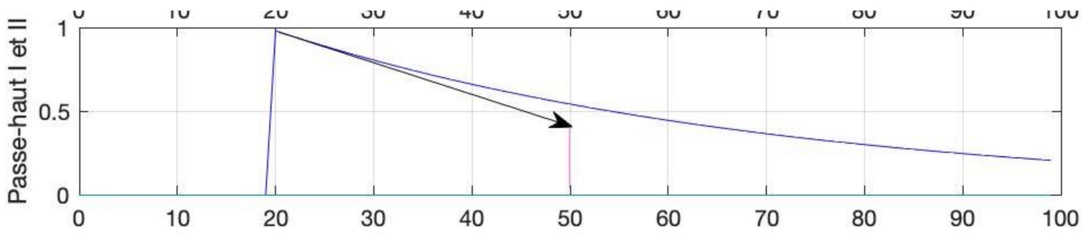
9. **Graph of High pass of first order**



K =1

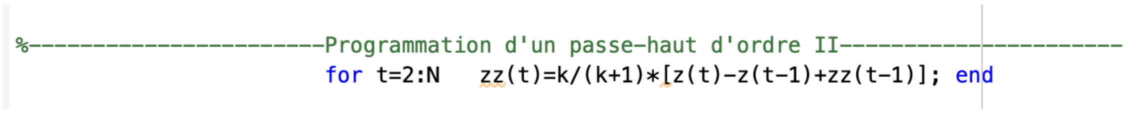


K =10



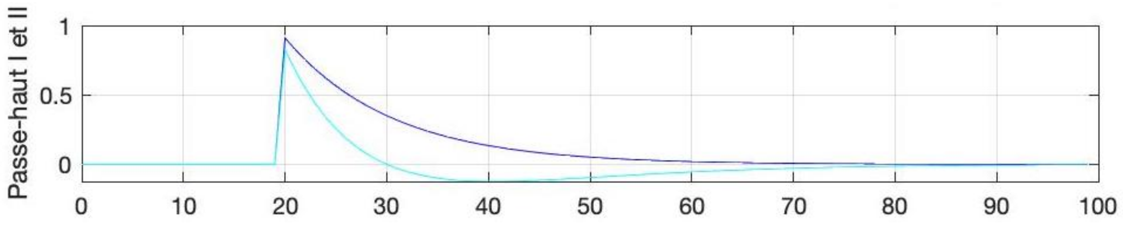
K =50

10. **The equation of second order (high pass) in Matlab**

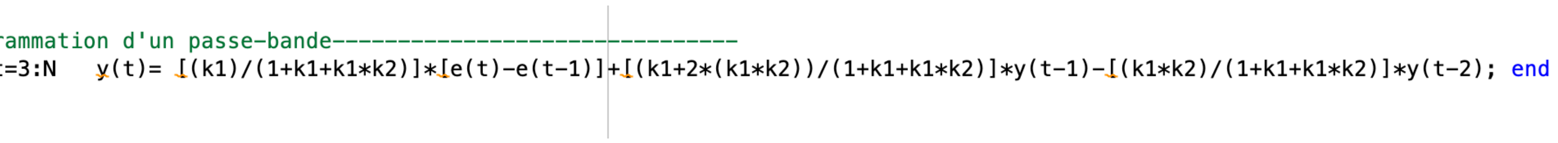


```
%----------------------Programmation d'un passe-haut d'ordre II----------------------
                     for t=2:N   zz(t)=k/(k+1)*[z(t)-z(t-1)+zz(t-1)]; end
```

11. **Graphs of the second order of high pass (k=10) and first order**



12. **Equation of a bandpass in Matlab**



```
rammation d'un passe-bande--------------------------------
=3:N   y(t)= [(k1)/(1+k1+k1*k2)]*[e(t)-e(t-1)]+[(k1+2*(k1*k2))/(1+k1+k1*k2)]*y(t-1)-[(k1*k2)/(1+k1+k1*k2)]*y(t-2); end
```

13. **Scheme of a band pass**

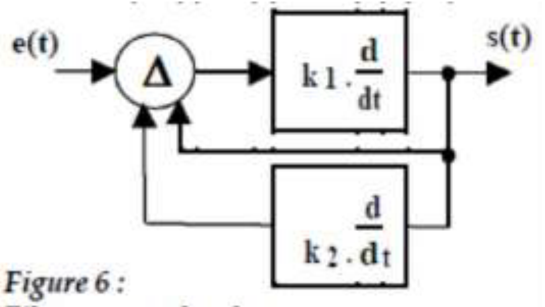
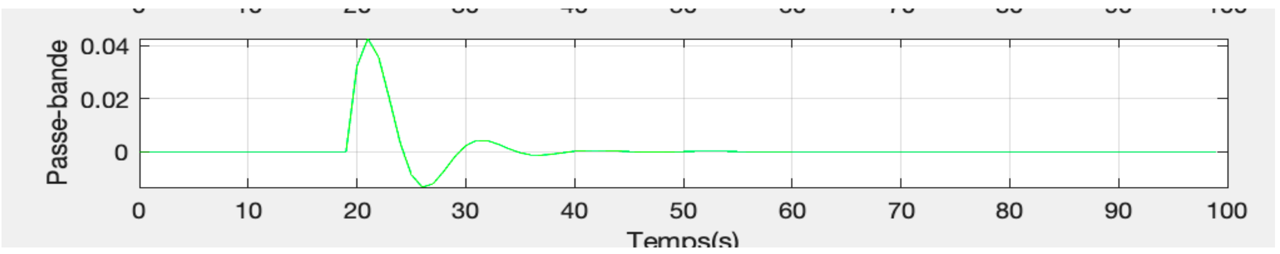
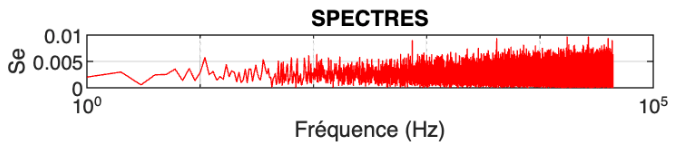


Figure 6 :
Filtrage passe-bande

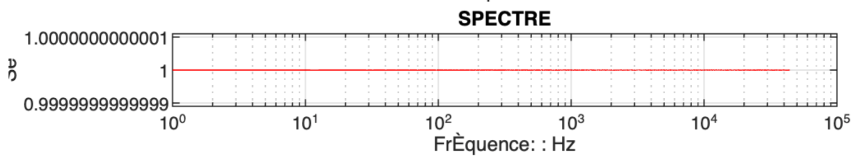
14. **Graph of a band-pass**



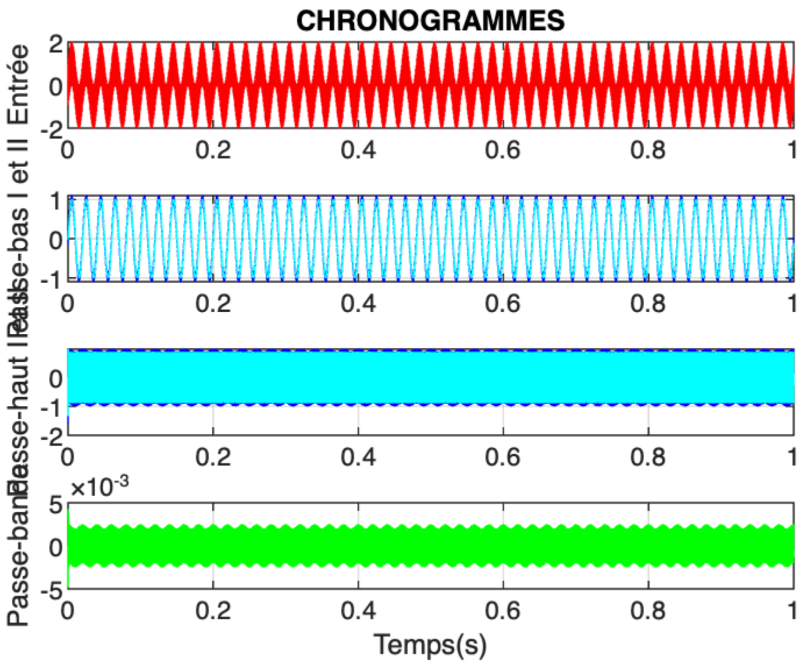
15. **White noise**

White noise

The spectrum is composed of intensified peaks. Its amplitude is groiwing up and its pseudo period is decreasing. When we hear the sound, we heard something like a sizzle which intensify with time.

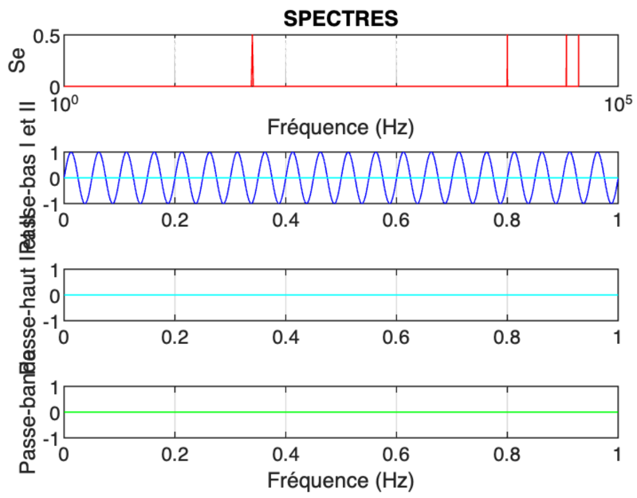
16. **Impulsion**

Impulsion

The difference is that the impulsion is a straight line, constant with a level of 1. When we hear this sound we can heard two bips separated .

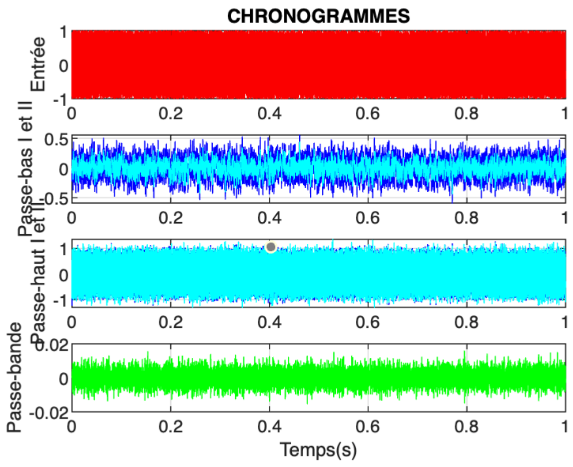
17. Chronograms

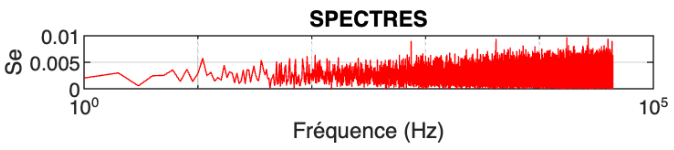


18. Spectrum

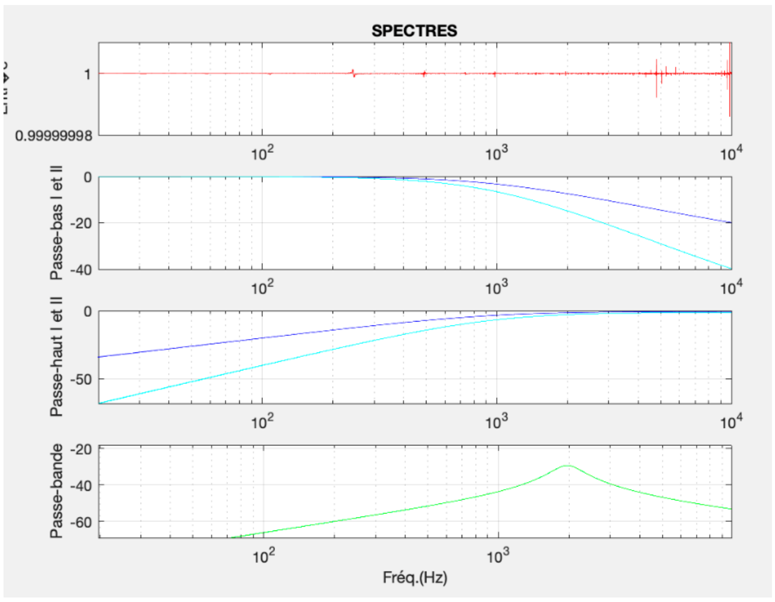


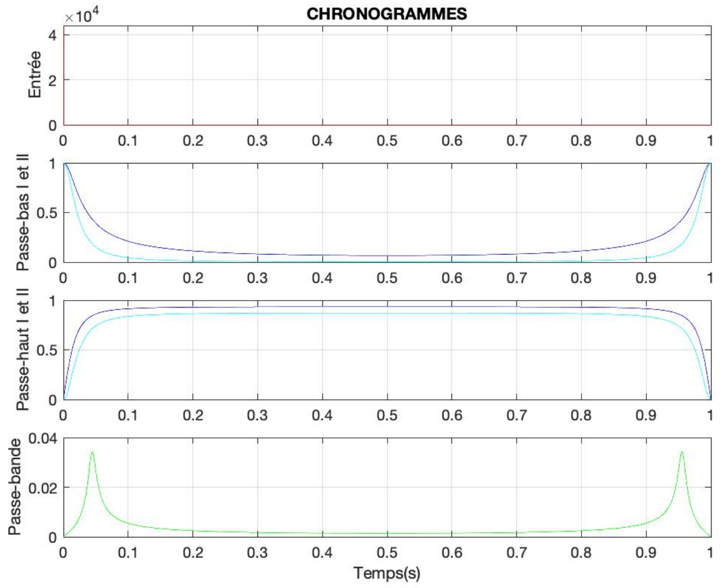
19. Replace the input



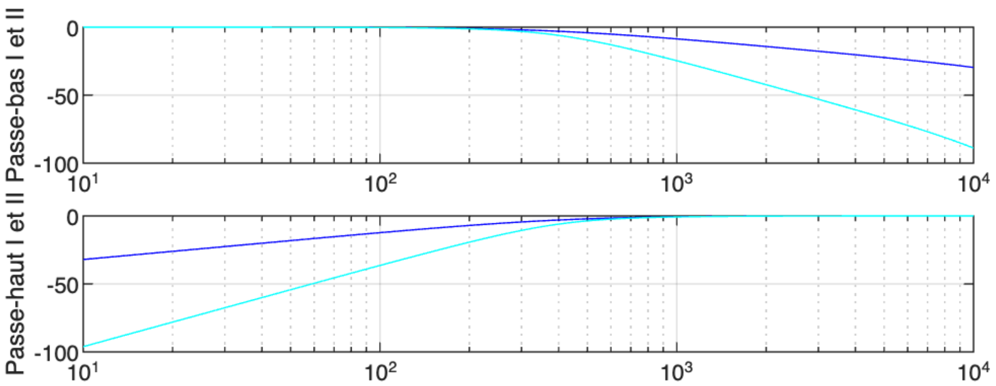


20. Impulse response

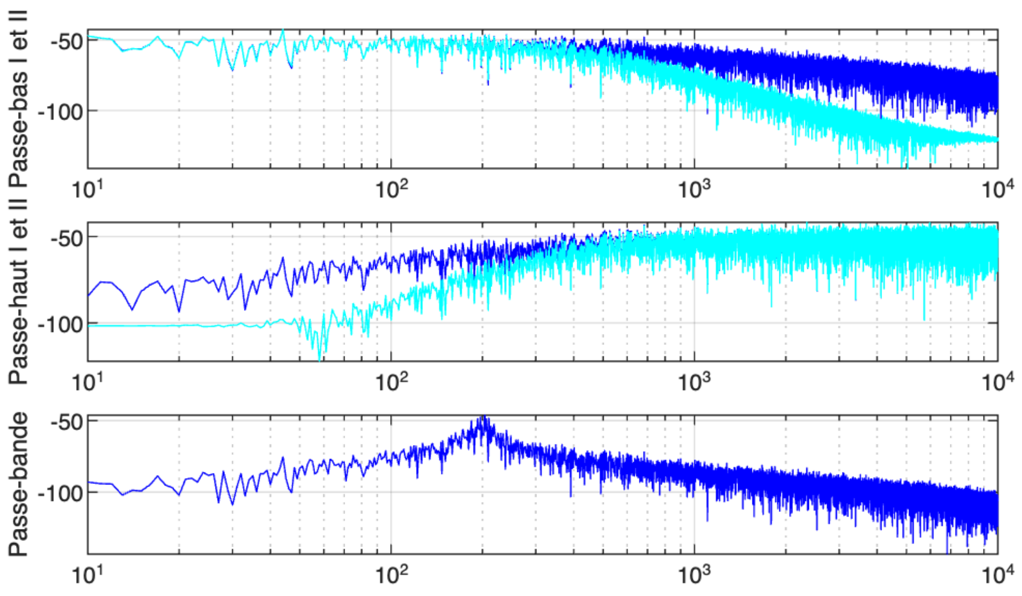




21. Low-pass and High-pass filtering



22. White noise


Passe-bas I et II
Passe-haut I et II
Passe-bande