L1 - year 2019 2020

From Systems to Functions
Session3 Project:
Digital Filtering

PROJECT : DIGITAL FILTERS IMPLEMENTATION AND SYNTHESIS
3rd session : Application of MATLAB filter tools

In the 1st lab session, students were able to program and observe the response of all 1st and 2nd order filter types: low-pass, high-pass and band-pass. It was also established that the time response y(t) of each filter type at the t iteration can be established as a function of its input e(t) using the following recurring equation:

$y(t)+a_1.y(t-1) + a_2.y(t-2) + a_3.y(t-3) +\ldots+ a_{n-1}.y(t-(n-1)) = b_0.e(t)+b_1.e(t-1)+\ldots+ b_{m-1}.y(t-(m-1))$

The two coefficients vectors (arrays) A= [1 a_1 a_2 a_3 … a_{n-1}] and B = [b_0 b_1 b_2 b_3 … b_{n-1}] provide the complete system response. We propose to use the MATLAB tools specifically designed to implement and synthesize digital filters from these two vectors.

MATLAB enables to establish arrays A et B according to the two different methods:

- ***Butterworth: Characterized by the most constant gain within its bandwidth.***
- ***Chebyshev: which tolerates gain variations within the bandwidth but imposes steeper slopes at the cut-off frequencies.***

I. LOW PASS FILTERING

Q1. The next program (lines 5 to 10) shows how to program a 1st order Butterworth low-pass filter with a 400Hz cut-off frequency. (Warning: these methods require a frequency normalization at N/2.)

- Identify the arrays which provide the filter coefficients?
- Run (execute) this program, record the coefficients and establish the recurrence equation between the input and output of this filter.
- **See Lab2**

```
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);
```

Q2. Adding the following lines to the program provides a 2nd order filter.

- Complete and run this program, record the coefficients of this newly designed filter and establish the recurring equation between the input and output of this filter.
- Record/plot the output signal and comment on its asymptotes.
- **See Lab3**
- **How should this program be modified to implement 1st and 2nd order high-pass filters?**
- **Validation1**

```
%-------------------------------Synthèse passe-bas 2
n=…;
[D,C] = butter(n,fn,'low')
ss = filter(…);

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]);
```

I. BANDPASS FILTERING

Q3. Band pass filters (or Band-reject) have two cut-off frequencies.

- Complete the next program taking into account that the cut-off frequencies are given in the LB vector.
- Execute this program, record the filter coefficients and write the recurring equation between input and output.
- Record/plot the output signal and comment on its asymptotes.

```
%-------------------------------Synthèse passe-bande
n=1;
LB=[190 210]; LBn=… ;
[V,U] = butter(n,LBn,'bandpass')
z = filter(…);
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 ;-)
```

Q4. **Validation2** : Bode Plots.

Q5. Substitute the impulse signal by white noise and run the program. Analyze the obtained plots. **Comments ?**