EE 324 LAB 7

Analog filter design

In this lab, you will learn how to design analog filters. You will first design low pass prototypes, and then use transformations to derive low pass, high pass and bandpass filters for given specifications.

Prelab:

1. Given that a Butterworth filter of order M is defined by: $|H(j\omega)|^2 = \frac{1}{1+\left(\frac{\omega}{\omega_c}\right)^{2M}}$ derive the transfer function $H_0(s)$ (in Laplace representation) of a normalized low pass prototype (i.e. $\omega_c = 1$) of order $M = 4$ analytically. Write down and explain all the steps.

2. Use the transformations $s \to \frac{s}{\omega_l}$, $s \to \frac{\omega_h}{s}$, and $s \to \frac{s^2+\omega_0^2}{Bs}$ to derive a low pass filter $H_1(s)$ with cutoff frequency $f_l = 100\ Hz$, a high pass filter $H_2(s)$ with cutoff frequency $f_h = 100\ Hz$, and a bandpass filter $H_3(s)$ with central frequency $f_0 = 100\ Hz$ and bandwidth $B = 40\ Hz$, respectively.

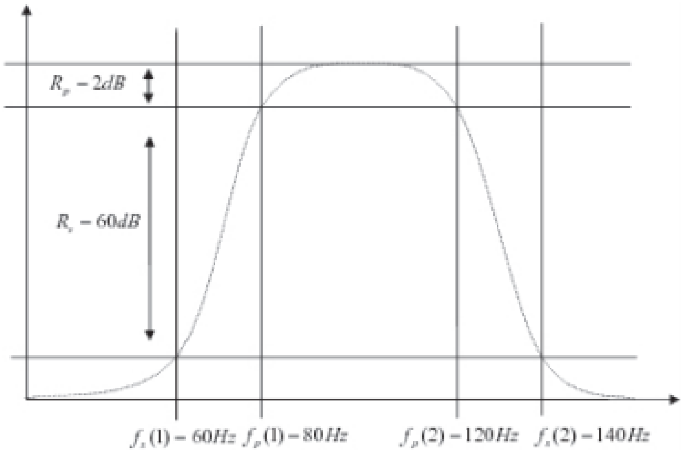

Figure 1 Filter Specifications

Laboratory Assignment:

Problem 1:

1. Verify your Prelab results against those from MATLAB using the function `butter`, by representing the Bode plots for all three filters. Note that in order to design an analog filter you have to use a syntax of the form:
   `[z,p,k]=butter(N,`w_n`,’filtertype’,’s’).`

2. Using MATLAB's function `buttord` (syntax of the form `[N,` w_n`] =buttord(`$w_p, w_s, R_p, R_s,' s'$`))`, to determine the minimum order and the cutoff frequencies, followed by `butter` command, design a Butterworth bandpass filter with the specifications of Figure 1, namely: $f_p = [80\ Hz, 120\ Hz], f_s = [60\ Hz, 140\ Hz], R_p = 2\ dB, R_s = 60\ dB.$

3. Compute the Bode plot for your filter, and verify the specifications are met.

4. Using the MATLAB functions `cheb1ord` and `cheby1`, design a bandpass Chebyshev filter with the specifications in Figure 1.

5. Compute the Bode plot for your filter, and verify the specifications are met.

6. Comment on the differences between the two types of filters.

Problem 2:

1. Write a MATLAB program which records five seconds of your voice (mono), and stores it into a one-dimensional vector *x*. You may use instructions like `y=audiorecorder`, or `record(y,5)`, and `x=getaudiodata(y)`. Use `audioplay` to listen to recorded data and `audiowrite` to save it as a sound file.

2. Look at the frequency characteristics of *x* (you may use the MATLAB function `fft` to compute a fast Fourier transform).

3. In a Simulink model, use the vector *x* as a source input to one of the bandpass filters designed in this lab.

4. Store the output in a vector *y* and play it using the MATLAB function `sound`, and create its sound file using `audiowrite.`