31606 - Assignment number 3

Frederik Ettrup Larsen (s163920), Pelle Michael Schwartz (s147170) and Irene Marie Malmkjaer Danvy (s163905)

I. Summary and objectives

The objectives in this assignent are as follows: (1) design a FIR filter generator with 5 equally wide different band passes completely covering a desired spectrum and use it. (2) Analyze a practical, real world example with a grandmother making and listening to a sweep affected by aliasing, and a grandfather making a running average filter. (3) Design butterworth bandpass filters with specific values. (4) Shorten impulse responses to some different lengths and inspect spectrums for each. Finally (5) design a filter from incomplete specifications. These objectives are all achieved in the following paper.

II. Methods

A. Five knobs to turn

We wish design a filter generator that given a five different gains designs a FIR filter which split the spectrum into five parts and amplifies each with one of the given gains.

To do this we first choose the order of the filter to be large enough to be fairly precise and allowing the number of samples to be divisible by 10. Order 99 was chosen, so that each frequency bucket could contain 10 samples (5 buckets on the positive frequency axis from 0 to the Nyquist frequency and five on the negative frequency axis from $-f_N$ to zero).

Then the frequencies need to be generated. Firstly the the negative frequencies are considered, creating each five buckets each with no amplification, multiplying them with the correct amplification and concatenating them in the correct order. Then, to achieve the symmetry necessary for linear phase, the resulting vector is concatenated with flipped version of itself covering the positive frequencies. The resulting code is shown below.

```
function [filter] = fiveInOne...
(gain1,gain2,gain3,gain4,gain5)
l = 100; %corresponding to order+1
H = horzcat(ones(1,(l/10))*db2mag(gain1),...
ones(1,(l/10))*db2mag(gain2),...
ones(1,(l/10))*db2mag(gain3),...
ones(1,(l/10))*db2mag(gain4),...
ones(1,(l/10))*db2mag(gain5));
H = horzcat(H,fliplr(H));
h = ifft((H),'symmetric');
filter = fftshift(h); %gives the
%impulse response which is also the
%tf, as this is a FIR filter
end
```

B. Nyquist and Aliasing

The Nyquist frequency is calculated as the sampling frequency divided by two. This is the highest frequency that is possible to measure. If the recorded signal gets to a frequency that is higher than the Nyquist frequency *aliasing* will happen and the frequencies will seem to decrease down from the nyquist frequency. Therefore it is important to use an *anti-aliasing filter* or in other ways be sure that no higher frequency gets recorded.

C. Running average filter and filter symmetry

A running average filter has an impulse response that resembles a *rect* function which is then divided by the length of the function. The impulse response of a second order running average filter looks as follows: `h(n) = [1/3 1/3 1/3]`. Assuming we're on the positive time axis, meaning we have a difference equation $y(n) = (x(n)+x(n-1)+x(n-2))(1/3)$, Z-transforming this expression to get the transfer function gives the following: $H(z) = (1 + z^{-1} + z^{-2}) \times \frac{1}{3}$. This is a symmetric transfer function and will thus result in a linear phase as the condition for this is that the impulse response must be symmetric (or anti-symmetric). This will be shown analytically in the results-section.

Based on the transfer function, we can find the zeroes $-0.5000 + 0.8660i = \mathrm{e}^{j2.0944\mathrm{rad}}$ and $-0.5000 - 0.8660i = \mathrm{e}^{-j2.0944\mathrm{rad}}$ and the double poles 0. The unit circle, in the z-plane, maps on to the frequency response. With a normalized response, the top half of the unit circle maps on to the positive normalized axis. Since there are πrad on half a circle, and the gain dips at poles, the frequency response dips at $\frac{2.0944\mathrm{rad}}{pi} = \frac{2}{3}\frac{1}{f_N}$.

D. Use of `buttord` and `butter` to make a butterworth bandpass filter in MATLAB

The `buttord` takes in the *normalized* frequencies of the desired passband frequencies `[max min]` as well as the frequencies of the stopband `[min max]`. This will then return the order and the passband frequency which now can be used to make a butterworth filter using `butter`.

E. Windowing in MATLAB

Windowing in MATLAB was handled by multiplying the impulse with a rect function, with a width carefully chosen to give the right areas of the impulse. As can be seen in the following code

```
% h is the impulse response,
% which is 601 samples long
window50 =...
    [zeros(150,1);ones(301,1);zeros(150,1)];
window25 =...
    [zeros(225,1);ones(151,1);zeros(225,1)];
window10 =...
    [zeros(270,1);ones(61,1);zeros(270,1)];
% window50 should give 50% of the impulse
% the same naming convention applies to
% the other windows
h50 = window50.*h; h25 = window25.*h;
h10 = window10.*h;
```

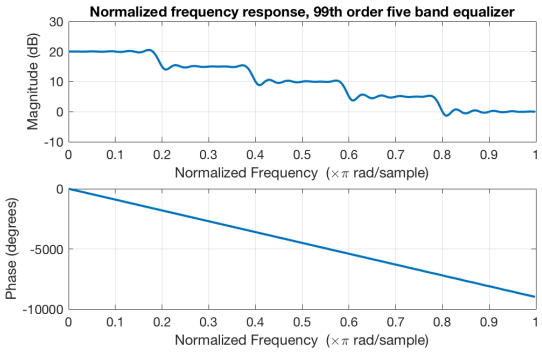


Figure 1: The positive axis frequency response of the FIR filter produced when giving the function described in section II-A the inputs $20, 15, 10, 5$ and 0dB. We see the frequency axis is divided into five equal bands, each amplified with the desired magnitude from lowest to highest. The phase is, as expected, linear.

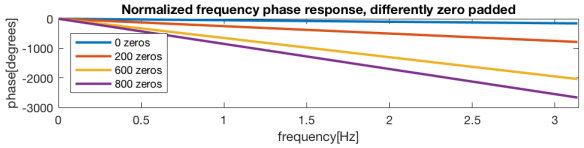


Figure 2: Normalized phase response of the five knob filter whose frequency response is shown in figure 1, with different amounts of zero padding on the left. The increasingly steep phase response as the signal is pushed to right mirrors the time delay zero padding on the right represents. The magnitudes of the frequency responses are not shown, are all identical to the one shown in figure 1.

III. Results

A. Using the filter with five knobs to turn

We give the filter designed in section II-A five gains, each decreasing in steps of 5 dB from the lowest to the highest subbands. The gain $20, 15, 10, 5$ and 0dB where chosen, giving the frequency response shown in figure 1.

When zero padding this filter's impulse response on the left, it has absolutely no effect. This makes sense, as zeroes after an expression doesn't contain information. Zeros after an impulse response has dies out are implied. Physically adding them doesn't make a difference. However when zero padding to the right, the phase of the frequency response increases, as can be seen in figure 2. This is to be expected, as a higher phase indicates time delay, and zero padding to the right is effectively time delay.

When passing randomly generated white noise through the filter we get a frequency response whose magnitude is shown in figure 3.

B. The grandmother's sweep

a) We would expect the sound to rise all the way up to 2500hz, at which point it would start falling again.

b) Our parents, having heard us generate many proper sweeps in MATLAB, would expect at sound going from a deep to a high note.

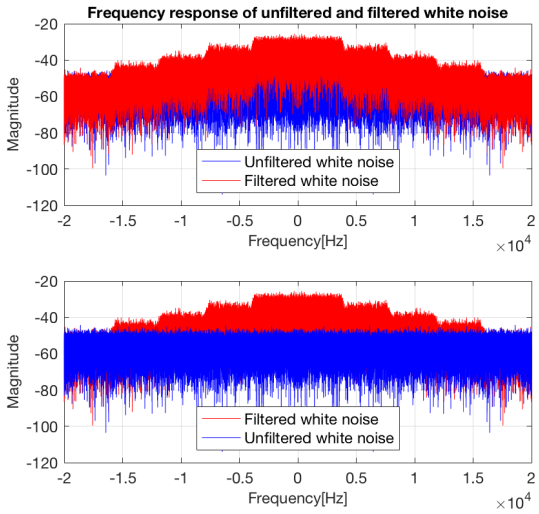


Figure 3: Magnitude frequency response of 10000 sample randomly generated white noise, unfiltered and filtered with our five knob filter In top plot the filtered signal is in front and in the bottom plot the unfiltered signal is in front. As expected the frequency spectrum has been broken up into 5 phases on the positive axis (mirrored on the negative axis), and respectively amplified with 20, 15, 10, 5 and 0 dB. The phase is not shown, as white noise phase has way to much random information to be informative.

c) As the sweep ends at the same frequency it samples at, 5000hz, it would start aliasing as soon as it reached the Nyquist frequency. For this reason, the sound would go up until it reached 2500hz, at which point it would start going down again.

Given that grandmother is probably at least 65, she would have a hard time hearing sounds above 4000hz where she would experience attenuation of around 20dB[2], for this reason we would propose doing a sweep to a maximum frequency of 2500Hz, which she would have a far easier time hearing, and would avoid any aliasing.

C. The grandfather's math

The frequency response can be found analytically from the the transfer function:

$$H(\omega) = H(z)\,|_{e^{j\omega}} \tag{1}$$

$$= (1 + z^{-1} + z^{-2}) \times \frac{1}{3}\,|_{e^{j\omega}} \tag{2}$$

$$= (1 + e^{-j\omega} + e^{-2j\omega}) \times \frac{1}{3} \tag{3}$$

$$= (e^{-j\omega}(1 + 2\cos(\omega)))) \times \frac{1}{3} \tag{4}$$

The magnitude response is then:

$$|H(\omega)| = \sqrt{H_R^2 + H_I^2} \tag{5}$$

$$= \left(\left(\frac{1}{3}\cos(\omega)(1 + 2\cos(\omega)) \right)^2 \right. \tag{6}$$

$$+\left(\frac{1}{3}\sin(\omega)(1+2\cos(\omega)\right)^2\Big)^{\frac{1}{2}} \tag{7}$$

$$=\frac{1}{3}|1+2\cos(\omega)| \tag{8}$$

And the phase response is:

$$\Theta(\omega)=\tan^{-1}\frac{H_I(\omega)}{H_R(\omega)} \tag{9}$$

$$=\tan^{-1}\frac{\sin(\omega)(1+2\cos(\omega))\frac{1}{3}}{\cos(\omega)(1+2\cos(\omega))\frac{1}{3}} \tag{10}$$

$$=\tan^{-1}(\tan(\omega)) \tag{11}$$

$$=\begin{cases}\omega, & 0\leq\omega\leq\frac{\pi}{2}\\ -\omega, & \frac{\pi}{2}\leq\omega<\pi\end{cases} \tag{12}$$

Which is obviously linear, as expected. If our 2nd order filter had been symmetric around 0, meaning the difference equation $y(n)=\frac{1}{3}(x(n+1)+x(n)+x(n-1))$ the phase response would have been[1]:

$$\Theta(\omega)=\begin{cases}0, & 0\leq\omega\leq\frac{\pi}{2}\\ \pi, & \frac{\pi}{2}\leq\omega<\pi\end{cases}$$

With a period of 1s, the sampling frequency is $f_s=1$Hz and the Nyquist ferquency is $f_N=0.5$Hz, meaning the completely suppressed frequency will be $\frac{2}{3}0.5=1/3\approx=0.3333$Hz.

D. 1.1 from Hands-on 8

The Butterworth filter was designed using a passband of `[0.2 0.3]` and with stopbands of `[0.1 0.4]`. The max attenuation of the passband is 2 and the minimum attenuation of the stopband is 100. This resulted in the filter with a frequency response and phase shown in figure 4. Shortening it's impulse response gives the results seen in figures 5 and 6.

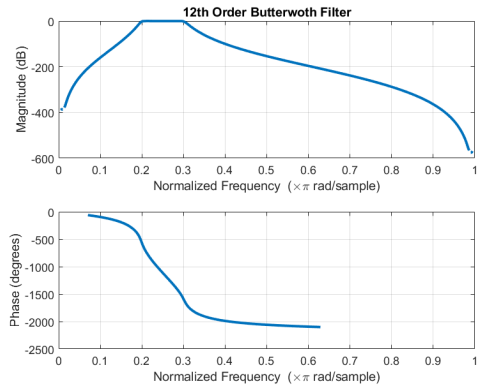


Figure 4: Butterworth filter designed with a passband of `[0.2 0.3]` and with stopbands of `[0.1 0.4]` the minimum attenuation requirement of 2 dB through the passband is met and the attenuation goes below 100 at both the lower and upper stopband end- and starting frequency. The phase goes really crazy but that is expected for a such high-order filter.

The frequency responses on figure 6 shows that you don't need the complete end of the impulse response to get the

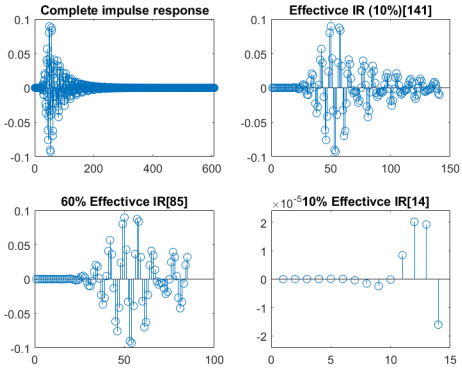
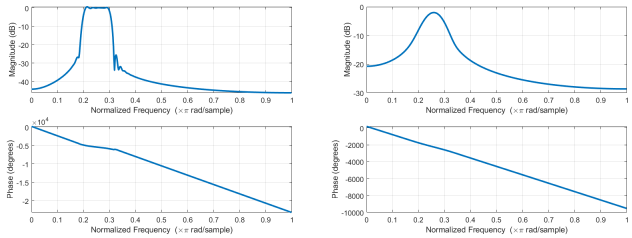


Figure 5: The impulse response generated from the normalized transfer function using `impz(B,A)` is shown in the top left in full and then cut off at its effective length(where it cuts down below 10% of max value). Then there is a plot 60% of the effective length and at 10% of the effective length. The brackets contain the length of each signal in samples.



(a) 100 % effective length (b) 40 % effective length

Figure 6: Frequency responses and phase responses made from the cut impulse response of the butterworth filter displayed a on figure 4. 6a is cut at it's effective length(where the magnitude of the peaks go below 10% of the maximum value). 6b is then 40% of the effective length of the impulse response. Only two of the 4 impulse responses are plotted to save space. Also notice the increasingly linear tendency of the phase.

filter's frequency response (with minor ripples and lesser stop band attenuation). However if you cut off too much of the frequency response's length you will end up getting an attenuation across the pass band frequencies as well. When going all the way to 10% of the effective length the minimum attenuation across the pass band is way below 80 dB, essentially making the filter useless.

E. 1.2 from Hands-on 8

Even though we don't know a lot about this filter, we do know enough to create it. We have the following information:

- It is an ideal lowpass FIR filter
- It passes frequencies from 0 to 5kHz (with a gain of 0 db)
- Cut off at $\frac{1}{3}$ on a normalized frequency axis
- The impulse response is 601 samples long

1) Finding the sampling frequency: Given that this is a lowpass filter, the cut off frequency on the normalized axis

must be the same as the last frequency that is passed. So we can say, that $\frac{1}{3}$ is equivalent to 5kHz.

The normalized frequency axis is computed by dividing all of the frequencies between 0 and the nyquist frequency with the nyquist frequency, this means, that the sampling frequency can be computed as:

$$\frac{1}{3} = \frac{5\text{kHz}}{f_n} = \frac{10\text{kHz}}{f_s} \implies f_s = 30\text{kHz}$$

2) Designing the filter: Given that our colleague has handed us such vague specifications, we have decided to just make an ideal lowpass filter, given that there is 601 samples, and it passes one third of them, the filter should be 200 ones and 401 zeros. Because we want this filter to be symmetric, and to handle the wackyness of the `fft` function, we create the filter in the following manner:

```
H = [ones(100,1);zeros(401,1);ones(100,1)];
```

The frequency response can be seen in figure 7.

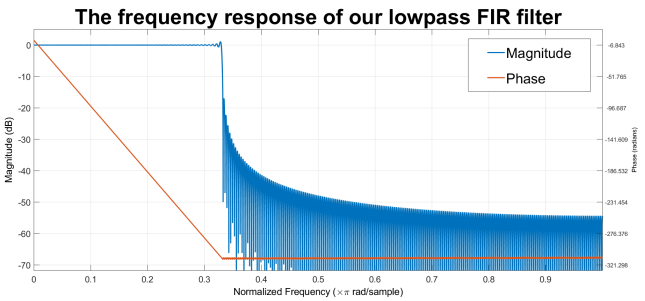


Figure 7: The phase and the magnitude plottet on top of each other, the magnitude goes from 0dB to around -70dB, and the phase goes from 0 to around 100π. Note that it cuts off at around 0.3333 or one third

Note that we make the sure that the impulse response is real, by calling it as 'symmetric':

```
h = fftshift(ifft(H,'symmetric'));
```

The response can then be seen in figure 8.

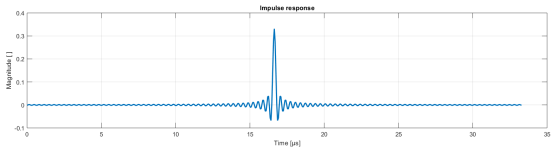


Figure 8: The impulse response is a sinc function, which is to be expected, as it a rect function in the frequency domain

We then reduce the length of the filter by windowing the impulse response. This results in the frequency responses that can be seen in figure 9, and the impulse responses that can be seen in figure 10.

It is clear that the order of the filter increases as we start windowing it more and more drastically. This makes sense, as the zeros or B of an FIR filter is the impulse response, so shortening it would reduce the order. Perhaps it would be best to send a message to our colleague asking which order which order is desired, and while we're at it, suggest that our colleague tries to keep their coffee from the specifications.

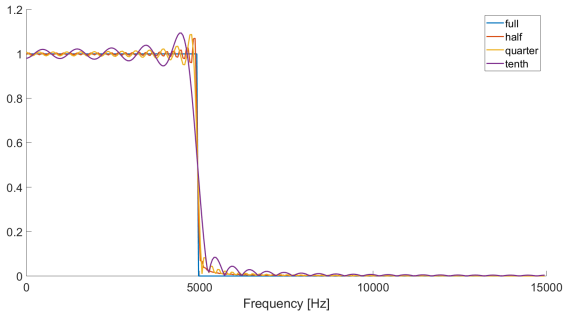


Figure 9: As we can see, the frequency response gets more and more ripples, this is what we would expect, as windowing is essentially multiplying a function with a rect function. This is equivalent to a convolution with a sinc function in the frequency domain, which results in ripples.

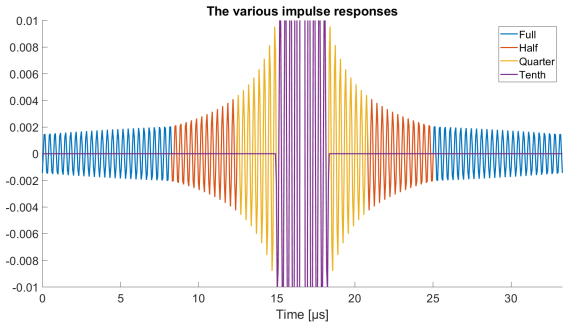


Figure 10: The impulse responses as more and more is cut off. Here it is clear to that it is multiplied by a rect function

IV. Discussion

The importance of making the right choice for sampling frequency is shown in the example with the grandma who makes the sine sweep. This shows that if not taken account for, a recording without anti-aliasing and/or a sampling frequency that is lower than half the maximum possible frequency (Nyquist frequency) will yield a result that does not affect the actual input. The "five band pass" FIR filter in 1.1 has been successfully made. It is shown how a running average filter has a linear phase. Lastly the Butterworth band-pass filter was designed, and it was shown how the shortening of the impulseresponse resulted in an increase of attenuation through the pass-bands and a less steep slope towards stop-bands this shows that while it is possible to shorten the impulse response, it will result in a decline of effectiveness. The "coffe stain" problem showed that even with few specifications a satisfying filter can be produced, eventhough, it is necessary to keep track of how long it should be.

References

[1] J. G. Proakis and D. G. Manolakis. *Example 5.1.2 from Digital Signal Proecessing - Principles, Algorithms and Applications, Fourth Edition*. Pearson Education, Inc., 2007. ISBN 0-13-187374-1.

[2] R. Russell. Listening and Hearing. http://www.roger-russell.com/hearing/hearing.htm. Accessed: 2018-11-14.