31606 - Assignment number 2

101010, Frederik Ettrup Larsen (s163920), Pelle Michael Schwartz (s147170) and Irene Marie Malmkjaer Danvy (s163905)

I. Summary and objectives

The first part of this assignment focuses on designing a digital lowpass filter using bilinear transform with prewarping. The filter is used to process a sinusoidal signal such that the resulting phase is zero, and then used to produce another filter with a higher order, at the allowed cost of a change in phase and regularity of the filter.

The second part of this assignment focuses on designing filters imitating a room, meaning in which sound bounces of walls with a delay using FIR and IIR filters.

II. Methods

A. Digital Filter Design in Matlab

To design a butterworth filter in matlab the `buttord` and `butter` functions has been used. In order to do so, it is necessary to chose a suitable sampling frequency. We let it to be twice the Nyquist frequency found from the highest frequency to be processed $\omega_h = 35\frac{\text{rad}}{\text{s}}$. The sampling frequency is then $f_s = 2 \cdot f_N = 2 \cdot 35\frac{\text{rad}}{\text{s}} \cdot \frac{1}{2\pi} = 11.14$Hz, corresponding to a sampling angular frequency of $\omega_{sampl} = 70\frac{\text{rad}}{\text{s}}$ used in the code for simplicity's sake. Then we declare the passband and stopband attenuation, the passband and stopband frequencies and then normalize them with the sampling frequency. The resulting matlab code (shown below) can then be executed.

```
O_sampl = 2*35; %sampling angular
% frequency
Gp = 2;  % passband maximum attenuation
Gs = 11; % Stopband minimum attenuation
Op = 10; % omega max passband frequency
% [rad/sec]
Os = 15; % omega min stopband frequency
% [rad/sec]

wp = Op/O_sampl;%normalize
ws = Os/O_sampl;%normalize

[n,Wn]=buttord(wp,ws,Gp,Gs);
%pre-warping and biletaral transform is
%included in the funciton
[B,A] = butter(n,Wn);
```

Where B and A represent respectively the parameters of the input and the output, as well as the numerator and denominator of the transfer function. Indeed the transfer function can be found using the matlab command `tf(B,A)`.

B. Messing around with the same filter

A filter with zero phase shift is desired, and the only tool to construct it is the low pass Butterworth filter designed according to the principles explained above. To fulfill this desire we need only pass the signal through our filter twice, first the intended way and then the opposite way.[1] This last step can be simulated by the inverse transfer function. Thus, designating our filter using `A` and `B`, the total filter our system passes through has the following shape (in matlab notation) `tf(B,A)*tf(A,B)`, which will always produce an all pass filter. This of course also means that the there is no magnitude change, but that is acceptable since we're prioritizing the phase.

In a similar situation a higher order filter than the one possessed is wished for. Then the signal must simply be passed through 2 or more of the possessed filters. Since filters in series produce a larger filter with a transfer function that is the product of the transfer functions in the chain, the total transfer function of the system achieved by chaining two identical filters will have a transfer function of an order corresponding to the sub-filters' order in the second.

C. Methods for finding the impulse responses of a FIR and IIR filter

Our FIR filter's expression is $y(n) = x(n) + \alpha x(n - delay)$, making the impulse response correspond to the factors multiplied by the x-elements, namely [1 (a number of zeros equal to $f_s \times$delay) α].

Our IIR filter's expression is $y(n) + \alpha y(n - delay) = x(n)$. The impulse response is, as it says in the name, infinite. However according to [1], the zero state response of an IIR system with $x(n) = \delta(n)$ is the impulse response.

The transfer function for the FIR filter is found by looking at the system equations and Z-transforming:

$$\begin{aligned} y(n) =& - a_1 y(n-1) + ... - a_L y(n-L) \\ & + b_0 x(n) + b_1 x(n-1) ... + b_M x(n-M) \\ y(n) =& x(n) + \alpha x(n - delay) \\ H_{FIR}(z) =& \frac{Y(z)}{X(z)} = \alpha z^{-delay \cdot f_s} + 1 \end{aligned}$$

The transfer function for the IIR filter is found similarly to the FIR filter, here however, it is realized, that the equation for the IIR filter is almost the same as the one for the FIR filter, with the y's and x's replaced. For this reason, the transfer function must be an inversion of H_{FIR}:

$$H_{IIR}(z) = H_{FIR}^{-1}(z) = \frac{1}{\alpha z^{-delay \cdot f_s} + 1}$$

[1] This can be done in the opposite order, but the result would be the same since the transfer function of two filters in series, is simply the product of the two filters' transfer functions and multiplication is commutative.

To verify this, we will try to find the transfer function of the IIR filter mathematically:

First we must find an explicit expression for the system output.

To simplify the process we consider the specific case where $delay = 3f_s$, so $y(n) = x(n) - \alpha y(n-3)$. This is done below.

$$
\begin{aligned}
y(0) &= x(0) - \alpha y(-3) \\
y(1) &= x(1) - \alpha y(-2) \\
y(2) &= x(2) - \alpha y(-1) \\
y(3) &= x(3) - \alpha y(0) = x(3) - \alpha x(0) + \alpha^2 y(-3) \\
&\vdots \\
y(n) &= x(n) - \alpha x(n-3) + \alpha^2 x(n-6) - ... \\
&\quad + (-\alpha)^{\lfloor n/3 \rfloor} x(n - 3\lfloor n/3 \rfloor) + (-\alpha)^{\lceil n/3 \rceil} y(3n\lceil n/3 \rceil) \\
&= \left(\sum_{k=0}^{\lfloor n/3 \rfloor} (-\alpha)^k x(n-3k) \right) + (-\alpha)^{\lceil n/3 \rceil} y(n - 3\lceil n/3 \rceil)
\end{aligned}
$$

If we let $d = delay \cdot f_s$ this can be generalized to $\left(\sum_{k=0}^{\lfloor n/d \rfloor} (-\alpha)^k x(n - d \cdot k)\right) + (-\alpha)^{\lceil n/d \rceil} y(n - d\lceil n/d \rceil)$. The zero-state response is then found by letting all values of $y(n), n < 0$ be 0. Since:

$$n - d\lceil n/d \rceil < 0 \ \forall \ n \leq 0 \in \mathbb{N}$$

The zero state response is:

$$y_{zs}(n) = \sum_{k=0}^{\lfloor n/d \rfloor} (-\alpha)^k x(n - dk), n \leq 0$$

The impulse response (assuming the system starts at rest) is then:

$$h(n) = \sum_{k=0}^{\lfloor n/d \rfloor} (-\alpha)^k \delta(n - dk)$$

This sum will look like a a series of pulses with interval d all a factor $-\alpha$ smaller (since $\alpha = 0.6 < 1$) than the previous one, as expected.

Z-transforming this, and using some well known sums, we get:

$$
\begin{aligned}
H(z) &= \sum_{n=0}^{\infty} h(n) \cdot z^{-n} \\
&= [1 \cdot z^0 \quad ... \quad - \alpha \cdot z^{-d} \quad ... \quad \alpha^2 \cdot z^{-2d} ...] \\
&= \sum_{n=0}^{\infty} (-\alpha)^n \cdot z^{-n \cdot d} \\
&= \frac{1}{1 + \alpha z^{-d}}
\end{aligned}
$$

As expected.

Making this in matlab, is just a matter of writing the following:

```
FIR = filter(A,B,s);
IIR = filter(B,A,s);
```

Where s is an impulse (vector with 1 in the first place followed by zeros), and `A` and `B` are the coefficients of the numerator and the denominator of the filters. For FIR, `A` is a 1 followed by a number of zeros equal to the delay multiplied with the sampling frequency minus 1 and then an α, and the `B` is just a 1

For IIR, we have just replaced `A` and `B` position in the function, which inverts the transfer function of the filter.

III. Results

A. Digital Filter Design in Matlab

The digital lowpass filter using bilinear transform has been made with the following values:

- $\omega_{sampl} = 2 \cdot 35 = 70 rad/s$
- $G_p = 2$
- $G_s = 11$
- $\omega_p = \frac{10}{\omega_{sampl}} \approx 0.143$
- $\omega_s = \frac{15}{\omega_{sampl}} \approx 0.214$

After using `buttord` to find a order of $n = 4$ the z-plane transfer function is found using `butter` resulting in the spectrum of the filter seen in figure 1.

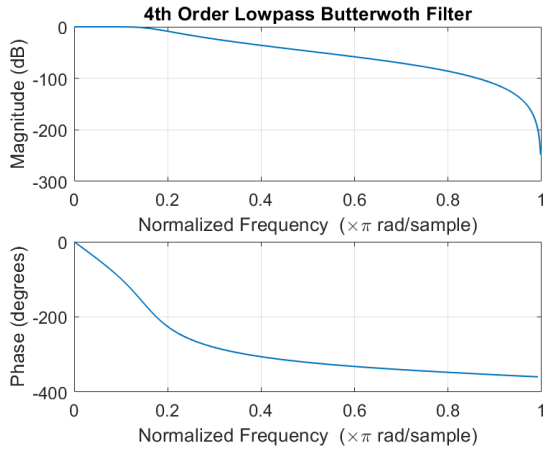

Figure 1. Normalized phase and magnitude responce of the Butterworth filter designed with a passband of $0 \leq \omega \leq 10 \frac{\text{rad}}{\text{s}}$ and a stopband of $\omega \geq 15$. The frequency along the x-axis is normalized with the sampling frequency $\omega_s = 70 \frac{\text{rad}}{\text{s}}$. We can see that over the pass band $0 \leq \omega \leq 10 \frac{\text{rad}}{\text{s}}$ (corresponding to $0 \leq \omega_{normalized} \leq 0.1429$ on the plot) there is a gain of no more than -2 dB. Likewise over the stopband $\omega \geq 15 \frac{\text{rad}}{\text{s}}$ (corresponding to $\omega_{normalized} \geq 0.2143$ on the plot) there is a gain of no less than -11 dB. Thus it is clear that the design has lives up to the criteria described in assignment and in the methods section of this report.

And the transfer function of the filter is:

$$H(z) = \frac{z^4 - 2.689z^3 + 2.86z^2 - 1.399z + 0.2633}{0.00226z^4 + 0.009039z^3 + 0.01356z^2 + 0.009039z + 0.00226}$$

B. Messing around with the same filter

For the new filter with zero phase shift we extend our existing filter with an inverted version of the same filter, as described in the methods section. This filter's transfer function will look as follows:

$$H_{NoPhaseShift}(z) = H(z) \cdot H(z)^{-1} = 1$$

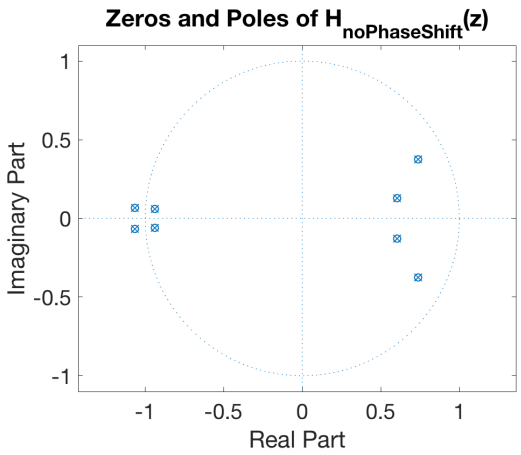

Figure 2. The zeros and poles of the first modified filter, designed not to change the phase. Clearly all the poles and zeros cancel each other out, making the filter all pass.

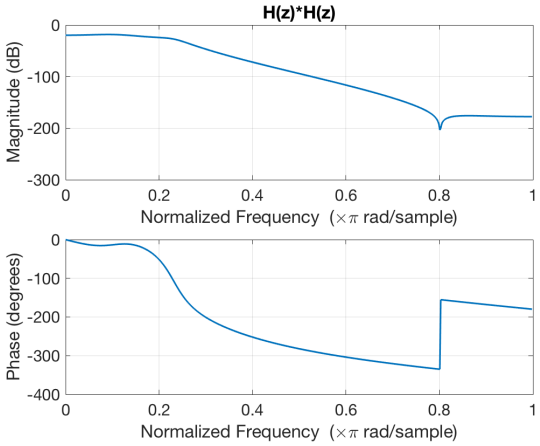

Figure 3. Phase and magnitude response of the 8th order filter resulting from daisy chaining 2 of our 4th order low pass Butterworth filter. If we compare this response with the one in figure 1 we see a steeper slope, both in the magnitude and phase, indicative of this filter's higher order. We also note that this filter is much less regular than the original Butterworth filter both in phase and magnitude. It is also no longer a pure lowpass filter, as can be seen from the upward going magnitude after 0.8 rad per sample. However these were both qualities that we were allowed to sacrifice on the alter of obtaining a higher order filter.

Which, as expected, is an all pass filter. This makes sense, as by taking the product of a transfer function and it's inverse all poles will have a corresponding zero at the same place and vice versa, as can be seen in figure 2.

For the new filter with a higher order we daisy chain two of our original filters, as described in the methods section. This gives the following transfer function:

$$
\begin{aligned}
H_{highOrder}(z) &= H(z) \cdot H(z) \\
&= \frac{\begin{aligned} &5.106 \cdot 10^{-6} z^8 + 4.085 \cdot 10^{-5} z^7 + 0.000143 z^6 \\ &+ 0.000286 z^5 + 0.0003574 z^4 + 0.000286 z^3 \\ &+ 0.000143 z^2 + 4.085 \cdot 10^{-5} z + 5.106 \cdot 10^{-6} \end{aligned}}{\begin{aligned} &z^8 - 5.377 z^7 + 12.95 z^6 - 18.17 z^5 + 16.23 z^4 \\ &- 9.415 z^3 + 3.462 z^2 - 0.7365 z + 0.06933 \end{aligned}}
\end{aligned}
$$

We can clearly see that we have produced a total filter of order eight, twice the order that of the original filter. The normalized frequency and magnitude response of our 8th order filter is shown in figure 3.

C. The FIR and IIR filter

The transfer function response of the filter with the difference equation $y(n) = x(n) + \alpha x(n - delay)$ is, as explained in section II-C, simply [1 (a number of zeros equal to $f_s \times$delay) α].

The transfer function response of the filter expressed with the difference equation $y(n) + \alpha y(n - delay) = x(n) \Leftrightarrow y(n) = x(n) - \alpha y(n - delay)$ can be found as explained in section II-C. Simply, the inverse of the FIR filter.

Doing so grants us the impulse responses seen in figures 4 and 5.

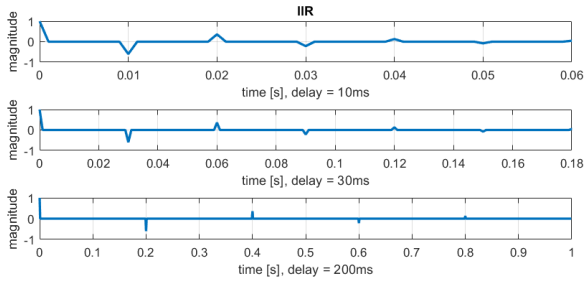

Figure 4. Impulse responses of the IIR filter with different delays, we decided to cut off the plot when the peaks were so small that they almost couldn't be seen. Note that changing the delay only changes the distance between each peak, since the distance between peaks corresponds exactly to the delay.

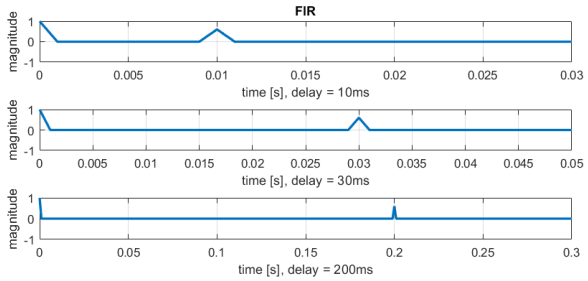

Figure 5. Impulse responses of the FIR filter with different delays, here it can be seen that the second peak appears after the delay has passed, attenuated by α, corresponding to the single echo in the FIR model of the room.

The IIR filter is essentially a comb-filter, we would then expect the spectrum of the filter to represent this, i.e being some form of comb with the distance between the peaks being inversely proportional to the delay. Due to the inverse relation between the two filters, we would except the same to be the case for the FIR filter, only with an upside down spectrum

compared to the IIR. These spectrums can be seen in figures 6 and 7.

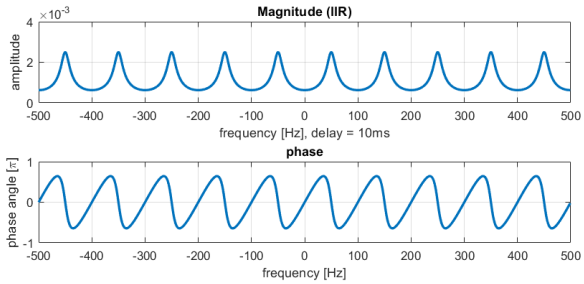

Figure 6. The magnitude and phase plot for a delay of 10 ms for the IIR filter, note that the sampling frequency is 1000Hz, and that the period is 100Hz, which is equal to $(2/t_{delay}) \cdot f_s$, this holds for the other impulse responses as well.

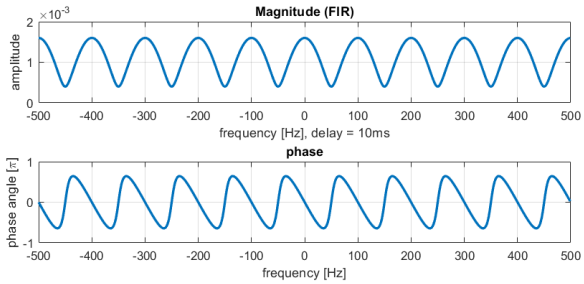

Figure 7. The magnitude and phase plot for the FIR filter, with a delay of 10ms, note that the phase response is the inverse of the IIR filter's phase in figure 6, and the magnitude is flipped and slightly differently scaled compared to the IIR filter's magnitude.

Listening to the filters: We also applied the filters on audio files. If applied to music, it was hard to tell any difference on any of the 6 produced filters. The 200ms a slight, but undefinable, distortion was audible. However, applying the FIR and IRF 200ms filters on audio with speech, the distortion was now distinguishable from the original audio and the speech now sounded like it was heard from the other end of a short pipe like a cardboard tube. Using it one a sine, all of the filters now really distrorts the audio in weird ways. This is because, as is seen on figure 6 and 7, the amplification of the diffrent frequencies are shaped as "waves" so every different frequency of the sweep is getting attenuated more or less, resulting in a wavy echo. It is possible to note the diffrence in sharp peaks on the IIR filters versus a softer peak on the FIR.

IV. Discussion

Digital filter design was done using matlab functions, avoiding manual transformations and manual pre-warping. The filters succesfully lived up to the demands of attenuation at max passband and min stopband frequencies. The changes in in phase shift to make an all-pass filter by running the filtered signal through an inverted filter was also succesful.The 8th order filter made by passing the filtered signal through the filter again resulted in achiving a filter with the given demands of higher order but not neccesarily as regular as the 4th order Butterworth filter. It also had a steeper phase which also was allowed.

A room was implemented through a FIR and IIR filter. Where the FIR was represented with a single attenuation of α after a certain delay, while the IIR filter was represented with an infinite amount of pulses attentuated by $-\alpha$. Tests showed that the 10ms and 30ms filters were hard to differ on recorded audio but the 200ms filters resulted in audible diffrences.

References

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