Politecnico di Torino

Applied Signal Processing Laboratory

Assignment 2: LTI Systems and Digital Filtering

Francesco Laterza, 271087 Giorgio Agnello, 272742

07/05/2023

Contents

1 LTI systems and digital filters

1.1 Exercise 1

For this exercise we are given the numerator and denominator coefficients of three digital filters.

1.1.1 Filter constellation

Using the Matlab function *roots* we can get the zeros and poles of the filter transfer function. We plot them using *zplane*.

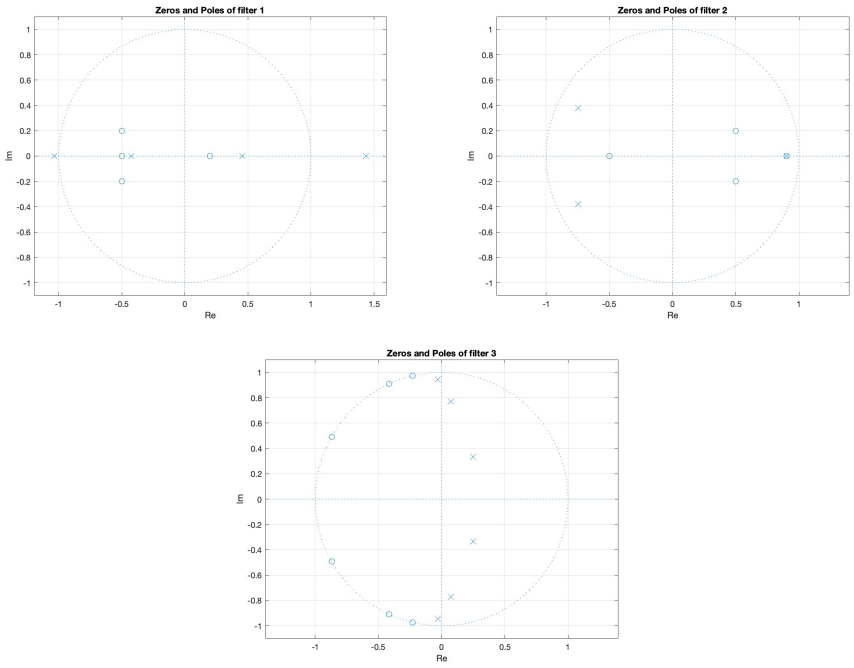

Figure 1: Filters constellation

In Figure 1 we can see the constellation of the three filters. The plots show the complex plane, highlighting the unitary circle. Zeros are represented by crosses and poles by circles.

To check if a filter is stable, it is enough to check if all poles are included in the unitary circle. Thus, filter 1 it is not stable.

1.1.2 Impulse response

We now compute the impulse response of each signal by applying each filter to a Kronecker delta function $\delta[n]$.

In Figure 2 the impulse response of the three filters are plotted. As expected the first filter impulse response diverges to $+\infty$, while the others converge to 0.

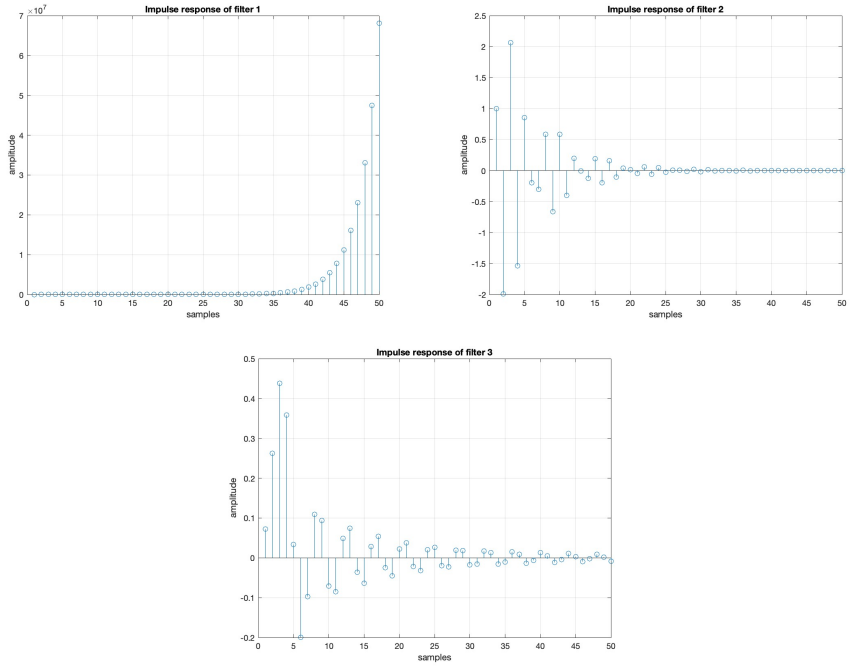

Figure 2: Filters impulse response

1.2 Exercise 2

In this exercise we design a custom function *my_filter*. This function accepts three input parameters: the two filter coefficients vectors and a signal vector.

1.2.1 Implementation

The function compute the resulting filtered signal $y(n)$ as

$$y(n) = b(1)x(n) + b(2)x(n-1) + \cdots + b(N_b)x(n-N_b+1) + \\ - a(2)y(n-1) - \cdots - a(N_a)y(n-N_a+1)$$

where N_a and N_b are respectively the length of the two coefficient vectors.

1.2.2 Impulse Response

Let's generate the filter coefficients with $[b, a] = \text{butter}(8, 0.1)$.

In Figure 3 we compare the impulse response computed by use of our custom function and by use of *impz*. The two impulse responses match.

1.2.3 Frequency Response

We can compare the frequency response first by computing the DFT of the coefficient vectors and then performing the ratio, and second by the use of *freqz*.

In Figure 4 we can see the result of the comparison of the frequency response.

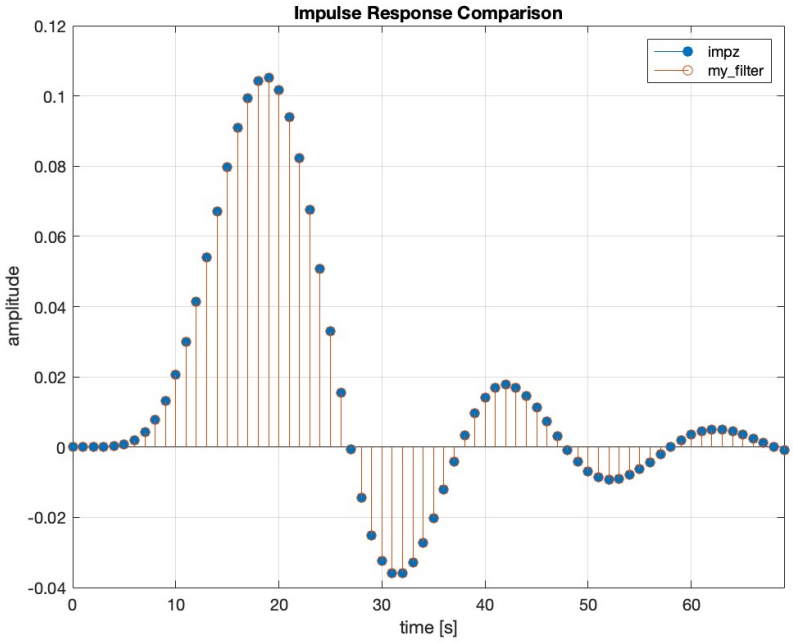

Figure 3: Filter impulse response comparison

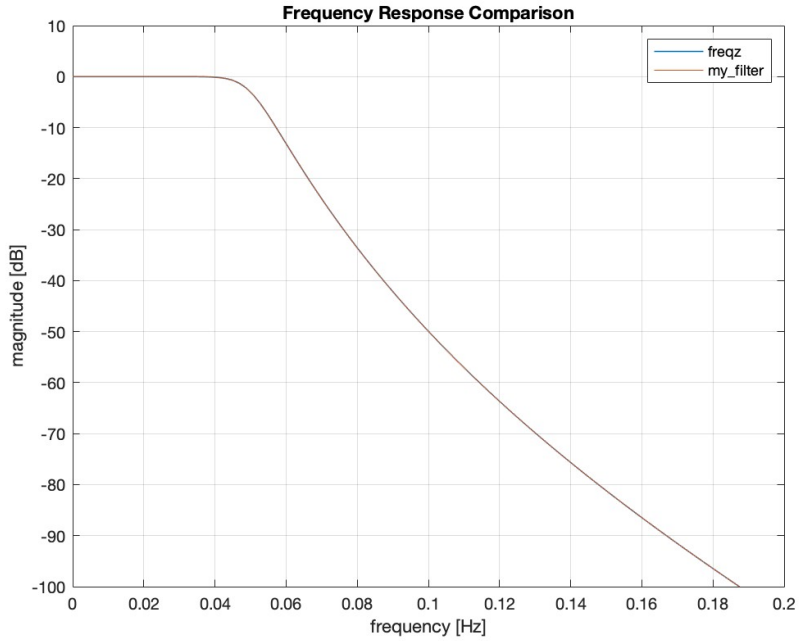

Figure 4: Filter frequency response comparison

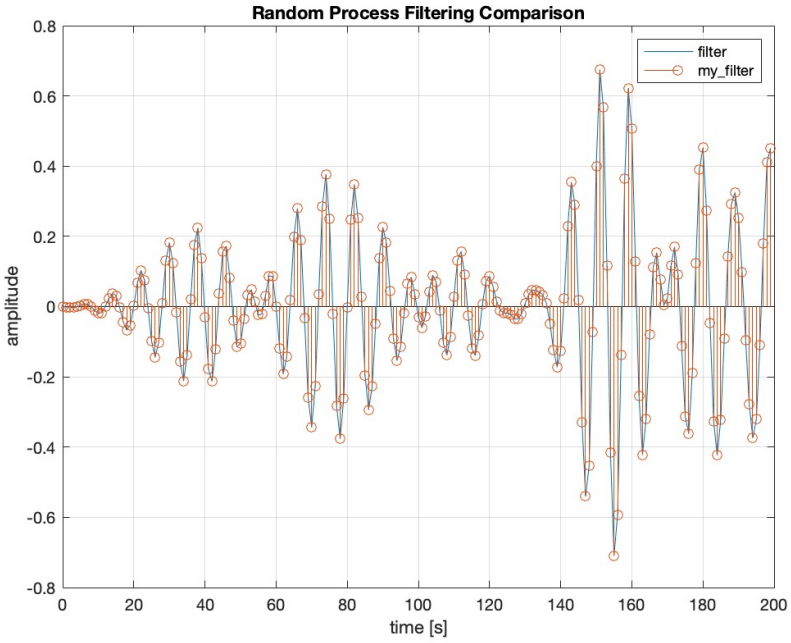

Figure 5: Filtered signal comparison

1.2.4 Filtering of Random Signal

We generate a random signal and filter it with both *filter* and *my_filter*.

In Figure 5 the comparison between the *my_filter* and *filter* function. As expected the two result match.

1.3 Exercise 3

Let's consider the following signal

$$x(t) = \cos(2\pi f_1 t) + \cos(2\pi f_2 t) + w(t)$$

where $f_1 = 5$Hz, $f_2 = 100$Hz and $w(t)$ is a zero-mean WGN process with variance 10.

1.3.1 PSD Estimation

We can estimate the power spectral density of our signal with the use of *pwelch* Matlab function

In Figure 6 it is plotted the estimated PSD. The four deltas are clearly visible at frequency ± 5Hz and ± 100Hz as expected.

1.3.2 Butterworth band-pass filter

We design a Butterworth filter in order to only let the 100Hz frequency of our signal pass.

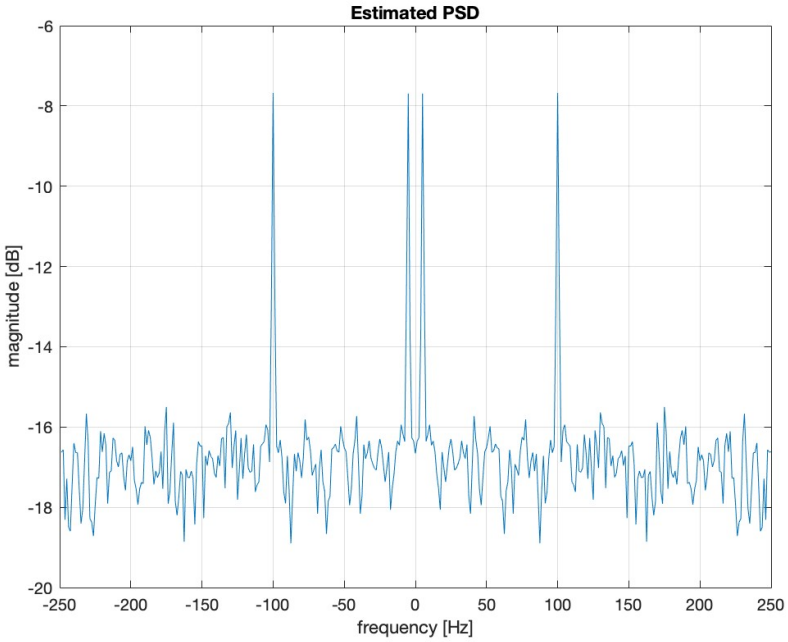

Figure 6: Power spectral density estimation

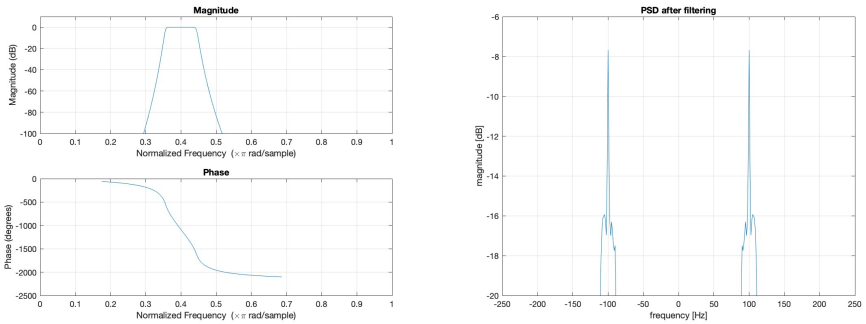

Figure 7: Butterworth filter frequency response and PSD after filtering

In Figure 7 we can see the frequency response of our Butterworth filter and the PSD of the signal after filtering. As expected only the ±100Hz frequency are present in the new PSD.

1.3.3 Elliptic band-pass filter

We repete the same process of the last section (1.3.2), this time we use an Elliptic filter.

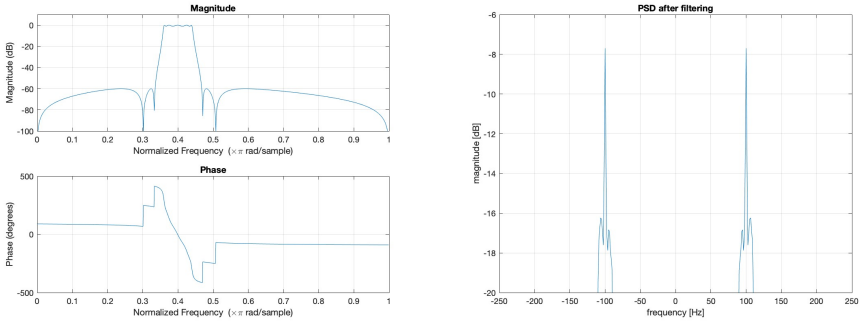

Figure 8: Elliptic filter frequency response and PSD after filtering

In Figure 8 we can observe a very similar result in the filtering to the previous one.

1.3.4 Equiripple FIR band-pass filter

As before, we repeat the process this time with an Equiripple FIR filter.

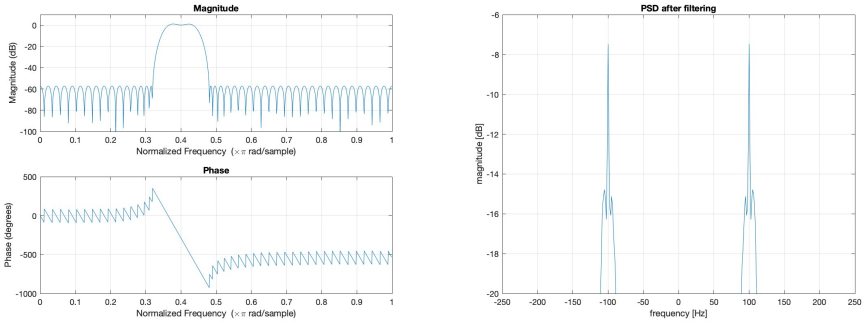

Figure 9: Equiripple FIR filter frequency response and PSD after filtering

In Figure 9 we can observe the result of the Equiripple FIR filtering. Also this time we get as expected only the 100Hz frequency in the PSD.

1.3.5 Filter comparison

From the different type of filtering, we can observe that the Elliptic filter is the most selective having the steepest transition band.

The filter with the most pass-band ripple is the Equiripple filter and the one with the most stop-band attenuation is the Butterworth filter.

Lastly the filter which requires the lowest order is the Elliptic filter being order 5.

1.3.6 Chebyshev type 1 low-pass filter

This time we design a Chebyshev type 1 low-pass filter to get only the 5Hz frequency of our signal.

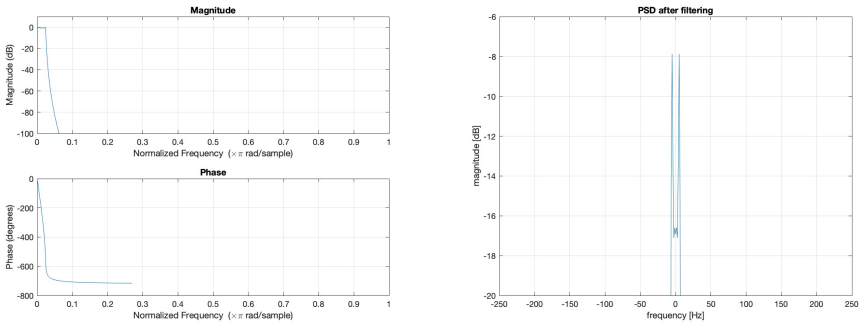

Figure 10: Chebyshev filter frequency response and PSD after filtering

In Figure 10 it is possible to see the frequency response of the Chebyshev filter. This time we can see that only the 5Hz frequencies deltas are present.

In Figure 11 we can see the comparison of our filter before and after the low-pass filtering. We can see how the the 5Hz sinusoid is visible, though it is disturbed. If we set the variance of the noise to 1, the 5Hz sinusoid is much clearer. If we decrease again the variance at 0.1 then the 5Hz sinusoid is almost clear of noise and perfectly visible.

2 Design of digital FIR filters

2.1 Exercise 4

2.1.1 Gibbs phenomenon

For this exercise we plot the following functions of sine integral to observe the Gibbs phenomenon

- $\frac{\mathrm{Si}(\pi T(f+\frac{B}{2}))}{\pi}$
- $-\frac{\mathrm{Si}(\pi T(f+\frac{B}{2}))}{\pi}$
- $\frac{\mathrm{Si}(\pi T(f+\frac{B}{2}))-\mathrm{Si}(\pi T(f+\frac{B}{2}))}{\pi}$

The plot is shown in Figure 12

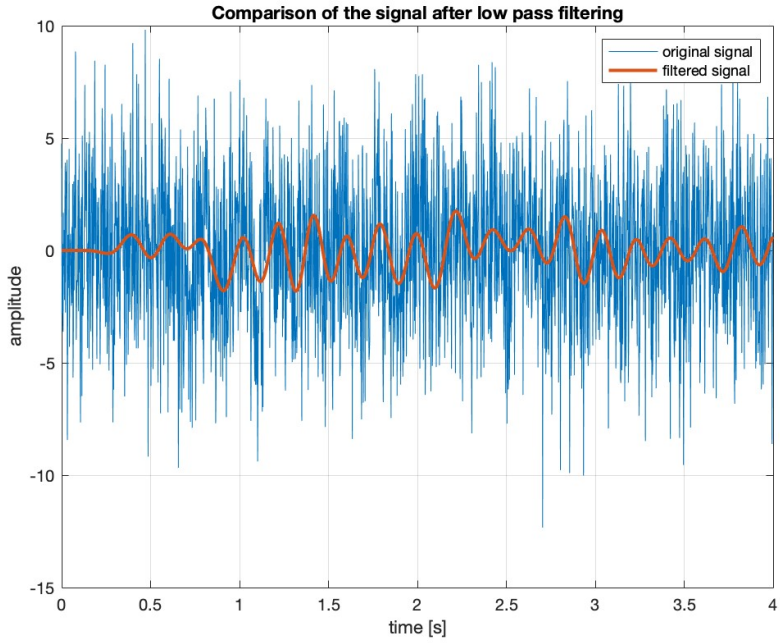

Figure 11: Signal comparison after low-pass filter

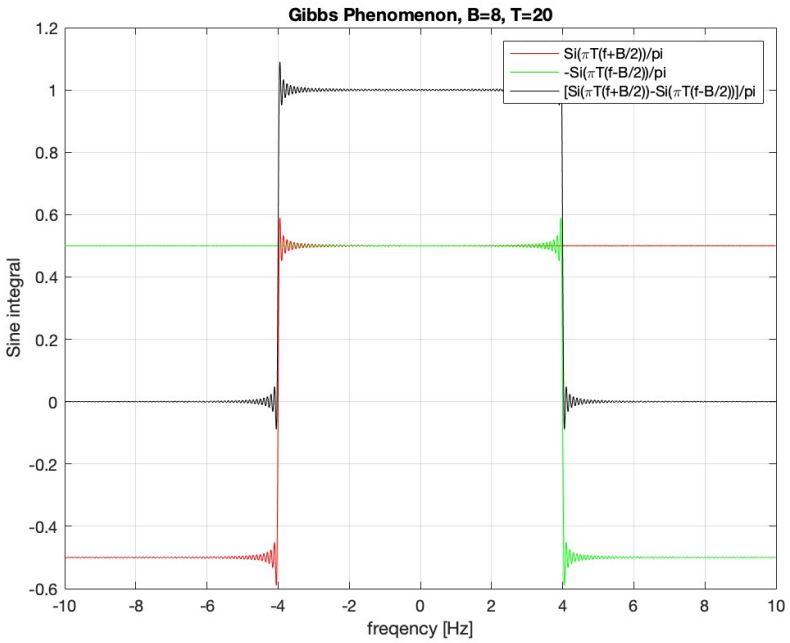

Figure 12: Gibbs phenomenon

2.1.2 Sinc function

We now generate a sinc function with the same initial parameters used B and T used to compute the previous sine integrals. We require 400 samples. The plot can be visualized in Figure 13

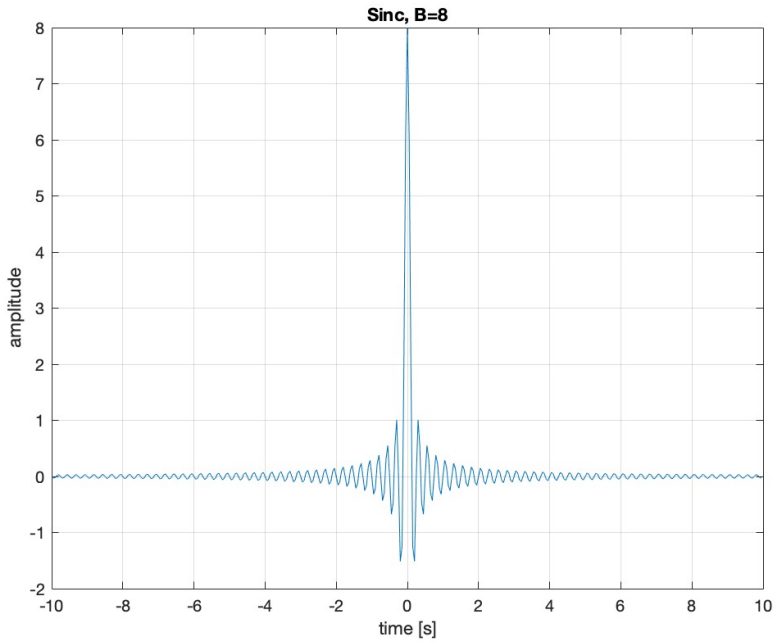

Figure 13: Sinc function

2.1.3 Comparison between DFT and CTFT

After zero-padding the generated sinc signal to increase the spectral resolution, we compute its DFT.

In Figure 14 we can see the results of the comparison. The two signals match.

2.1.4 Time window increase

We now increase the duration of the time window to observe the change in the ripple frequency and amplitude.

First we duplicate the duration, then we duplicate again.

In Figure 15 we can see the effect duplicating the duration of the time window. We can notice that the ripple frequency doubles, but the amplitude stay unchanged.

In Figure 16 we can observe the same behaviour if we double the duration of the time window again.

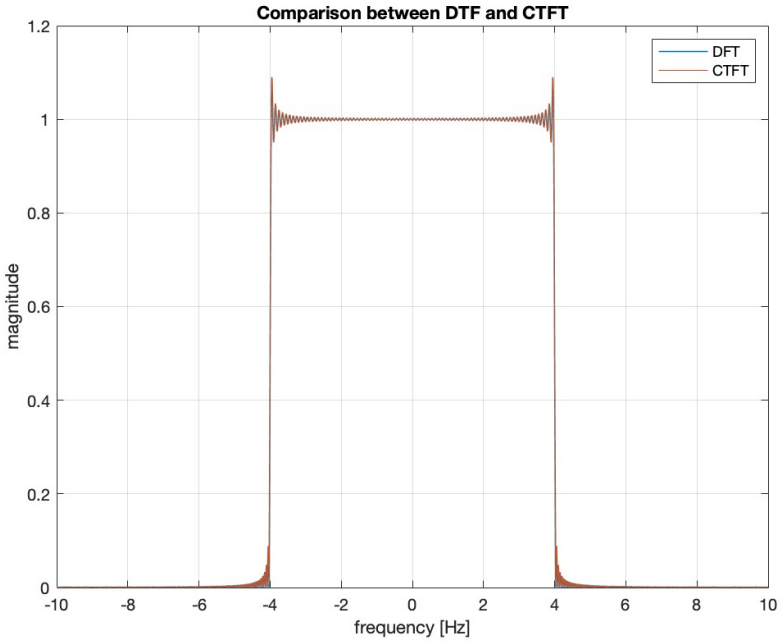

Figure 14: Comparison between sinc DFT and sine integral function

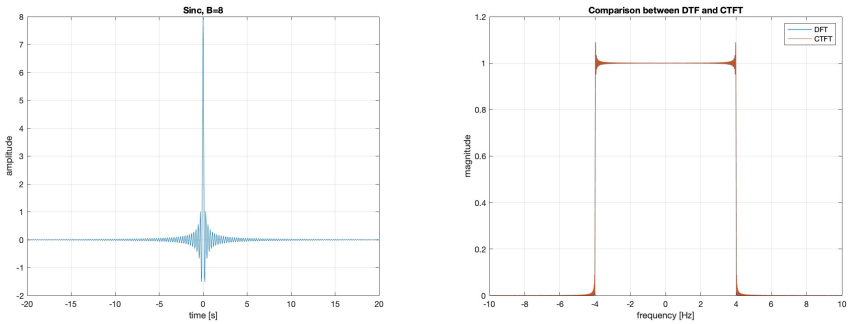

Figure 15: Comparison between sinc DFT and sine integral function, $T = 40$

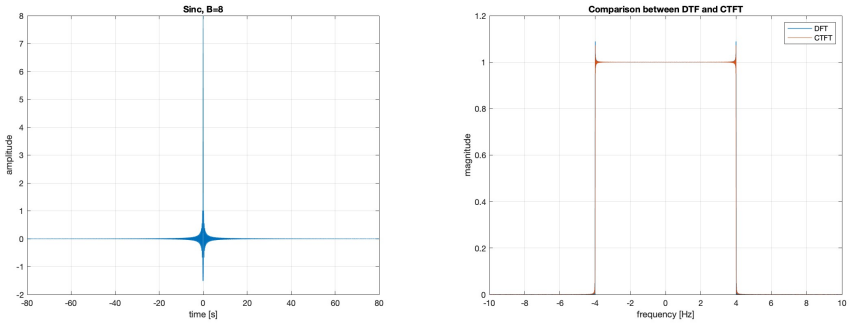

Figure 16: Comparison between sinc DFT and sine integral function, $T = 160$

2.2 Exercise 5

In this exercise we design three FIR filters with the window method. For each filter design we choose an appropriate window that satisfy the stop band attenuation and a cut-off frequency in the middle of the transition band. Consequently we find the lowest number of coefficient. After computing the window we can compute the ideal filter with proper spectral transformation formulas.

2.2.1 High Pass filter

The stop band attenuation for this high-pass filter is $A_s = 40$dB. Thus we use a Hann window. We can find the minimum number of coefficient required by computing $N = 6.1\pi/B_T$, where B_T is the transition band.

The stop-band edge is 140Hz and the pass-band edge is 160Hz.

In Figure 17 the frequency response of the high pass filter is plotted. As we can see it respects the given requirements.

2.2.2 Band Pass filter

In this case we should design a filter with pass band region of 28Hz centered at 80Hz. The transition band is 12Hz.

The stop band attenuation required is 50dB. For this reason we choose a Hamming window. The required number of samples is computed as $N = 6.6\pi/B_T$.

In Figure 18 the frequency response of the filter is plotted. As expected we have the stop band edges at 54Hz and 106Hz, while the pass band edges at 66Hz and 94Hz.

2.2.3 Stop Band filter

This last filters require a stop band attenuation of 70dB. We use a Blackman window, thus the required number of samples is $N = 11\pi/B_T$.

The stop band edges are at 102Hz and 158Hz with 16Hz of transition band.

We can visualize the frequency response in Figure 19.

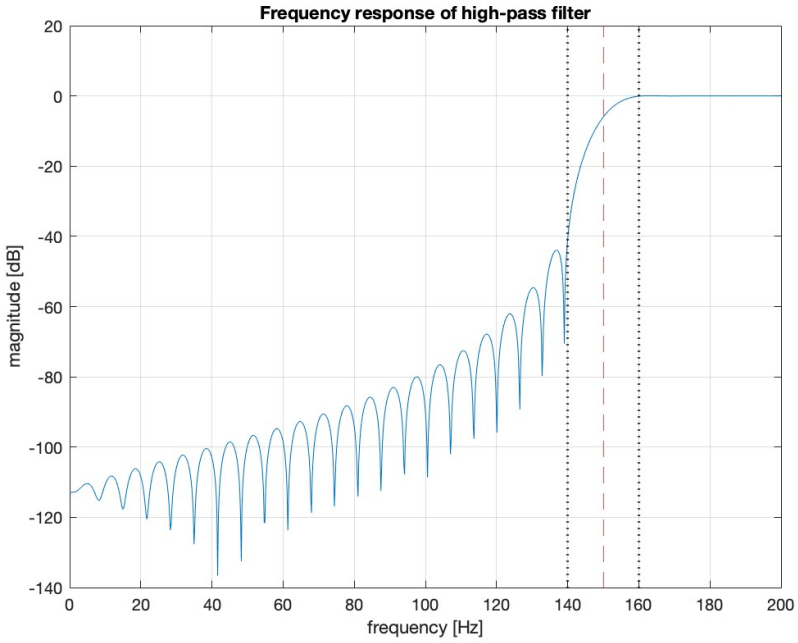

Figure 17: Frequency response of the high-pass filter

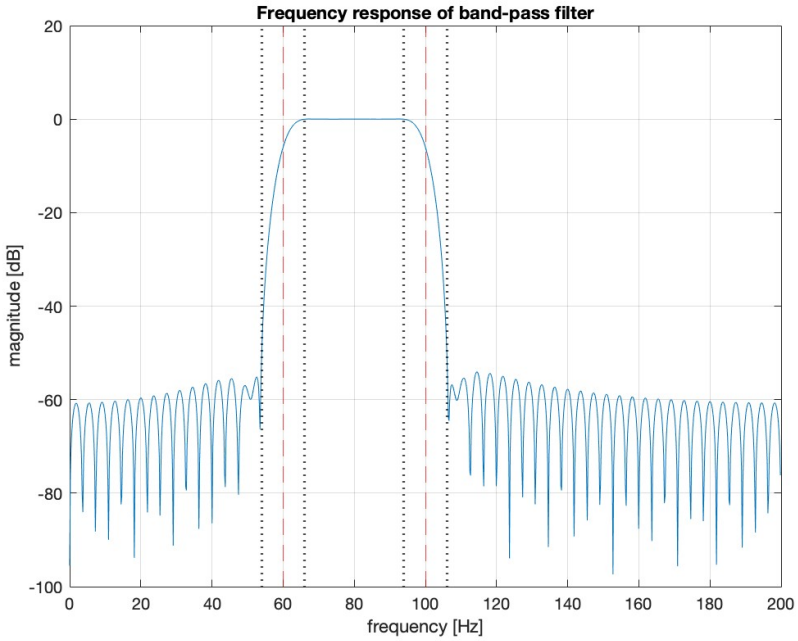

Figure 18: Frequency response of the band-pass filter

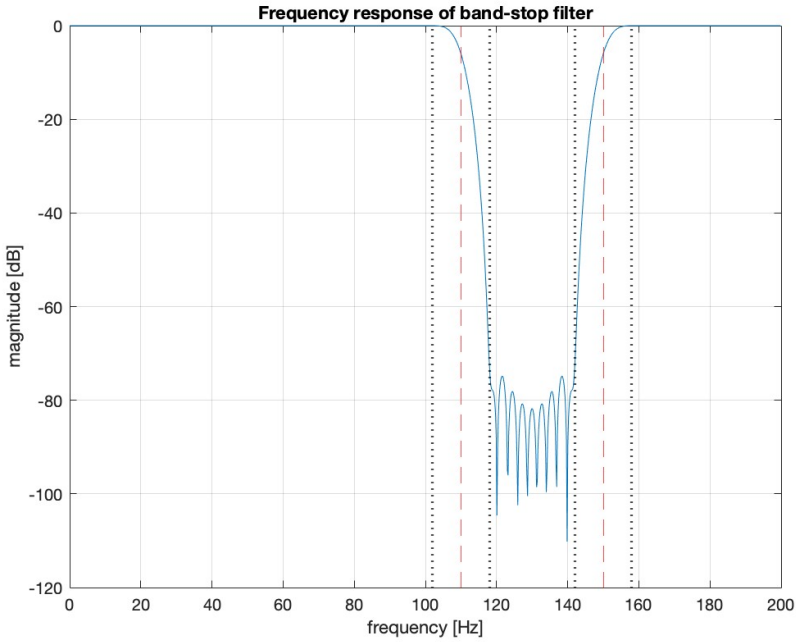

Figure 19: Frequency response of the stop-band filter

2.3 Exercise 6

For this exercise we are required to create a custom function called *my_Kaiser_filter* which takes as inputs the stop-band attenuation A_s, the transition band B_T, the cut-off frequency f_c, the sampling frequency f_s and the filter type.

The function outputs the filter coefficient h, the number of coefficients N and the parameter β.

2.3.1 Implementation

As first, we check that the input parameters are correct. Then we can proceed to compute the Kaiser window.

The β parameter is computed as

$$\beta = \begin{cases} 0.1102 \cdot (A_s - 8.7) & A_s > 50 \\ 0.5842 \cdot (A_s - 21)^{0.4} + 0.07886 \cdot (A_s - 21) & 21 \leq A_s \leq 50 \\ 0 & A_s < 21 \end{cases}$$

Then we can compute the number of coefficient as

$$N = \left\lceil \frac{A_s - 8}{2.285 \cdot B_T} \right\rceil$$

It is now possible to compute the Kaiser window as

$$w(n) = \begin{cases} \frac{I_0\left[\beta\sqrt{1-(\frac{2n-N+1}{N-1})^2}\right]}{I_0[\beta]} & 0 \leq n \leq N-1 \\ 0 & \text{elsewhere} \end{cases}$$

After computing the Kaiser window, the ideal filter is generated based on the filter type. The vector of coefficient h is obtained by multiplying the window by the ideal filter.

2.3.2 Test

Let's test the custom function with a high pass filter with cut-off frequency at 3.2KHz and stop band attenuation of 40dB.

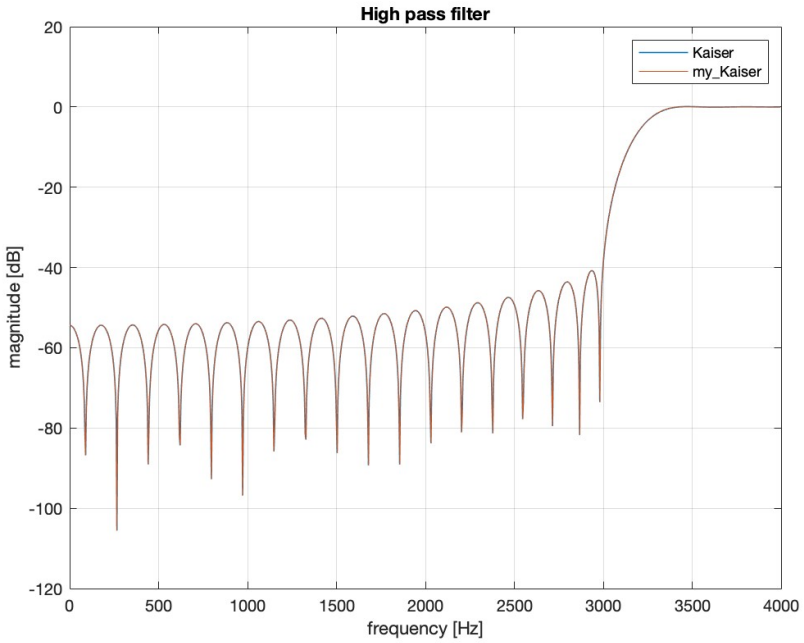

Figure 20: Frequency response of low-pass filter

In Figure 20 is plotted the frequency response computed with the custom function and compared with the a filter computed using the Matlab function *Kaiser*.

We test again the custom function with a band pass filter with cut-off frequency at 1.6KHz and 3.2KHz and 60dB of stop band attenuation.

In Figure 21 we can see that the frequency response of the band-pass signal generated with the custom function coincide with the one generate by the built-in Matlab function.

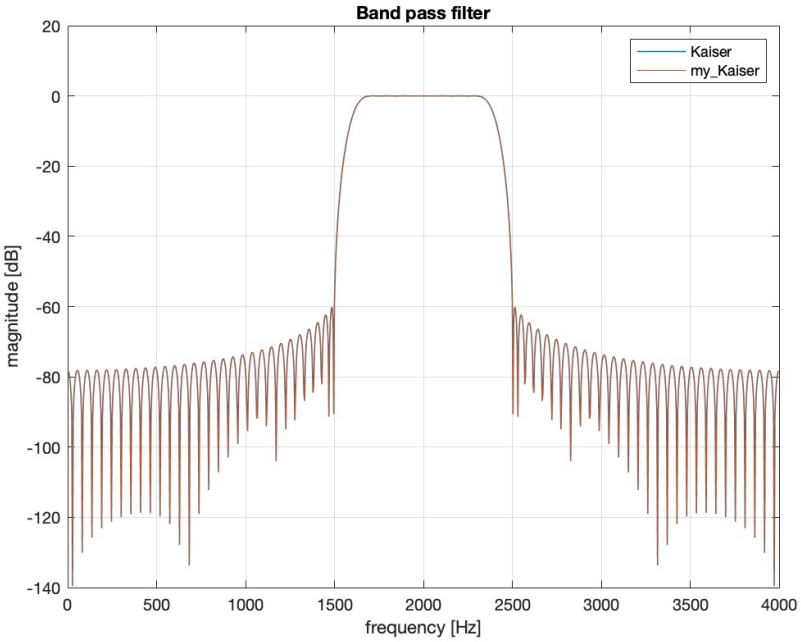

Figure 21: Frequency response of band-pass filter

3 Design of IIR filters and processing of biosignals

3.1 Exercise 7

In this exercise we are asked to design a low pass IIR filter first finding its analog prototype, and then getting its discrete version. In the end we transform first to an high pass and later to a band pass filter.

3.1.1 Analog prototype

The analog expression for the analog prototype of a Butterworth low pass filter is

$$|H_a(j\Omega)|^2 = \frac{1}{1 + (\frac{\Omega}{\Omega_c})^{2N}}$$

where Ω_c is the cutoff frequency and N is the filter order.
We can find Ω_c as the mean value between this two cutoff frequency

$$\Omega_c = \frac{\Omega_p}{\sqrt[2N]{10^{R_p/10} - 1}}$$

$$\Omega_c = \frac{\Omega_s}{\sqrt[2N]{10^{A_s/10} - 1}}$$

and we can get the value of N as

$$N = \left\lceil \frac{\log_{10}[\frac{10^{R_p/10}-1}{10^{A_s/10}-1}]}{2\log_{10}(\Omega_p/\Omega_s)} \right\rceil$$

We take as parameters

- $\Omega_p = 2\text{Hz}$
- $\Omega_s = 6\text{Hz}$
- $R_p = 1\text{dB}$
- $A_s = 70\text{dB}$

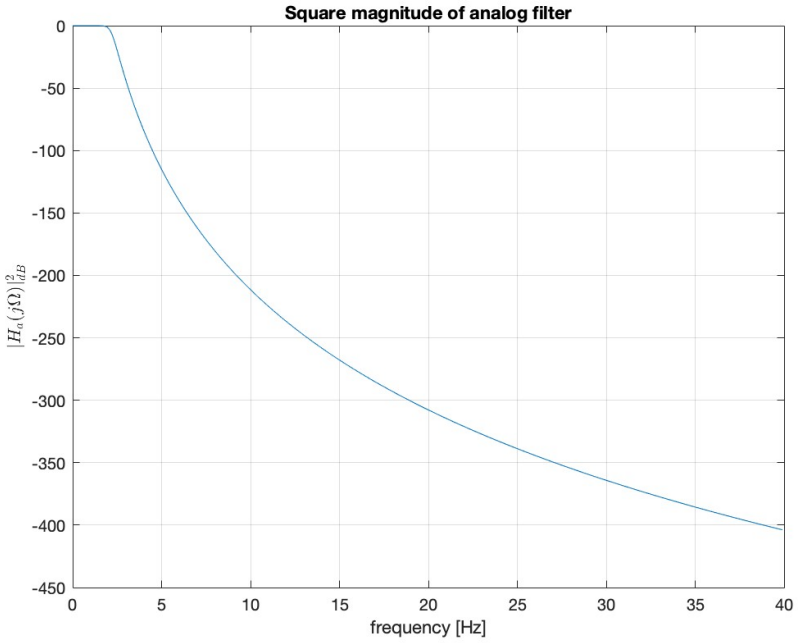

Figure 22: Square magnitude of the frequency response of the analog prototype of the low-pass Butterworth filter

We now find the poles of the transfer function. We only need to compute half of the poles since the other half are the complex conjugate of the previous.

In Figure 23 we can see that all poles lay on the plane negative half, which assure stability.

3.1.2 Discrete-time domain

Let's now convert our analog model to a discrete model. We use two methods: the impulse invariance method and the bilinear transform method. We use sampling frequency 40Hz.

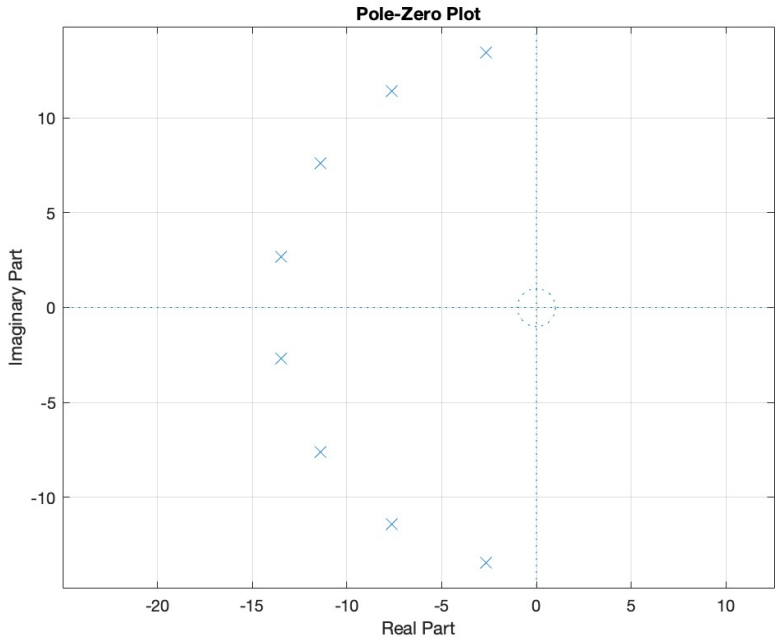

Figure 23: Poles of the transfer function of the analog filter

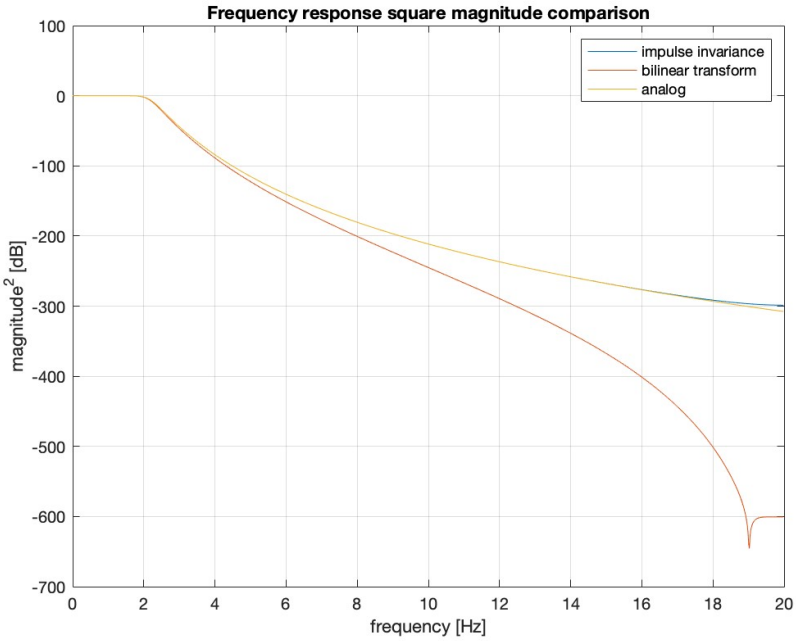

Figure 24: Comparison between square magnitude of the frequency responses ($f_s = 40$Hz)

In Figure 24 it can be observed the comparison between the frequency response of the digital filter obtain with the impulse invariance method and the bilinear transform with the analog prototype. As we can notice the impulse invariance method presents some aliasing on the tails as expected. The bilinear transform instead present a drop.

We now double the sampling frequency to 80Hz.

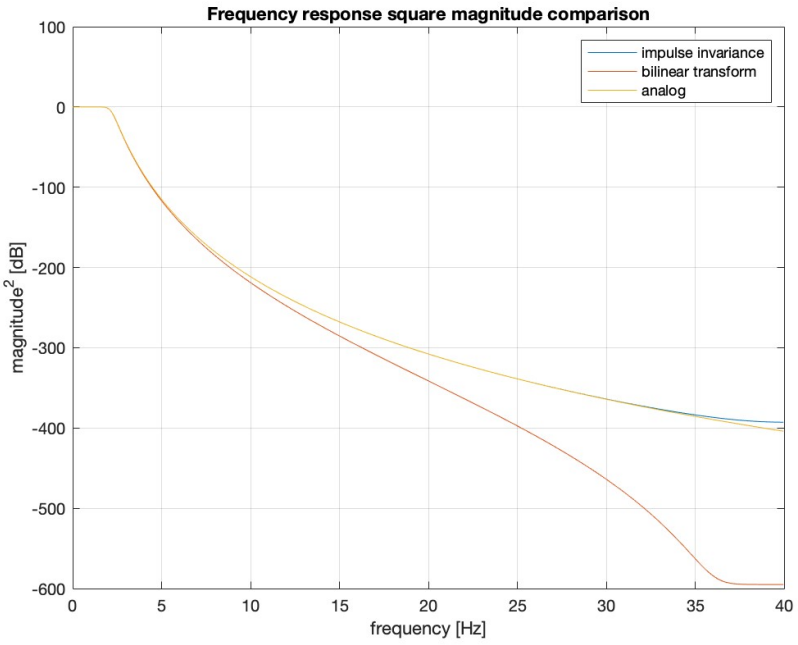

Figure 25: Comparison between square magnitude of the frequency responses ($f_s = 80$Hz)

In Figure 25 it can be seen how increasing the sampling frequency will increase the frequency range of the filter.

3.1.3 Filter transformation to high pass

We can now transform our IIR low pass filter to a high pass filter with the use of the *iirlp2hp* Matlab function

3.1.4 Filter transformation to band pass

Again we transform our IIR low pass filter to a band pass filter with the use of the *iirlp2bp* Matlab function

3.2 Exercise 8

This exercise consist in the calculation of the pulse rate and the oxygen saturation from data taken from a pulse oximeter.

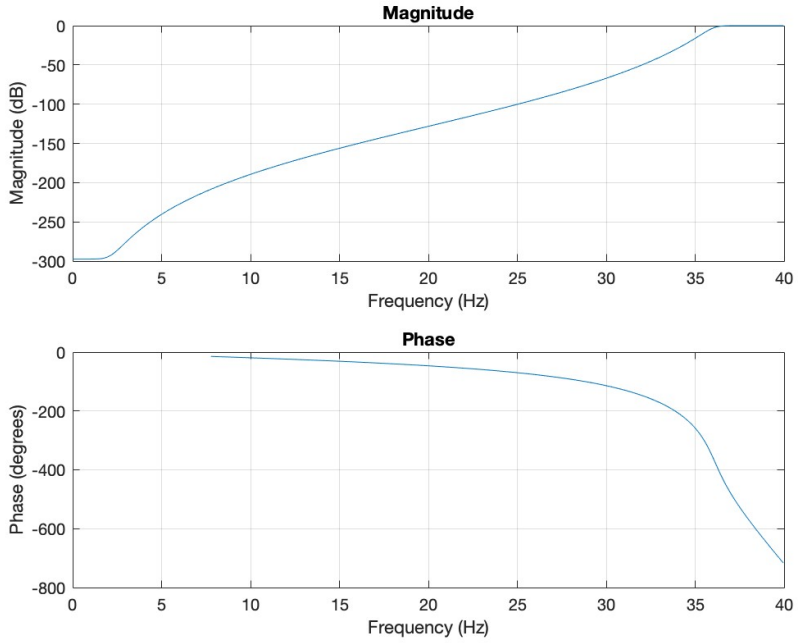

Figure 26: High pass IIR filter frequency response

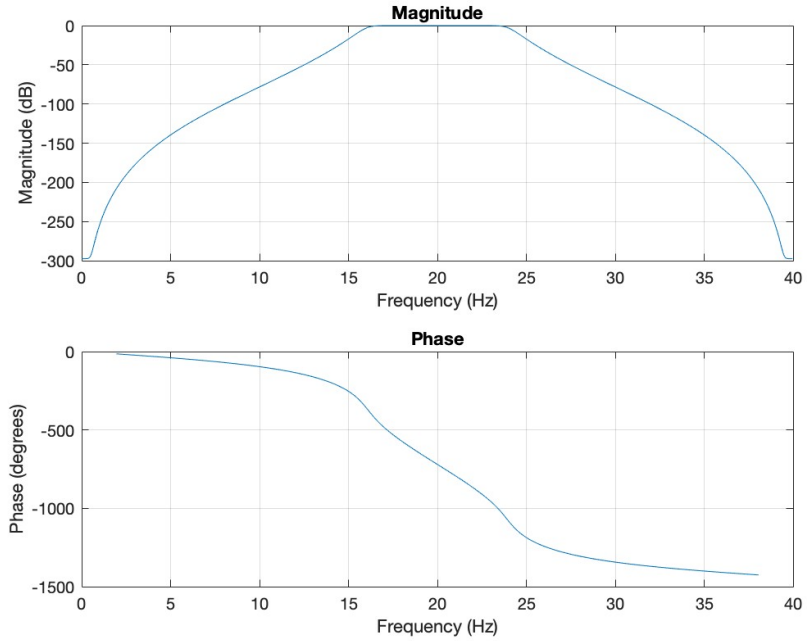

Figure 27: Band pass IIR filter frequency response

3.2.1 Data imported

The pulse oximeter uses two signal. A Red light and an infrared light. After we import them in Matlab we can plot them. We will only use a 60s interval of the whole signal, after discarding the first 10s.

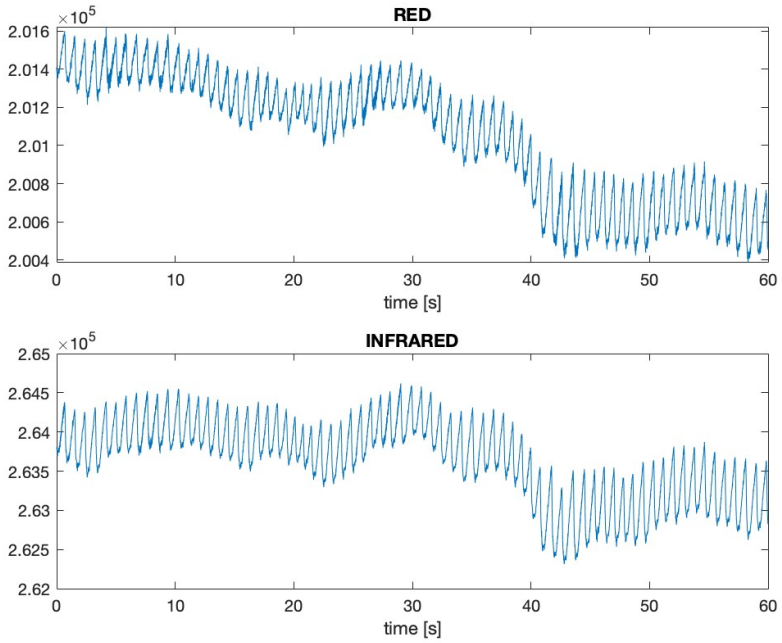

Figure 28: Red and Infrared signal

3.2.2 Signal filtering

In order to properly read the data from the signal we apply a low pass filter to discard all frequency above 180BPM, thus 3Hz. In order to do that we design an IIR filter.

We then apply also an high pass filter to discard the frequency below 0.5Hz. We manage this be designing an FIR filter with the window method.

In Figure 29 can be visualized the frequency response of the two designed filters

In Figure 30 instead the signals after first the low pass and then the high pass filtering is applied.

3.2.3 Pulse rate

To compute the pulse rate we compute the FFT of the filtered signal. The resulting pulse rate will correspond with the maximum value of the frequencies.

In Figure 31 it is plotted the frequency spectrum of the filtered signal. It can be notice that the highest point is located around ± 1.17Hz. Thus the pulse rate is 70.3125BPM.

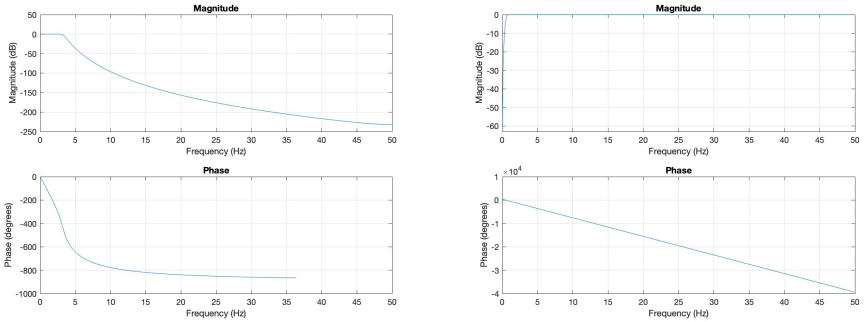

Figure 29: Frequency response of low pass and high pass filter

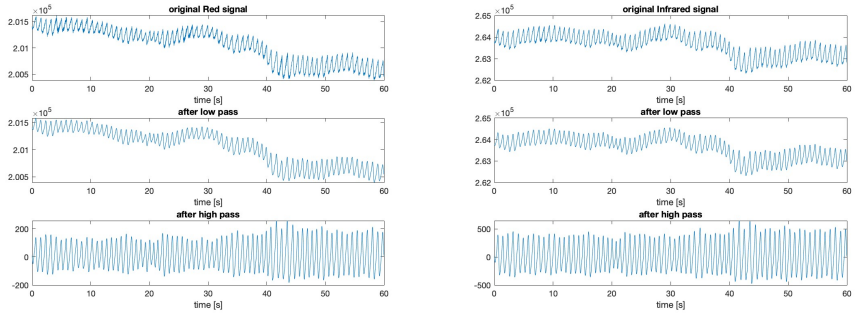

Figure 30: Signals after low pass and after high pass filtering

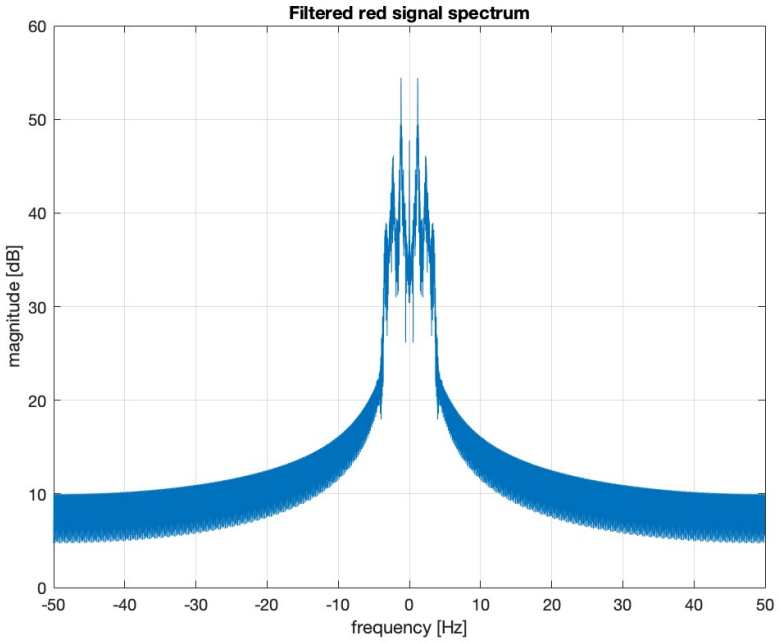

Figure 31: Filtered signal spectrum

3.2.4 Saturation

In order to find the value of the saturation, we need to compute $\bar{R}$ as

$$\bar{R} = \left\langle \frac{\frac{I_{AC}(RED)}{I_{DC}(RED)}}{\frac{I_{AC}(INFRARED)}{I_{DC}(INFRARED)}} \right\rangle$$

We consider the signal after the low pass filter for the DC component and the signal after both filters for the AC component.

To compute the value of $\bar{R}$ we find the upper and bottom envelope for the AC component and the lower envelope for the DC component of both signals.

The saturation percentage is then found as $SaO_2 = 110 - \bar{R} = 98.1653\%$

In Figure 32 it can be observed the AC and DC components of both signals red and infrared. The value of the saturation is computed and printed as the title of the figure.

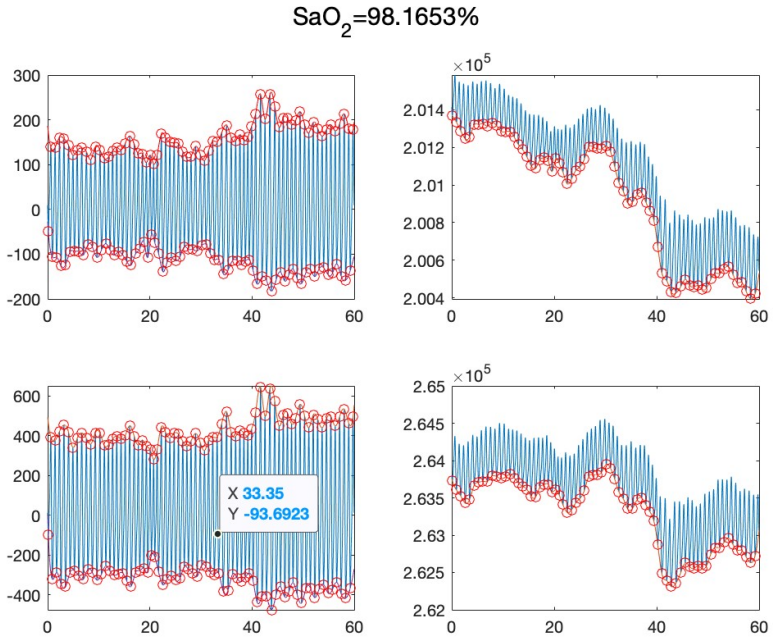

Figure 32: Filtered signal spectrum