Digital Signal Processing A.Y. 2019/2020
Homework 2

Donald Shenaj
1238939

January 6, 2020

Contents

Exercise 1

1.1 Vuvuzela sound

A football match recording from World Cup has an annoying vuvuzela noise described by a loud monotone note at about 235Hz, which is the first harmonic f_0. The following are at about 465Hz, 705Hz and so on. The original audio in visible in Figure 1.1.

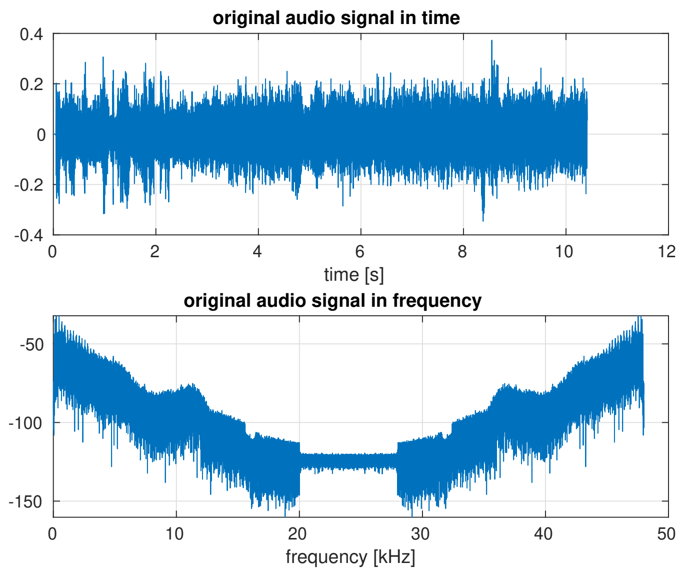

Figure 1.1: Original audio: football match recording

1.2 IIR notch filter design

We have to design a IIR notch filer to remove the unwanted frequencies. In order to do so we can exploit what we have seen in class for a second order IIR notch filter and extend the expression of $H(Z)$ in such a way to remove more than one harmonics in the range $[0, \frac{F_p}{2}]$. For instance we choosed to remove five harmonics $(f_0, f_1, f_2, f_3, f_4, f_5)$, so the new expression will have ten poles and zeros:

$$H(Z) = \frac{\prod_{i=1}^{10}(1 - z_i Z^{-1})}{\prod_{i=1}^{10}(1 - p_i Z^{-1})} \tag{1.1}$$

We found the values of the poles p_i and zeros z_i, visible in Figure 1.2, according to the following relations:

$$\begin{aligned} p_k &= re^{i\theta_k} \\ z_k &= e^{i\theta_k} \\ p_{k+5} &= {p_k}^* \\ z_{k+5} &= {z_k}^* \\ k &= [1,2,3,4,5] \\ \theta &= 2\pi T[f_0, f_1, f_2, f_3, f_4, f_5] \\ r &\simeq 1 \end{aligned} \tag{1.2}$$

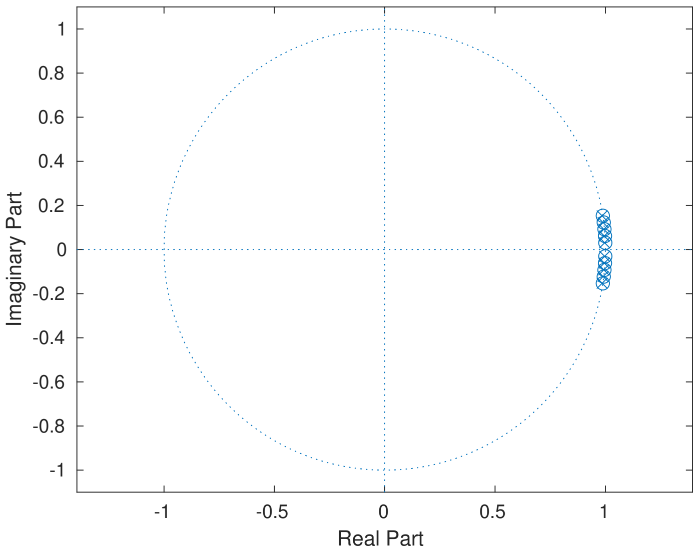

Figure 1.2: Poles and Zeros in Z plane

1.3 Frequency response

The poles and zeros found can be seen in Figure 1.2. The frequency response of the filter instead can be seen in Figure 1.3. We can tell that the filter is working as expected because we see five frequencies removed, corresponding to the first five harmonics.

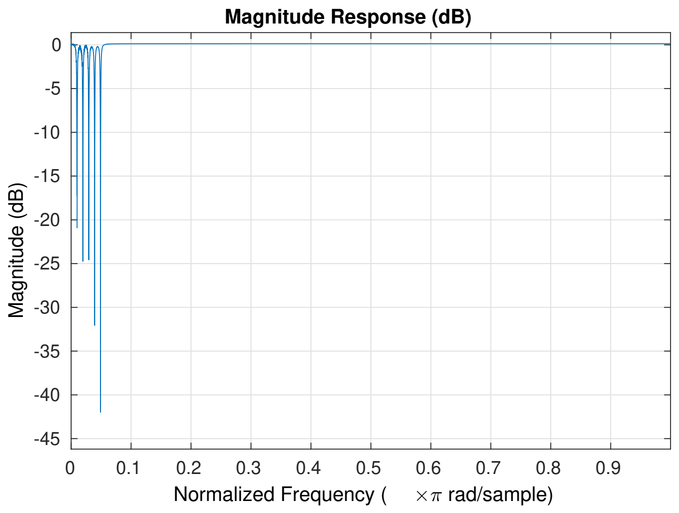

Figure 1.3: IIR notch filter frequency response

1.4 Filtering audio signal

To check if the audio is actually filtered correctly it was produced a wav file called 'donald_shenaj_hw2_1.wav'. We can hear that the sound of the vuvuzelas is lower than in the original one which confirms the correct behaviour.

1.4.1 Matlab iircomb function

Matlab does provide some functions, fore example like 'iircomb', which can be used to implement the IIR notch filter. In this case we set to remove all the harmonics of the vuvuzela sound in our band, which are 102. The frequency response of the filter can be seen in Figure 1.4. To hear the filtered signal was produced a wav file called 'donald_shenaj_hw2_2.wav'. The resulting filtered signal is visible in Figure 1.5 where we compared the results for both flters: the one implemented to remove five harmonics and the complete notch filter implemented with 'iircomb'.

1.4.2 Results

We can say that our filter does work but in order to have a complete removing of the vuvuzela sound is advisable to use larger 'bands' for the notch frequencies.

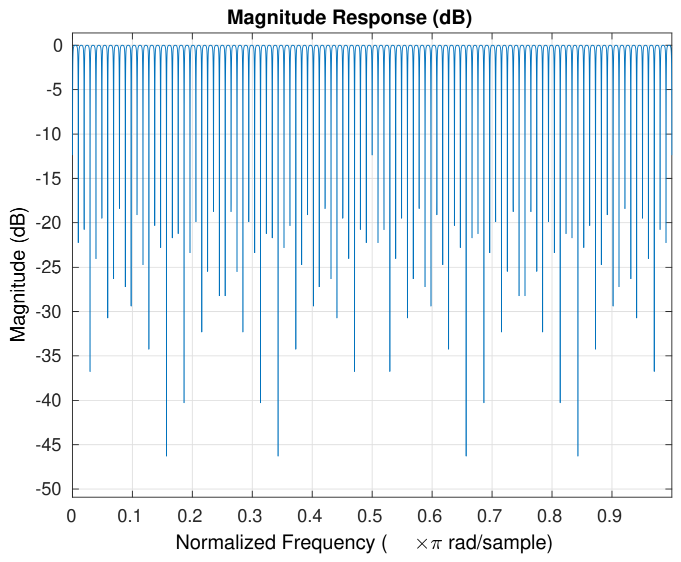

Figure 1.4: Matlab iir notch filter

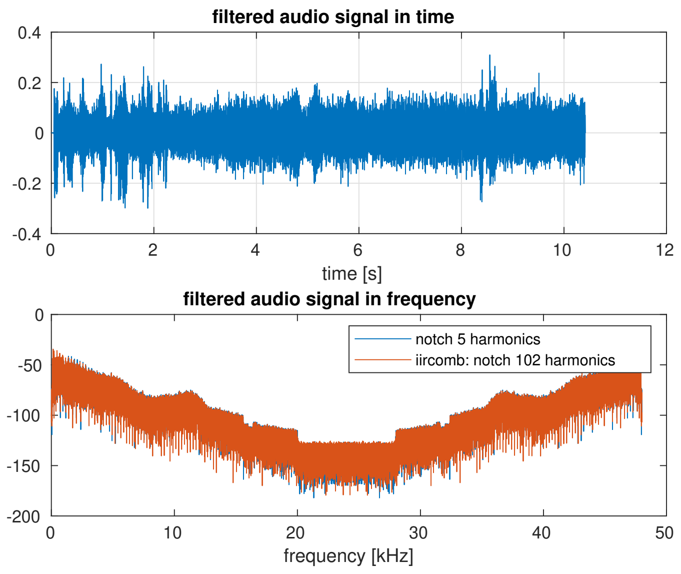

Figure 1.5: Comparision between both filtered audio signals

Exercise 2

2.1 Introduction

In order to implement the RIAA equaliser we'll start from the analog filter with Laplace transform $H_a(s)$ and find the analog filter $H_a{}'(s)$ by multiplying in the expression of $H_a(s)$ the value s by $\frac{2}{T}$:

$$H_a(s) = \frac{(1+sT_2)}{(1+sT_1)(1+sT_3)} \quad \Rightarrow H_a{}'(s) = \frac{TT_2}{2T_1T_3}\frac{\left(s+\frac{T}{2T_2}\right)}{\left(s+\frac{T}{2T_1}\right)\left(s+\frac{T}{2T_3}\right)} \tag{2.1}$$

In this way, when we'll apply the transform method, substituting $s = \frac{1-Z^{-1}}{1+Z^{-1}}$, we will find the digital filter with the desired frequency behaviour.

2.2 IIR filter design

In the analog filter we found the poles $P_1 = -\frac{T}{2T_1}$, $P_2 = -\frac{T}{2T_3}$ and the zero $Z_1 = -\frac{T}{2T_2}$. After applying the transform method we get the following expression:

$$\begin{aligned} H(Z) &= \frac{TT_2}{2T_1T_3}\frac{\left(\frac{1-Z^{-1}}{1+Z^{-1}} - Z_1\right)}{\left(\frac{1-Z^{-1}}{1+Z^{-1}} - P_1\right)\left(\frac{1-Z^{-1}}{1+Z^{-1}} - P_2\right)} \\ &= \frac{TT_2}{2T_1T_3}\frac{(1-Z^{-1}-Z_1(1+Z^{-1}))\,\frac{1}{1+Z^{-1}}}{(1-Z^{-1}-P_1(1+Z^{-1}))\,(1-Z^{-1}-P_2(1+Z^{-1}))\,\frac{1}{(1+Z^{-1})^2}} \\ &= \underbrace{\frac{TT_2}{2T_1T_3}\frac{(1-Z_1)}{(1-P_1)\,(1-P_2)}}_{\text{Constant}}\,\frac{\left(1+Z^{-1}\right)\left(1-\left(\frac{1+Z_1}{1-Z_1}\right)Z^{-1}\right)}{\left(1-\left(\frac{1+P_1}{1-P_1}\right)Z^{-1}\right)\left(1-\left(\frac{1+P_2}{1-P_2}\right)Z^{-1}\right)} \end{aligned} \tag{2.2}$$

From this expression we can see that the IIR filter has $M = N = 2$. We evidenced the constant factor and the discrete domain poles $p_1 = \frac{1+P_1}{1-P_1}$, $p_2 = \frac{1+P_2}{1-P_2}$ and zeros $z_1 = \frac{1+Z_1}{1-Z_1}$, $z_2 = -1$.
Now we can easily find the 'a' and 'b' coefficients of the IIR filter from the discrete domain poles and zeros.

2.3 Frequency response

The frequency and time responses of the designed filter, visibile respectively in Figure 2.1 and 2.2, behave like we have seen in the Homework 1. Note that the frequency response is represented in a frequency axis normalized by the factor $\frac{F_p}{2}$.

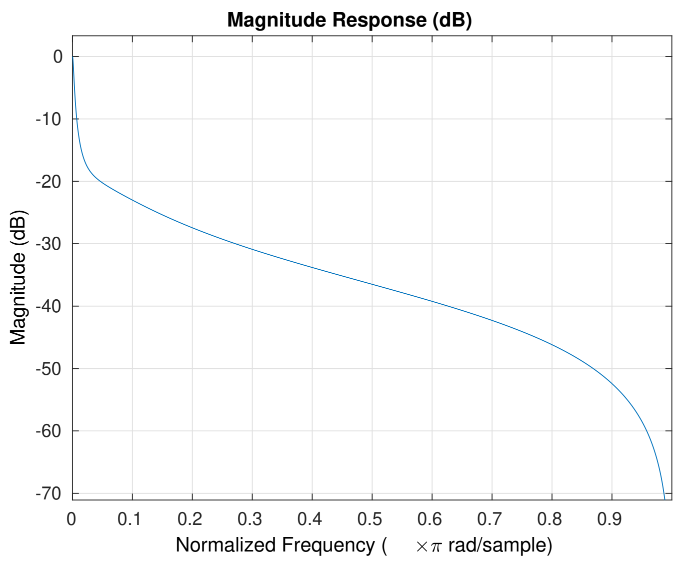

Figure 2.1: Frequency response

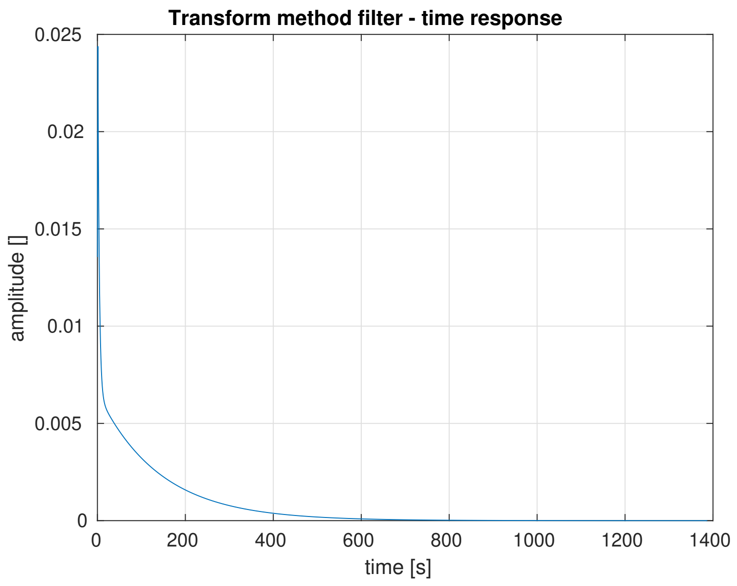

Figure 2.2: Time response

2.4 Filtering audio signal

After applying the filter to the audio sample it was produced the audio file called 'donald_shenaj_hw2_2_1.wav'. By hearing the audio we can say that the filter is working correctly.

2.5 Direct optimization method

We'll implement now the filter adopting a direct optimization method that matches as closely as possible the absolute value $D(f) = |H_a(i2\pi f)|$. We identified the zeros and poles thought linear programming, setting $N = 2$, so the resulting filter will have two poles and two zeros. The frequency response of the filter compared with the reference $D(f)$ is visible in Figure 2.3. The time response is visible in Figure 2.4.

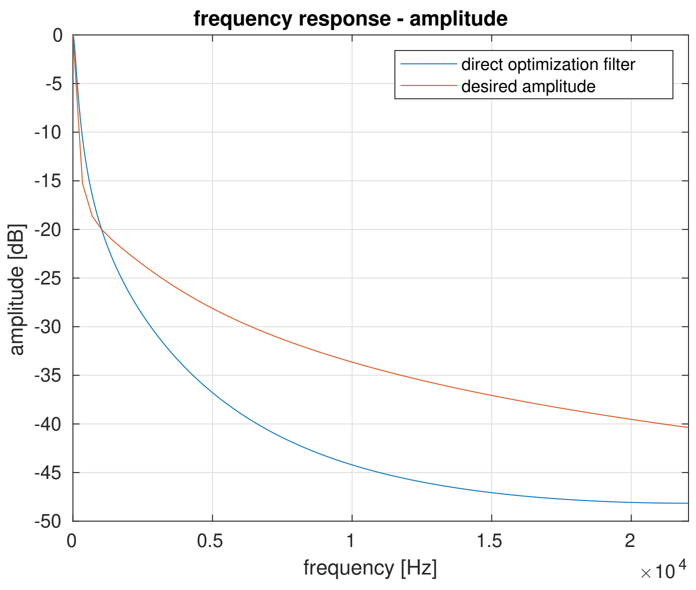

Figure 2.3: Direct optimization filter frequency response

2.5.1 Results

Also in this case we can tell that the filter is working as expected, and it's very similar to the behaviour obtained in the Homework 1. It was produced a file called 'donald_shenaj_hw2_2_2.wav' to hear the differences between the two implementation of this exercise.

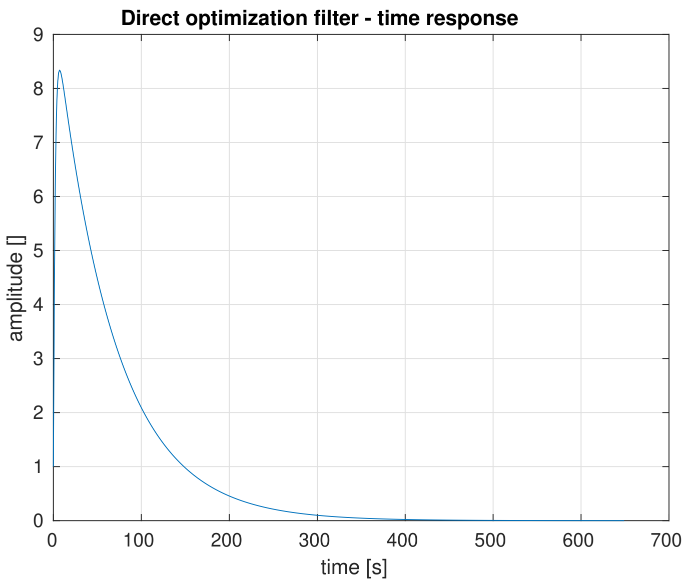

Figure 2.4: Direct optimization filter time response

Exercise 3

3.1 Design of a rate conversion algorithm

In this exercise we have to implement a rate conversion system in order to resample the input audio signal $x(nT)$ from the frequency $F_1 = 44.1\text{kHz}$ to $F_2 = 48\text{kHz}$. Since we have $F_2 > F_1$ $(L > M)$ the filter should be a low pass with $f_0 = \frac{F_1}{2} = \frac{F_p}{2L}$. The block diagram in visibile in Figure 3.1.

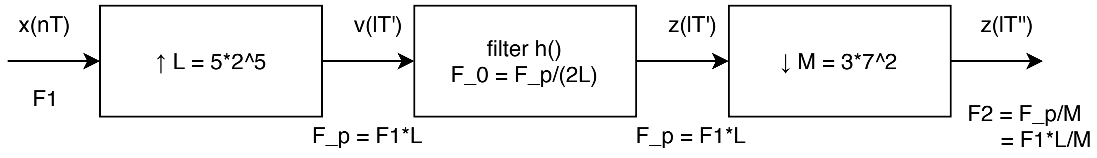

Figure 3.1: Rate conversion block diagram

3.2 FIR filter design

We designed a FIR filter by Remez for his semplicity, but it could have been possible to use instead an IIR for reducing the order. In Figure 3.2 are visibile the time and frequency responses and in Figure 3.3 there is a zoom.

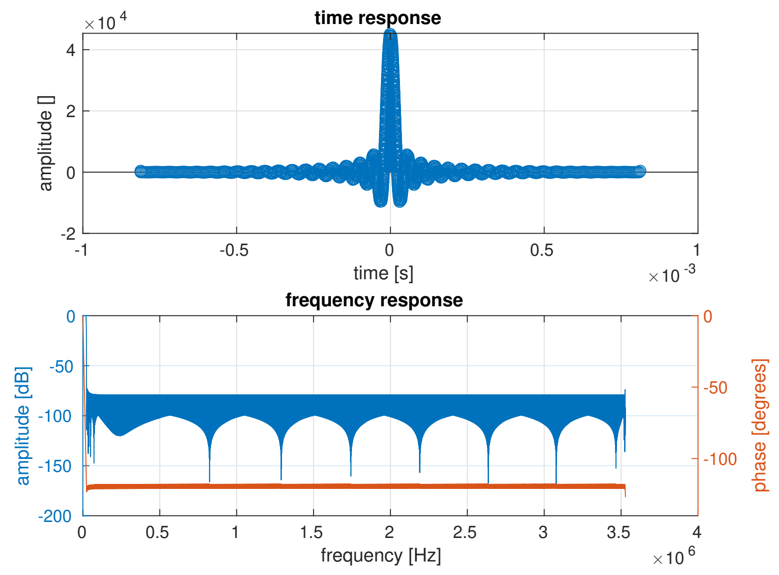

Figure 3.2: FIR filter frequency and time responses

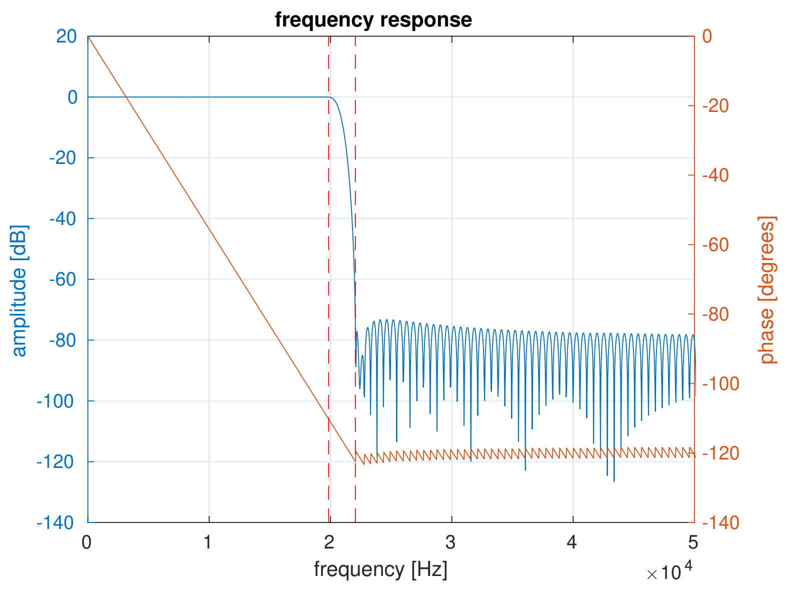

Figure 3.3: Zoom in the range $f \in [0, 5e4]$Hz

3.3 Audio rate conversion

It was produced a file called 'donald_shenaj_hw2_3_1.wav' to hear the result of the rate conversion system. Even if it's working properly in this solution we see that the order of the filter $N = 12473$ is very high, so we can try to implement the system using a multistage approach.

3.4 Multistage rate conversion

We'll implement now the multistage rate conversion using the same rationale used before. The new block diagram of the system is visible in Figure 3.4.

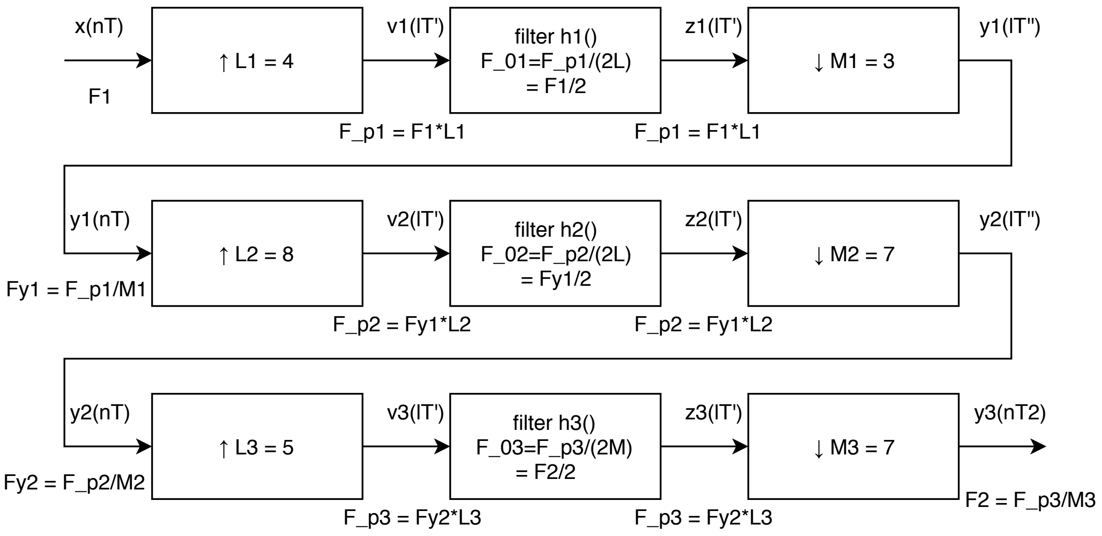

Figure 3.4: Multistage rate conversion block diagram

3.4.1 Filters

For designing the low-pass filters we followed the same approach as we did before, so we implemented three FIR filters using Remez. We tried also to define the transition bandwith efficiently in such a way to reduce the order of the filters. For the first filter we used as upper limit frequency the value $F_{p1} = f_0 * 0.9$ and stop band lower limit the value $f_{01} = \frac{F_1}{2}$ since we have $F_1 < F_{y1}$, where $F_{yi} = \frac{F_i L_i}{M_i}$. For the second filter we choosed as upper limit frequency the value $F_{p2} = F_{p1}$ as before and as stop band limit the value $f_{02} = \frac{F_{y1}}{2}$ since we have $F_{y1} < F_{y2}$. For the last filter we choosed as upper limit frequency the value $F_{p3} = F_{p1}$ as before and as stop band limit the value $f_{03} = \frac{F_2}{2}$ since we have $F_{y2} > F_2$. The frequency and time responses of all the filters can be seen respectively in Figure 3.5, 3.6 and 3.7.

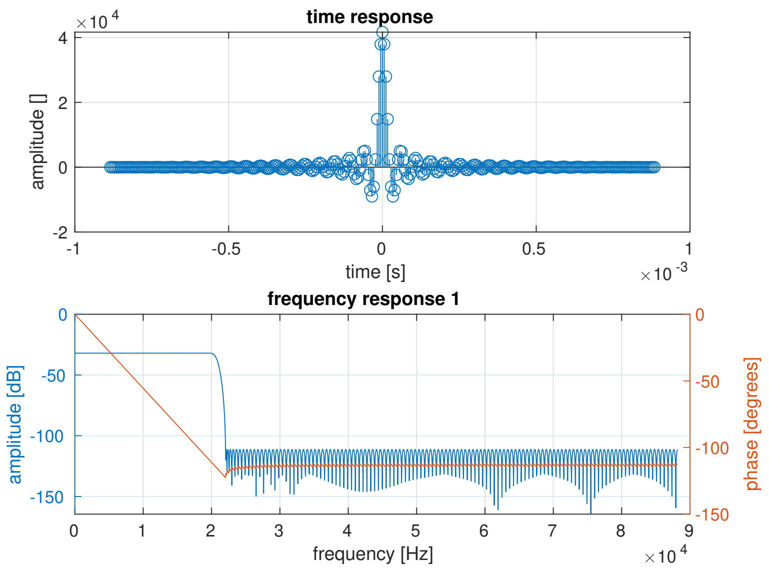

Figure 3.5: First stage FIR filter, $f_{01} = \frac{F_1}{2}$, $N_1 = 312$

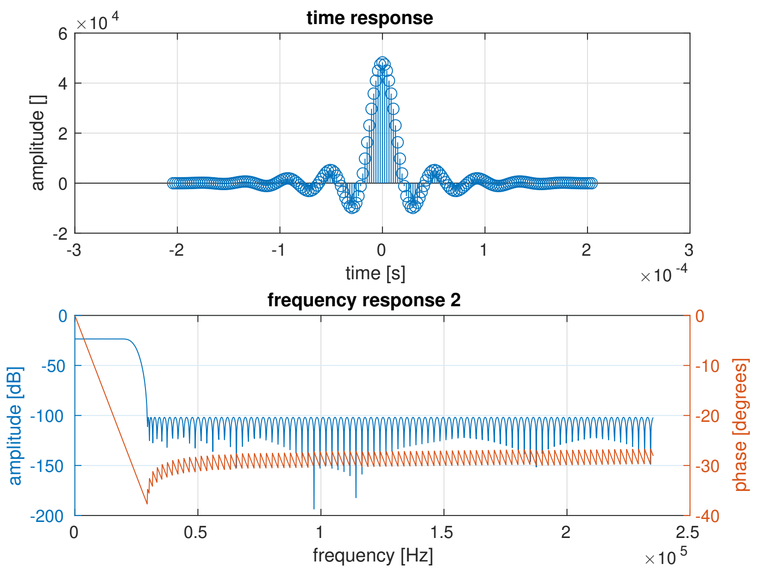

Figure 3.6: Second stage FIR filter, $f_{02} = \frac{F_1 L_1}{2M_1}$, $N_2 = 192$

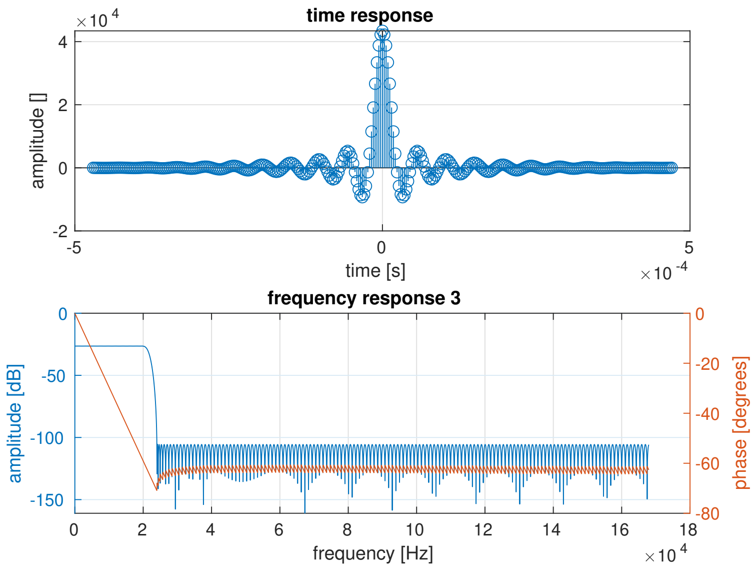

Figure 3.7: Third stage FIR filter, $f_{03} = \frac{F_2}{2}$, $N_3 = 316$

3.4.2 Results

We see that by using a multistage approach we can reduce a lot the number of coefficients used for filtering the signal. In fact from $N = 12473$ we get $N = N_1 + N_2 + N_3 = 312 + 192 + 316 = 820$, where N_1, N_2 and N_3 are the order of the filters used in the three stages. So we reduced more than 90% the number of coefficients. For more improvement we could adopt IIR filters. The result of the rate conversion throught the multistage system can be heard in the audio file donald_shenaj_hw2_3_2.wav' In Figure 3.8, 3.9 and 3.10 are represented respectively the original audio, the rate-converted audio trought single stage and the multistage rate-converted audio, both in time and frequency. From those figures we confirm again the correct behavior of both the systems.

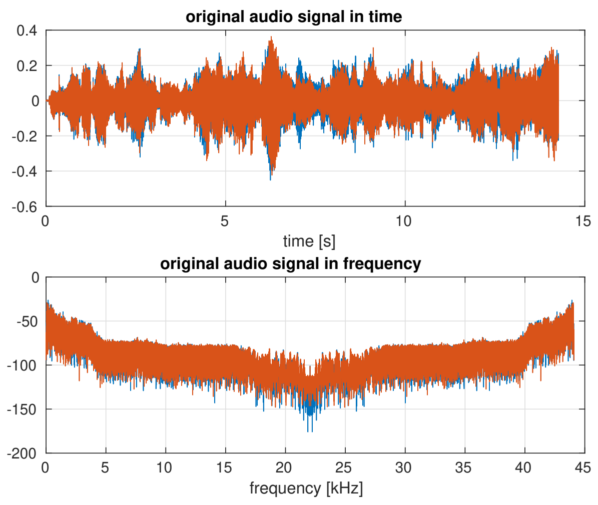

Figure 3.8: Original audio signal in time and frequency

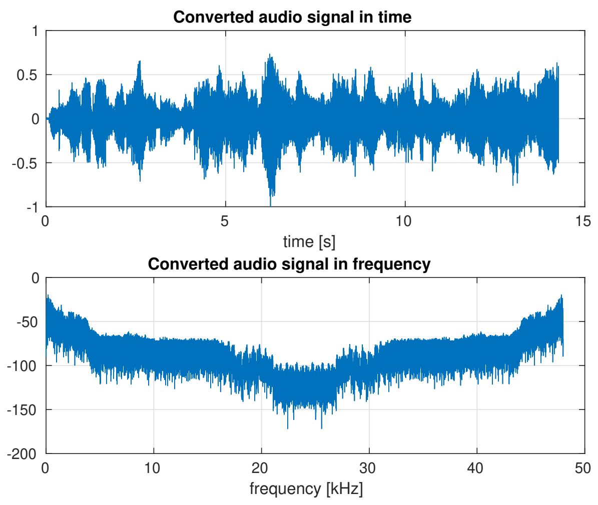

Figure 3.9: Single stage converted audio signal in time and frequency

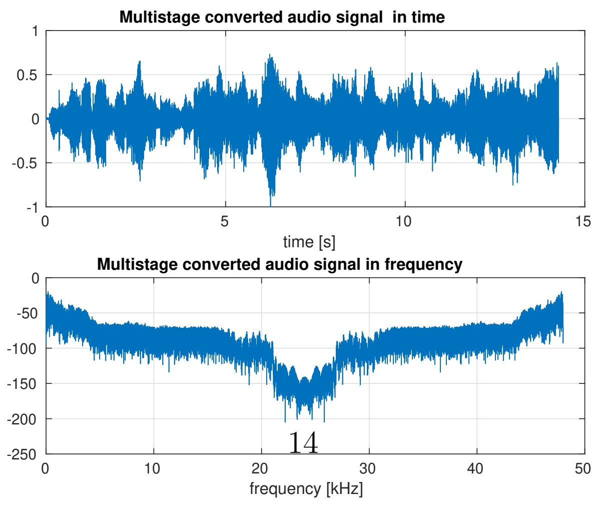

Figure 3.10: Multistage converted audio signal in time and frequency