
	<p style="text-align: center;">Faculty of Mechanical Engineering and Robotics</p> <p style="text-align: center;">Department of Robotics and Mechatronics</p>	
<h2 style="margin: 0;">Mechatronic Systems Identification</h2>		
<p>Lab 2: Digital filtration: FIR and IIR</p>		

1. Filtration methods in Matlab

Create a **y** signal with a length of 1000 samples and a sampling frequency of 1000 Hz, consisting of the sum of two sine signals with frequencies of 10 and 300 Hz and amplitudes of 1 and 0.2. Create a rectangular window **h** with a length of 21 samples and an amplitude of each sample of 1/21. Perform the following exercises.

Ex. 1.1

Carry out the filtration by convolution of the input signal with a rectangular window. Use the function below (**conv**, convolution). Note that the length of the signal after the convolution is equal to the sum of the lengths of the convoluted signals minus one sample. So the signal should be shortened to the length of the original signal

```
yc = conv(y,h);
yc = yc(1:1000);
```

Perform the same operation using **filter** function:

```
yf = filter(Num,1,y);
```

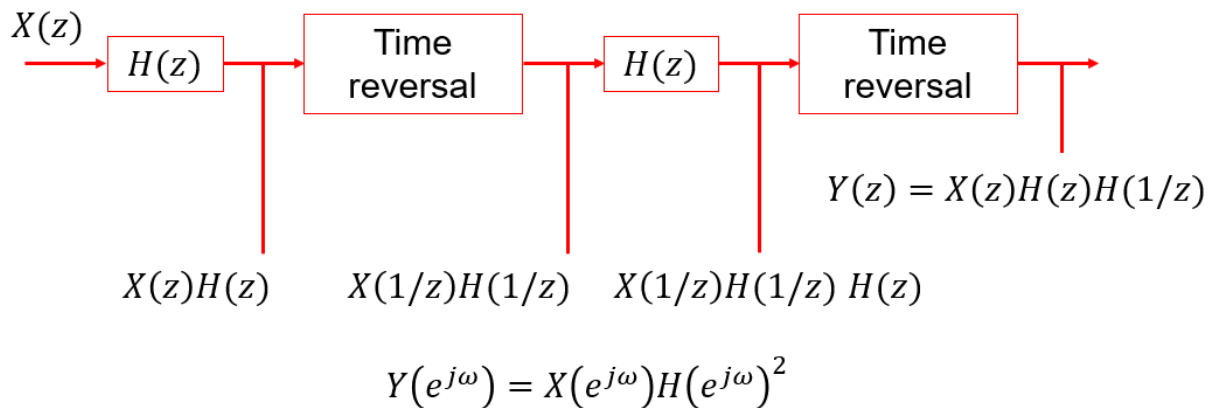
Take the values of the rectangular window vector **h** as the coefficients of the filter numerator **Num**. Compare the obtained results in the time domain and their frequency spectrum. Such a comparison can be made, for example, with the **hold on - hold off** functions:

```
figure(1)
plot(t,y)
hold on
plot(t,yf)
plot(t,yc,'-.')
hold off
xlabel('czas (s)')
ylabel('amplituda (~)')
```

Plot the window characteristics using the function `freqz(Num,Den,s,fs)`, where Num and Den are the numerator and denominator coefficients of the filter, s is the number of samples in the filter characteristic, and fs is the signal sampling rate. See if the changes in the amplitude of the frequency components of the input signal agree with the characteristics of the filter.

Ex. 1.2

Apply zero-phase filtering with the `filtfilt()` function and compare the result with the `filter()` function. Then try to implement your own method to obtain zero-phase filtering considering the diagram below. Note: the time reversal in Matlab can be written as `x(end:-1:1)`.



Ex. 1.3

Change the window length to 11 and 51 samples, where each element has amplitudes of 1/11 and 1/51 respectively. Display the characteristics of the new filters using the `freqz()` function and compare them to the characteristics of a filter with a length of 21 samples. Describe how the length of the window affects its frequency response (half-power method -3dB).

The report should include the Matlab code used to perform the above tasks. A graph showing the comparison of time signals from different types of filtration, the original time signal and the filter characteristics from Task 1.1 should also be included. Comment on the compatibility of the characteristics with the filtration result.

From Task 1.2, you should include a graph comparing the operation of the `filter()`, `filtfilt()` functions and the effect of your own method of zero-phase filtration. Describe the characteristics of zero-phase filtration.

From Task 1.3, include a graph comparing the characteristics of filters with windows of 11, 21 and 51 samples and describe how the window length affects the frequency response.

2. Designing IIR i FIR filters

In this problem, you need to filter the y signal generated by the code given below. The signal consists of four sine waves with different instantaneous frequencies that are separated in time. To generate the signal, use the following code:

```
% sampling frequency
fs = 4000;
% time duration
tmax = 1-1/fs;
% time vector
t = 0:1/fs:tmax;
```

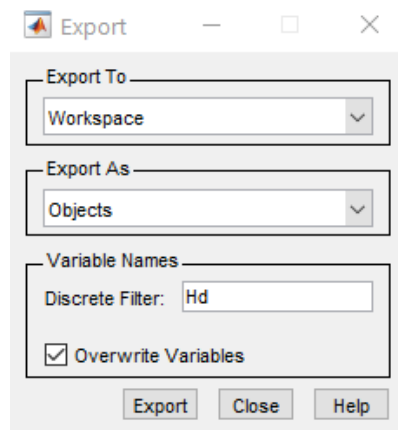
```

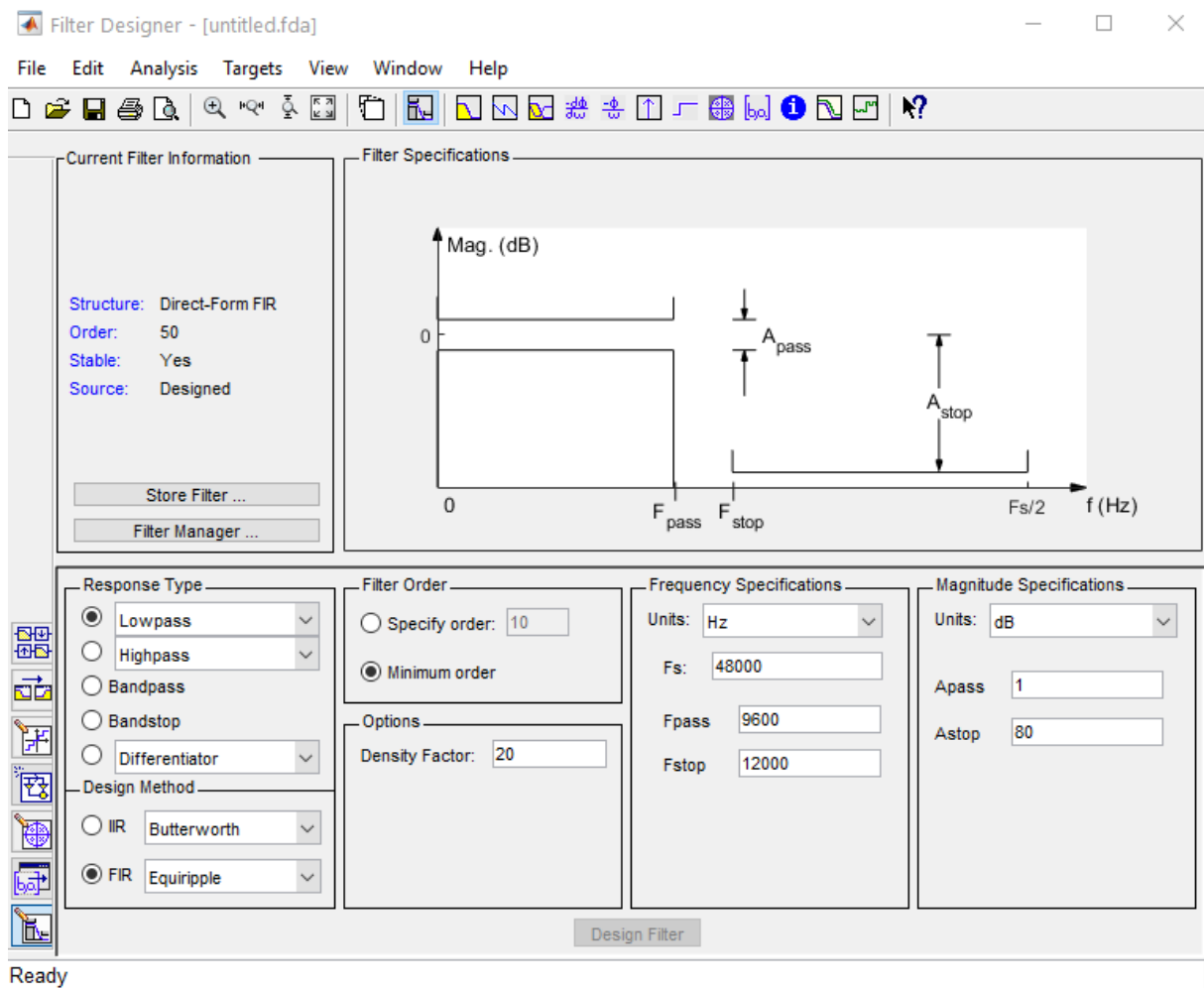
s = length(t);
% frequency vector
f = linspace(0,fs-fs/s,s);

% centra lfrequency of sine waves
w = [200,400,700,850];
% hann window 200 samples
mask = hann(200);
% time delay of the signals
inds = [1400,1000,1200,800];
% allocare memory for y
y = zeros(length(t),1);
% add sine waves
for i = 1:length(w)
    m = zeros(s,1);
    m(inds(i):inds(i)+length(mask)-1) = mask;
    y_temp = sin(2*pi*t*w(i)).*m.';
    y = y+y_temp.';
end

```

On the given signal, it will be necessary to carry out a series of filtrations using FIR and IIR filters. To do this, use the **filterDesigner** filter design tool. After typing **filterDesigner** in the Matlab command window, the window shown below appears. To create a filter according to the given parameters, press the Design Filter button, and to export it, click **File -> Export**, and then select **Export As -> Objects**.





Ex. 2.1

Using **filterDesigner**, design an FIR low-pass filter to filter out the 850 Hz center frequency wavelet and keep the other wavelets in the signal. For this purpose, the **Fpass** and **Fstop** frequencies must be selected accordingly. The FIR filter should be designed using the Kaiser window and the Minimum order method should be used so that the program selects the minimum order of the filter to meet the set parameters. In the Design Method tab, select the **FIR -> Window** option. The sampling frequency should be the same as for the **y** signal. The amplitude drop in the stop band (**Astop**) should be 40 dB, and the amplitude in the pass band (**Apass**) 1 dB. After creating the filter, export the filter to Matlab and apply filtering using the **filter(Hd,x)** function. The effect of the obtained filtration should be described in terms of amplitude change and phase shift. To do this, use the filter characteristics graphs from the **filterDesigner** tool.

Ex 2.2

Using **filterDesigner**, design an IIR low-pass filter with the same parameters **Fpass**, **Fstop**, **Apass** and **Astop** as in the case of the FIR filter. The order of the filter must be selected with the Minimum order setting. In the Design Method setting, we select the Butterworth option.

After generating the filter, export the filter to the workspace in Matlab. Then filter the **y** signal with the **filter()** function. The obtained results should be compared with the results of filtration with the FIR

filter. Discuss the differences between the characteristics of FIR and IIR filters and how they affect the filtration result. For which filter was the lower order obtained?

Finally, zero-phase filtration should be performed using the IIR filter. Compare the results with those obtained using the **filter()** function.

The report should include graphs showing the amplitude-frequency and phase-frequency characteristics of the created FIR and IIR filters, along with a description of their order. Graphs showing the filtration results for both filters should also be included. Describe how the filtration results differ from each other and what causes these differences. Finally, it should be described how zero-phase filtration changes the results of the IIR filter.

3. Filter out noise

The recording named **kotto.wav** was very noisy with noise with very high components in higher frequencies. Design a low-pass filter to remove noise without removing the content of the recording. The sampling frequency in the recording is 44100 Hz. To read the recording into Matlab, use the **audioread** function. Please note that the recording has two channels and both should be filtered, as shown in the example below:

```
% data - data from file
% fs - sampling frequency
[data,fs] = audioread('kotto.wav');

% Num - numerator of FIR
dataF(:,1) = filter(Num,1,squeeze(data(:,1)));
dataF(:,2) = filter(Num,1,squeeze(data(:,2)));
```

Listen to the sound using the following command **sound**:

```
sound(dataF, fs)
```

The report should include the characteristics of the filter used to remove noise from the recording. Also include the name of the actor who is speaking the line you hear.