Lab 6: Frequency response of FIR filters

Grading

This Lab consists of three exercises. Once you have submitted your code in Matlab Grader AND once the deadline has past, your code will be checked for correctness. Note here, that upon submission, your code is already subjected to some basic checks that are aimed to verify whether your code will compile; these basics checks don't say anything about the correctness of your submission. You can visit Matlab Grader again after the deadline (give the servers some time to do all the assessments; this might even take a few days) to see how well you did. In case Matlab Grader indicates you failed an exercise, this does not automatically imply that you failed the entire exercise. Each exercise is subjected to n tests, where the number of tests can vary between exercises. In case Matlab Grader indicates you failed the exercise, this means that not all tests were passed (e.g. in an exercise with 7 tests, you could have passed 6 and Matlab Grader will indicate you failed the exercise). Your grade is calculated based on the number of tests you passed and not on the number of exercises you passed.

1 Introduction

FIR filters can be used for many purposes. These filters are commonly used to manipulate (components of) the input signal. In this lab the effects of a FIR filter on different sinusoidal signals will be shown. The frequency response of a FIR filter is uniquely related to the impulse response. When the input of a FIR filter is a sinusoidal signal with a certain frequency, amplitude and phase, the corresponding output signal is again a sinusoidal signal with the same frequency but with a possibly different amplitude and phase.

2 Frequency response

The goal of this lab is to study the (frequency) response of FIR filters to inputs such as sinusoids. We will use the Matlab function `conv` to implement filters and `freqz` to obtain in an easy way the frequency response plots. As a result you should learn how to characterize a filter by knowing how it reacts to different frequency components in the input.

This Lab also introduces a practical filter: a nulling filter, which can be used to remove sinusoidal interference, e.g. jamming signals in a radar.

2.1 Useful definitions

Frequency response:
Recall from the previous lab that the output $y[n]$ for a given system with impulse response $h[n]$ and input $x[n]$ is given by the following formula, where $h[n]$ is defined by the parameters b_k:

$$y[n] = \sum_{k=0}^{N-1} h[k]x[n-k] = \sum_{k=0}^{N-1} b_k x[n-k] \tag{1}$$

Now let us assume that the input signal $x[n]$ is a complex exponential (phasor) with normalized frequency θ_1: $Ae^{j\phi}e^{jn\theta_1}$ The output of the FIR filter to this input signal can be determined as follows:

$$\begin{aligned}\sum_{k=0}^{N-1} b_k A e^{j\phi} e^{j(n-k)\theta_1} &= \sum_{k=0}^{N-1} b_k A e^{j\phi} e^{jn\theta_1} e^{-jk\theta_1} \\ &= (\sum_{k=0}^{N-1} b_k e^{-jk\theta_1}) \cdot A e^{j\phi} e^{jn\theta_1} \\ &= H(e^{j\theta_1}) \cdot A e^{j\phi} e^{jn\theta_1}.\end{aligned} \tag{2}$$

Here, $H(e^{j\theta_1})$ is the **frequency-response function** of the system as a result of the input signal which consists of one single frequency θ_1.

2.2 Freqz function in Matlab

The Matlab function `freqz` computes the frequency response $H(\mathrm{e}^{\mathrm{j}\theta})$ of a discrete-time LTI system. This frequency response is a series of complex numbers, each with magnitude (amplitude) and angle (phase). Try out the following Matlab code that can be used to compute and plot both the magnitude (absolute value) and phase of the frequency response of a 2-point averaging filter

$$y[n] = \frac{1}{2}(x[n] + x[n-1]) = \frac{1}{2}\sum_{k=0}^{1} x[n-k]$$

as a function of θ in the range $-\pi \le \theta \le \pi$.

```
h = [0.5, 0.5];
ww = -pi:(pi/100):pi;
freqresponse = freqz(h,1,ww );
subplot(2,1,1);
plot(ww,abs(freqresponse))
```

```
subplot(2,1,2);
plot(ww,angle(freqresponse))
xlabel('Normalized radian frequency')
```

Notes:

- For FIR filters the second argument of `freqz(-,1,-)` must always be equal to 1.
- If the output of the `freqz` function is not assigned, then plots are generated automatically. However the magnitude is given in decibels, which is a logarithmic scale. For linear magnitude plots a separate call to `plot` is necessary, as in the example above.
- The frequency vector `ww` should cover an interval of 2π for the variable θ and its spacing must be fine enough to give a smooth curve for $H(\mathrm{e}^{\mathrm{j}\theta})$.

Exercise 1 [4 tests]

Use Euler's formula to show that the frequency response of a 4-point averaging filter

$$y[n] = \frac{1}{4}\sum_{k=0}^{3} x[n-k]$$

is given by:

$$H(\mathrm{e}^{\mathrm{j}\theta}) = \left(\frac{2\cos(0.5\theta) + 2\cos(1.5\theta)}{4}\right) \cdot \mathrm{e}^{-\mathrm{j}1.5\theta} \tag{3}$$

Implement this frequency response directly in MATLAB. Use a vector that includes a total of 2001 samples on the domain $[-\pi, \pi]$. Since the frequency response is a complex valued quantity, use `abs` and `angle` to extract the magnitude and phase of the frequency response for plotting. Make two subplots underneath each other on the specified domain. The first subplot should contain the magnitude of the frequency response and the second subplot should contain the angle of the frequency response. Also add labels **exactly** like the ones in figure 1. If done correctly your plot should look identical to this figure.

2.3 MATLAB find()

Often signal processing functions are performed in order to extract information that can be used to make a decision. The decision process inevitably requires logical tests, which might be done with `if - then` constructs in Matlab. However, Matlab permits vectorization of such tests and the `find` function is one way to do lots of tests at once. In the following example `find` extracts all the numbers that 'round' to 3:

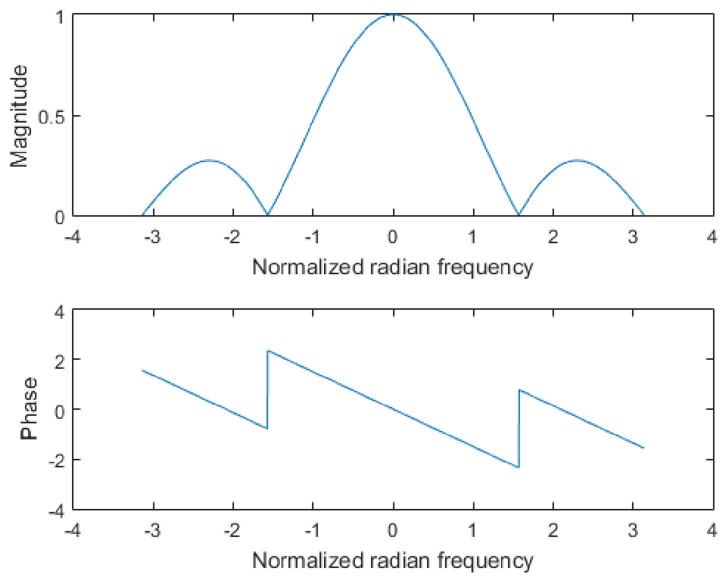

Figure 1: Frequency Response of a 4-point averaging filter

```
xx = 1.4:0.33:5;
k = find(round(xx)==3);
xx(k)
```

The argument of the `find` function can be any logical expression. Notice that `find` returns a list of indices where the logical condition is true.

Exercise 2 [5 tests]

Now suppose that we have a frequency response of a 4-point averaging filter as given in the previous exercise.

Use the `find` command to determine the values for θ where $H(e^{j\theta})$ is zero and save these normalized frequencies in the array `nulling` in the order from lowest to highest. Compare your answer to the frequency response that you have plotted for the 4-point averaging function.

Note: Since there might be round-off errors in calculating $H(e^{j\theta})$, the logical test should probably be a test for those indices where the magnitude (absolute value) of $H(e^{j\theta})$ is smaller than some rather small number. Use 1×10^{-6} for this purpose.

3 Nulling filters

In this section you are going to study the effects of a nulling filter to extract information from a mixture of sinusoidal signals. Nulling filters are filters that completely eliminate frequency components. If the frequency is $\theta = 0$ or $\theta = \pi$ then a 2-point FIR filter can do the nulling. The simplest possible general nulling filter can have as few as 3 coefficients. If θ_{nul} is the desired nulling frequency, then the following length-3 FIR filter

$$y[n] = x[n] - 2\cos(\theta_{nul})x[n-1] + x[n-2] \tag{4}$$

will have a zero in its frequency response at $\theta = \theta_{nul}$. For example, a filter designed to completely eliminate signals of the form $Ae^{j0.5\pi n}$ would have the following coefficients

$$b_0 = h[0] = 1 \ ; \ b_1 = h[1] = -2\cos(0.5\pi) = 0 \ ; \ b_2 = h[2] = 1$$

because we would pick the desired nulling frequency to be at $\theta_{nul} = 0.5\pi$.

Exercise 3 [5 tests]

Generate an input signal $x_1[n]$ that is the sum of two sinusoids:

$$\begin{aligned} x_1[n] &= x_2[n] + 22\cos(0.44\pi n - \pi/3) \\ x_2[n] &= 5\cos(0.13\pi n) \end{aligned} \tag{5}$$

Make the range of the input signal $x_1[n]$ 150 samples long over the range $0 \leq n \leq 149$ and plot in Matlab. Design a 3-point nulling filter that will eliminate the following input frequencies: $\theta = 0.44\pi$. For this part derive the filter coefficients (`filterco`) of the nulling filter and implement the filter in MATLAB. Use `conv` to filter the signal $x_1[n]$ by the designed filter. The output signal is $y[n]$.
Make 3 subplots underneath each other in the following order (from upper to lower graph) with the domain: $x_1[n]$ on [0,149], $x_2[n]$ on [0,149] and y[n] on [0, 151].