

T-61.3010 Digital Signal Processing and Filtering

(v. 1.0, 13.3.2009), Matlab #4 (23.-27.3.2009)

Registration in WebOodi. Bring your own **headphones** if you have. The assistant will guide you through the exercises, but you may go on your own speed. Feel free to ask the assistant, if you have troubles. You can also consult http://www.cis.hut.fi/Opinnot/T-61.3010/how_to_start_with_matlab.shtml or [kuinka_aloitan_matlabin.shtml](http://www.cis.hut.fi/Opinnot/T-61.3010/kuinka_aloitan_matlabin.shtml).

Getting started: In **Windows** just click **Programs - Matlab**. Write down the code into separate files in your working directory (e.g. Z:\DSP\) for future use. Set the "Current Directory" in Matlab to point to the working directory (or type `cd <workdir>`).

The problems marked with [Pxx] are from the course exercise material (Spring 2009).

In the end of this session **you should know**: (a) how to define specifications both for IIR and FIR filters in Matlab, (b) how to design, check and apply digital filters in Matlab.

- [M4006] **Task**: Look at the (IIR) filter specifications for each filter in Figure 1, read the values for the following quantities and convert them to values that Matlab requires (e.g. `help buttord`). See also Table 1 for conversion. Draw the specifications using `speksit(Wp, Ws, Rp, Rs, type, fT)`.

- filter type: lowpass / highpass 'high' / bandpass / bandstop 'stop'
- sampling frequency $f_T =$ Hz, in Matlab it corresponds to 2.
- passband cut-off freq. $f_p =$ Hz, in Matlab $W_p = [$]
- stopband cut-off freq. $f_s =$ Hz, in Matlab $W_s = [$]
- maximum passband attenuation (maximum ripple) in decibels $\alpha_{max} =$ dB, in Matlab $R_p = [$]
- minimum stopband attenuation in decibels $\alpha_s =$ dB, in Matlab $R_s = [$]

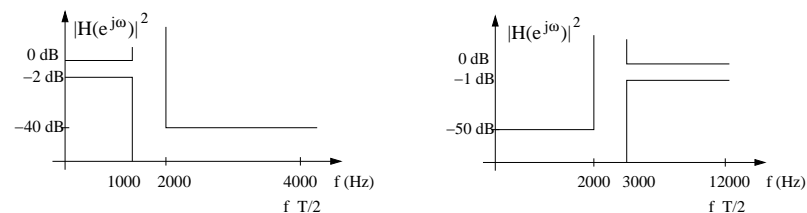


Figure 1: Filter specifications of two IIR filters of Problem 1. Here f_T is the sampling frequency.

- [M4002] Basics of filter design are introduced in (*Mitra 2Ed Sec. 7.1 / 3Ed Sec. 9.1, 10.1*). Here is a short list:

- Determine the specifications of the filter. You may need to use it, for example, as a lowpass filter. Draw the specifications on paper. You may also have to analyse the input signal in order to know what is the sampling frequency of the signal and what are the cut-off frequencies needed.

- Choose the type of the filter. Is it IIR or FIR? Is it linear-phase (FIR) or not? Choose from IIR filters: Butterworth, Chebyshev I, Chebyshev II, Elliptic. Choose from FIR filters: window method, frequency method, least-square, equiripple, etc.
- Estimate the order of the filter. Filter order should be big enough to fulfill the specifications and it should be low in order to be avoid computation. Some Matlab functions: `buttord`, `cheblord`, `cheb2ord`, `ellipord`, `firpmord`, etc. You may also have to look at formulas in order to get the correct order for a fixed window FIR filter, `fir1`, (*Mitra 2Ed Table 7.2 / 3Ed Table 10.2*).
- Compute the coefficients of the transfer function. This means in Matlab finding the vectors B and A for $H(z) = B(z)/A(z)$. Some Matlab functions: `butter`, `cheby1`, `cheby2`, `ellip`, `fir1`, `fir2`, `firls`, `firpm`, etc. There are also lots of window functions: `boxcar`, `hamming`, `hann`, `blackman`, `kaiser`.
- Check that the specifications are fulfilled! For instance, use `freqz`. You can draw the specifications in the same figure with the function `speksit.m` (IIR) or `speksitFIR.m`, which can be found from the course web page.
When plotting the spectra of signals or the frequency responses of filters, zoom the x-axis to be from 0 to $f_T/2$. You can use `axis([0 fT/2 minx maxy])`; or set the axis properties directly `set(gca, 'XLim', [0 fT/2])`;
- Apply your filter to the input signal. For instance, use `filter`.

Task: Read in signal M4002.wav and verify that its sampling frequency f_T is 8 kHz. Design an elliptic IIR lowpass filter with $f_p = 1000$ Hz (end of passband frequency) and $f_s = 2000$ Hz (start of stopband frequency). Passband ripple may be 2 dB, and minimum stopband attenuation at least 40 dB.

Go on all phases of the list above. Read `help ellipord | ellip`.

Write down the order of the filter and the coefficients of the transfer function. Plot also the pole-zero diagram, where the filter order is $\max\{\#\text{zeros}, \#\text{poles}\}$.

In the end, filter a demo signal and listen to the result.

```
% Read and analyze the signal
[x, fT] = wavread('M4002.wav');
figure(1); clf;
plot(x); % NOTE! Index values at x-axis, not seconds
% See Matlab 2 / Problem 3 for seconds

figure(2); clf;
spectrogram(x, 512, 256, 512, fT, 'yaxis');
```

% (i) Determine specifications

Sampling frequency f_T (f_s ?)	0	$f_T/4$	$f_T/2$	$3f_T/4$	f_T	Hz
Frequency f , $f_T = 20$ kHz	0	5000	10000	15000	20000	Hz
Angular frequency Ω	0	10000π	20000π	30000π	40000π	rad/s
Normalized angular frequency ω	0	$\pi/2$	π	$3\pi/2$	2π	rad/sample
Normalized frequency f_{norm}	0	0.5	1	1.5	2	1

Table 1: Examples on frequency normalization, with the sampling frequency $f_T = 20$ kHz. Matlab uses f_{norm} . Angular frequency Ω is used with analog signals, ω with digital sequences.

```

fT                % sampling frequency of signal & system
Wp = ; Ws = ; Rp = ; Rs = ; % get correct values!
figure(3); clf;
speksit(Wp, Ws, Rp, Rs, '', fT); % draw the specs!
grid on; xlabel('Hz'); ylabel('dB'); title('|H(e^{-j \omega})|^2');

%% (ii), (iii) Estimate filter order, help ellipord
[N, Wn] = ellipord( ); % add correct variables

%% (iv) Compute filter coefficients, help ellip
[B, A] = ellip( ); % add correct variables
max([length(B) length(A)])-1 % order of the filter
figure(4); clf;
zplane(B, A); % pole-zero plot

%% (v) Check if specifications are fulfilled
[H, w] = freqz(B, A, 512, fT);
magdB = 20*log10(abs(H)/max(abs(H))); % squared magnitude in dB
figure(3); hold on; % draw into the same figure No. 1
plot(w, magdB); % amplitude response in decibels

%% (vi) Use the filter and test it
y = filter(B, A, x); % apply X -> H(z) -> Y

%% Analyze the input and output signal
figure(1); hold on;
py = plot(y);
set(py, 'Color', [1 0 0], 'LineStyle', '-.', 'LineWidth', 2);
%
figure(5), clf;
spectrogram(y, 512, 256, 512, fT, 'yaxis');
%
xF = 20*log10(abs(fft(x)));
yF = 20*log10(abs(fft(y)));
figure(6), clf;
plot(xF); hold on; % NOTE! Index values at x-axis, not Hertz
plot(yF, 'r'); title('Spectra'); % See Matlab 2 / Problem 3 for Hertz

%% ... listen to the signals
soundsc(x, fT); pause; % original; press any key to continue
%
soundsc(y, fT); % filtered

```

3. [M4005] FIR filters are often designed with linear scale in y-axis. The passband ripple varies between $1 - \delta_p \dots 1 + \delta_p$. The stopband ripple is designed to be less than δ_s .

Parks-McClellan algorithm provides an optimal equiripple FIR filter. The estimation of the order is computed with `firpmord` and then the coefficients of the filter with `firpm`.

Task: Draw specifications for a FIR highpass filter: sampling frequency $f_T = 22050$ Hz, stopband ends at $f_p = 3300$ Hz, passband starts from $f_s = 4000$ Hz, the ripples are $\delta_p = 0.01$ and $\delta_s = 0.1$.

Compute $H(z)$. Are the specifications fulfilled? Read `help firpmord`. What is the filter order? Test also what is the filter order if you decrease width of transition band (band between f_s and f_p) from 700 Hz to 300 Hz.

Filter the demo signal.

What does variation $1 - \delta_p \dots 1 + \delta_p$ correspond in IIR specifications where the maximum is scaled to 0 decibels (1 in linear scale) and α_{max} is given in power decibels $|H(e^{j\omega})|^2$?

```

%% Specifications
fT = 22050;
F = [ 3300 4000 ];
A = [0 1];
DEV = [0.1 0.01];
figure(1); clf;
speksitFIR(F, A, DEV, fT);

%% Filter order
[ fill here ] = firpmord( fill here );

%% Filter coefficients
% Transfer function H(z) = B(z)/A(z)
b = firpm( fill here );
a = 1; % FIR: no feedback <=> A(z) == 1

%% Compute and plot the amplitude response
[H, F] = freqz(b, a, 512, fT);
figure(1); hold on;
plot(F, abs(H));
grid on;
xlabel('frequency (Hz)'); ylabel('|H(e^{-j \omega})|');

%% Plot the phase response
figure(2); clf;
plot(F, unwrap(angle(H)));

%% Group-delay of the filter
% -(d/dw) <H(e^{jw})
figure(3); clf;
grpdelay(b, a, 512, fT);

%% Pole-zero plot
figure(4); clf;
zplane(b, a)

%% Filter kiisseli.wav
[x, dummy] = wavread('kiisseli.wav');
y = filter(b, a, x);

```