T-61.3010 Digital Signal Processing and Filtering

(v. 1.0, 5.2.2009), Matlab #3 (18.2., 23.2., 24.2.2009)

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`.

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) what is a matched filter used for (with respect to convolution), (b) how to produce "echo" effect with convolution, (c) how to express a LTI filter in Matlab using vectors `B` and `A`, (d) how to use basic functions (`freqz`, `zplane`, `impz`, `filter`) in Signal Processing Toolbox.

1. [**M2038**] Linear convolution can be computed using `conv` in Matlab.

   Task: Run the example below. If considering $h[n] = \delta[n] - 2\delta[n-1] + \delta[n-2]$ as a matched filter, what kind of "signal parts" does it detect?

```
x  = [1 0 2 1 9 8 9 9 8 9 0 1 0 1]   % input sequence: [low high low]
h  = [1  -2  1]                      % impulse response h[n]
y  = conv(h, x)                      % output sequence y = h * x
figure(1); clf;
subplot(2,1,1);  stem(x);
subplot(2,1,2);  stem(y);
figure(2); clf;  freqz(h, 1);        % is filter h lowpass/highpass/...?
```

   Task: Now consider a case of synthesizing signal effects. Read `kiisseli.wav` into the vector `x` with `wavread`. Create reverbaration or "echo" effect by convolving `kiisseli` with impulse response $h[n]$ depicted in Figure 1. Listen with `soundsc(y, fs)`. Hint: one way to write $h[n]$: `h = [1 zeros(1,N) 0.8 zeros(1,N) 0.7 ...  0.1];`, see `help zeros`.

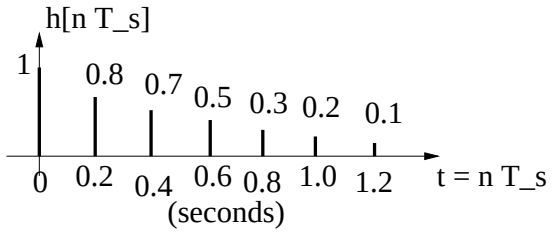

Figure 1: Problem 1: Impulse response $h[n]$ for echo. For Matlab, insert a correct number of zeros (corresponding 0.2 seconds) between each non-zero element of $h[n]$.

2. [**M2039**] Recall that in the first Matlab session a complex-valued function $H(\omega) = 2 - e^{-j\omega}$ was computed and visualized in many ways. It was mentioned that frequency responses (frequency-domain) of all LTI filters resemble $H(\omega)$.

   Digital (causal) LTI systems have transfer functions of type

$$\begin{aligned} H(z) &= b_0 + b_1 z^{-1} + \ldots + b_M z^{-M} \\ H(z) &= \frac{b_0 + b_1 z^{-1} + \ldots + b_M z^{-M}}{a_0 + a_1 z^{-1} + \ldots + a_N z^{-N}} \end{aligned}$$

   where the first is a FIR (finite impulse response) and the latter IIR (infinite impulse response). The frequency response is received by substitution $z = e^{j\omega}$. It can be written in Matlab using the coefficients of the numerator and denominator polynomials:

$$H(e^{j\omega}) = \frac{b_0 + b_1 e^{-j\omega} + \ldots + b_M e^{-jM\omega}}{a_0 + a_1 e^{-j\omega} + \ldots + a_N e^{-jN\omega}}$$

```
B = [B(1) B(2) ... B(M+1)];   % or ''num'' = numerator polynomial
A = [A(1) A(2) ... A(N+1)];   % or ''den'' = denominator polynomial
```

   For example, in [**P55**] we have a second-order LTI system with feedback as shown in Figure 2.

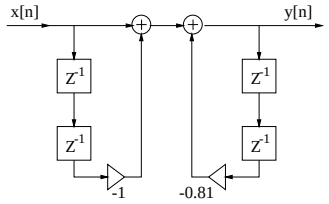

Figure 2: Problem 2: LTI system of Problem [**P55**].

   The frequency response is received from the difference equation:

$$\begin{aligned} y[n] &= x[n] - x[n-2] - 0.81y[n-2] \\ y[n] + 0.81y[n-2] &= x[n] - x[n-2] && \text{| z-transform} \\ Y(z) + 0.81z^{-2}Y(z) &= X(z) - z^{-2}X(z) \\ Y(z)(1 + 0.81z^{-2}) &= X(z)(1 - z^{-2}) && \text{| : } X(z) \\ Y(z)/X(z)(1 + 0.81z^{-2}) &= (1 - z^{-2}) && \text{| : } (1 + 0.81z^{-2}) \\ H(z) = Y(z)/X(z) &= \frac{1 - z^{-2}}{1 + 0.81z^{-2}} && \text{| } z \leftarrow e^{j\omega} \\ H(e^{j\omega}) &= \frac{1 - e^{-2j\omega}}{1 + 0.81e^{-2j\omega}} \end{aligned}$$

   The numerator polynomial is $B(e^{j\omega}) = 1 + 0 \cdot e^{-j\omega} - e^{-2j\omega}$. The coefficients of the polynomial are $B_k = \{\underline{1},\ 0,\ -1\}$ with respect to powers of $e^{-j\omega}$. In the same way, the denominator polynomial is $A(e^{j\omega}) = 1 + 0 \cdot e^{-j\omega} + 0.81e^{-2j\omega}$, and the coefficients are $A_k = \{\underline{1},\ 0,\ 0.81\}$.

   Shortly, analogous to Matlab #1 / Problem 5, we have now a complex-valued function $H(\omega)$, which could be analyzed at points `w = [0 :  pi/512 :  pi];` receiving frequency response `H = (1 - exp(-j*2*w)) ./ (1 + 0.81*exp(-j*2*w));` However, the filter can be analyzed using the following commands from Signal Processing Toolbox:

```
%% Analysis of LTI filter
% H(z) = [1 - z^(-2)] / [1 + 0.81 z^(-2)]  = B(z)/A(z)
% B(z) = [1  + 0 z^(-1)  -1 z^(-2)]     % numerator polynomial of H(z)
B      = [1    0         -1];
% A(z) = [1  + 0 z^(-1)  +0.81 z^(-2)]  % denominator polynomial of H(z)
A      = [1    0          0.81];
tf2latex(B, A)       % download from course web page
%% default freqz-plot
figure(1); clf;
freqz(B, A);         % default amplitude (log) and phase response
%% amplitude response |H(e^jw)|
figure(2); clf;
[H, w] = freqz(B, A);
plot(w, abs(H));    % linear y-axis; OR in decibels (w, 20*log10(abs(H)))
grid on; xlabel('angular frequency \omega (rad)');
title('Amplitude response');
%% phase response <H(e^jw)
figure(3); clf;
plot(w, angle(H));   % phase response
grid on; xlabel('angular frequency \omega (rad)');
title('Phase response');
%% pole-zero plot
figure(4); clf;
zplane(B, A);        % pole-zero plot
roots(B)             % roots of polynomial B(z) == ``zeros''
roots(A)             % roots of polynomial A(z) == ``poles''
[Z,P,K] = tf2zp(B, A) % zeros, poles and gain.
%% impulse response h[n]
figure(5); clf;
[h, n] = impz(B, A); % impulse response
stem(n, h);
grid on; xlabel('index n');
title('Impulse response h[n]');
imp2latex(h, n(1:10))   % download from course web page,
                        % print first 10 values of h[n]
```

Task: Consider the following LTI systems and classify if they are lowpass / highpass / bandpass / bandstop / allpass / comb / notch filter? What is the order of each filter? Are they FIR or IIR?

a) $H_1(e^{j\omega}) = \frac{1}{1+0.2e^{-j\omega}+1.2e^{-2j\omega}}$

b) $H_2(z) = \frac{0.07+0.1z^{-1}+0.1z^{-2}+0.07z^{-3}}{1-1.4z^{-1}+1.1z^{-2}-0.3z^{-3}}$

c) $H_3(e^{j\omega}) = 1 + e^{-4j\omega}$

d) Poles are at $p = 0.5 \pm 0.5j$ and zeros at $z = e^{\pm j0.25\pi}$. Use `zp2tf` to find the transfer function $H(z)$.

3. [**M2040**] Earlier `conv` was used to compute convolution. However, filtering can be efficiently executed using the command `filter`: `y = filter(B, A, x)`. In the time-domain the filtering is convolution (`conv`) of the input and the impulse response - in the frequency-domain it is product of the input spectrum and frequency response.

$$y[n] = h[n] \circledast x[n] \qquad \leftrightarrow \qquad Y(e^{j\omega}) = H(e^{j\omega}) \cdot X(e^{j\omega})$$

Task: Consider an elliptic 3rd order filter $H(e^{j\omega})$

$$H(e^{j\omega}) = K \cdot \frac{1 - 2.26e^{-j\omega} + 2.26e^{-2j\omega} - e^{-3j\omega}}{1 - 0.6e^{-j\omega} + 0.72e^{-2j\omega} + 0.1e^{-3j\omega}}$$

Apply the filter for the signal `music.wav` using its sampling frequency. Listen to the original signal and the filtered signal.

Analyze the filter $H(e^{j\omega})$ using the tools in Problem **??**. Compute the amplitude response with frequency (Hertz) in x-axis: `[H, F] = freqz(B, A, NDFT2, fs)`, where `NDFT2` is the number of points where DFT is evaluated (default 512) and `fs` is the sampling frequency of the signal (default 2 corresponding 2π). Draw the amplitude response in decibel scale and Hertz in x-axis. What is the scaling factor K, so that $\max\{|K \cdot H(e^{j\omega})|\} = 1$ or 0 decibels, try `max(abs(H))`.