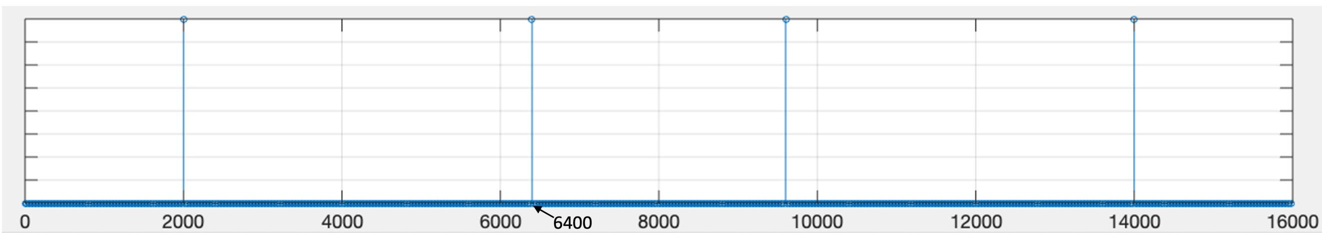
Multimedia Signal Processing 1st Module

MATLAB part [12 pts]

July 19th, 2022

Text:

1) [2 pt] Build a second order all-pass filter A(z) with pole radius 0.9 and phase pi/4.
   - Plot the absolute value and the phase of its DTFT by using 2048 frequency samples.
2) [2 pt] You are given the following plot of a real DFT as a function of frequency [Hz] starting from 0:



   - Define the discrete-time signal x(n) related to this DFT. The amplitude of each signal component is equal to 1. The signal has 800 samples.
3) [5.5 pt + 1 extra-pt] Define the signal y(n) as the signal x(n) filtered with the all-pass filter A(z). Then, define the signal z(n) as the arithmetic mean between the signals x(n) and y(n).
   - Compute the DFTs of the signals x(n), y(n) and z(n) and plot (with the function "stem") their absolute values versus normalized frequencies. Which differences do you notice between the DFTs of x(n) and y(n)? And between those of x(n) and z(n)?
   - Find the filter H(z) such that Z(z) = X(z) * H(z), where Z(z) and X(z) are the Z-transform of the signals z(n) and x(n), respectively. Define the filter coefficients at the numerator and at the denominator.
   - Plot the zeroes and the poles of the filter in the complex plane. Which kind of filter is it?
   - [1 extra-pt] Can you find a motivation why H(z) has this behaviour, considering the way we defined the signal z(n)?
4) [2.5 pt] We want to change the sampling rate (by reducing it) of the signal x(n), being careful to avoid aliasing. Changing the rate of the signal x(n), we want to obtain the signal w(n), whose DFT contains only two peaks located at 4/5*pi and its symmetric (apart from other small signal contributions).
   - Which should be the final sampling rate to obtain the signal w(n)?
   - Define the signal w(n), compute its DFT and check if it fulfils the above requirements. If you need to use a filter, you can choose the filter order.

Solution:

```
close all
clearvars
clc

%% 1. [2 pt]

% Build a second order all-pass filter A(z) with pole radius 0.9 and phase
pi/4.
% Plot the absolute value and the phase of its DTFT by using 2048
frequency samples.

rho = 0.9;
theta = pi/4;
A_ap = [1, -2*cos(theta)*rho, rho^2];
B_ap = fliplr(conj(A_ap));

[H_ap, omega] = freqz(B_ap, A_ap, 2048, 'whole');
figure,
plot(omega./(2*pi), abs(H_ap));
title('|DTFT| of the all-pass filter');
grid;
xlabel('f [norm]');

figure,
plot(omega./(2*pi), angle(H_ap));
title('Phase of the DTFT of the all-pass filter');
grid;
xlabel('f [norm]');

%% 2. [2 pt]

% Define the discrete-time signal x(n) related to this DFT.
% The amplitude of each signal component is equal to 1.
% The signal has 800 samples.

f0 = 2000;
f1 = 6400;
Fs = 16e3;
N = 800;

duration = N/Fs;
time = 0:1/Fs:duration - 1/Fs;

x = cos(2*pi*f0*time) + cos(2*pi*f1*time);
```

```
%% 3. [5.5 pt + 1 extra-pt]

% Define the signal y(n) as the signal x(n) filtered with the all-pass
filter A(z).
% Then, define the signal z(n) as the arithmetic mean between the signals
x(n) and y(n).
% Compute the DFTs of the signals x(n), y(n) and z(n) and plot their
absolute
% values versus normalized frequencies.
% Which differences do you notice between the DFTs of x(n) and y(n)?
% And between those of x(n) and z(n)?
% Find the filter H(z) such that Z(z) = X(z) * H(z), where Z(z) and X(z)
% are the Z-transform of the signals z(n) and x(n), respectively.
% Define the filter coefficients at the numerator and at the denominator.
% Plot the zeroes and the poles of the filter in the complex plane.
% Which kind of filter is it?
% Can you find a motivation why H(z) has this behaviour,
% considering the way we defined the signal z(n)?

y = filter(B_ap, A_ap, x);
z = 1/2*(x + y);

X = fft(x);
Y = fft(y);
Z = fft(z);
freq_axis = 0:1/N:1 - 1/N;

figure;
stem(freq_axis, abs(X));
leg = {};
leg{1} = 'Absolute value of the DFT of the signal x(n)';
hold on;
stem(freq_axis, abs(Y));
leg{2} ='Absolute value of the DFT of the signal y(n)';
stem(freq_axis, abs(Z));
leg{3} ='Absolute value of the DFT of the signal z(n)';
grid;
legend(leg);
xlabel('f [norm]');

% The DFTs of x(n) and y(n) are really similar, since y(n) is the result
of
% an all-pass filtering.
% The DFTs of z(n) is strongly attenuated in correspondence of the lowest
% sinusoid of the signal x(n), while the highest sinusoidal component
% remains basically untouched.

% To find the filter H(z), we should write the relationship between X(z)
% and Z(z) --> Z(z) = (X(z) + X(z)*A(z)) / 2 = X(z) (A(z) + 1)/2
```

```
% --> H(z) = (A(z) + 1)/2.
% By making easy hand-written computations, we find the numerator and
% denominator coefficients.

B = [1+rho^2, -4*rho*cos(theta), 1 + rho^2];
A = A_ap;

figure;
zplane(B, A);
title('zeros and poles of the filter H(z)');

% The filter is a notch. We can understand it by looking at the position
of
% the zeros and poles --> they have the same phase, but the zeros are on
% the unit circle.

% z(n) is the arithmetic mean between the signals x(n) and y(n).
% y(n) is the result of an all-pass filtering --> the phase of y(n) is
% affected by the phase of the all-pass filter, which has a shift of pi
% exactly at the lowest frequency of the signal (due to the maximum phase
zeroes).
% Therefore, when we sum the two signals at this frequency component,
% their absolute values are the same but phases are opposite
% --> the total contribution is 0. It is why the filter has a notch in
this
% position.

%% 4. [2.5 pt]

% We want to change the sampling rate (by reducing it) of the signal x(n),
% being careful to avoid aliasing. Changing the rate of the signal x(n),
% we want to obtain the signal w(n), whose DFT contains only two peaks
% located at 4/5*pi and its symmetric (apart from other small signal
contributions).
% Which should be the final sampling rate to obtain the signal w(n)?
% Define the signal w(n), compute its DFT and check if it fulfils the
% above requirements. If you need to use a filter, you can choose the
filter order.

% If we change the sampling rate, we know that the final peaks of the
% signal x(n) will end in the initial normalized angular frequencies *
M/L.
% --> omega_0 = pi/4 will end in pi/4 * M/L.
% --> omega_1 = 4/5*pi will end in 4/5*pi * M/L.
% Since we are reducing the sampling rate, M > L.
% If we should find a single sinusoidal contribution in w(n), it means
that
% the ratio M/L is so large that we will need to filter out the highest
% sinusoidal component of x(n) to avoid aliasing.
```

```
% --> The unique sinusoidal component that remains is pi/4 * M/L.
% --> to obtain a final peak in 4/5*pi, M should be = 16, L = 5.

L = 5;
M = 16;
Fs_new = Fs * L/M;

% upsampling
x_upsampled = zeros(1, length(x) * L);
x_upsampled(1:L:end) = x;

% filtering
cutoff = min([1/(2*L), 1/(2*M)]);
cutoff_filter = 2*cutoff;
h_multirate = L*fir1(64, cutoff_filter);
x_f = filter(h_multirate, 1, x_upsampled);

% downsampling
w = x_f(1:M:end);

% compute the DFT
W = fft(w);
freq_axis = 0:1/length(W):1 -1/length(W);
figure;
plot(freq_axis, abs(W));
grid;
title('|DFT| of the signal w(n)');
xlabel('f [norm]');

% there are only two peaks in 4/5*pi and its symmetric.
```