MultimediaSignal Processing 1st Module

MATLAB part [12 pts]

November 9th, 2020

Text:

1. [1 pt] Close the opened figures, clear the workspace and clear the command window.
2. [2 pt] You are given a filter H(z) = B(z)/A(z). A(z) = 1 + 0.8z^-1, while B(z) = 1 - z_0*z^-1.
    - Select the value z_0 among these possible values = {-4, 0.25, 0.5, 3exp(j*pi/3)} such that the all-pass transfer function related to H(z) is real and stable.
    - the function 'find_allpass.m' (the code is reported below) receives as input the numerator and denominator of a filter and returns the numerator and the denominator of the all-pass transfer function related to that filter. Use the function 'find_allpass.m' (copy it into your MATLAB code) to compute the numerator and denominator of the all-pass transfer function Hap(z) = b_ap(z)/a_ap(z) related to the filter H(z), defining b_ap(z) and a_ap(z).
    - Compute the zeroes and the poles of Hap(z) and plot them in the complex plane.
3. [2 pt] Define a sinusoidal signal s(n) = cos(2*pi*f0*n), n = [0, N-1], N = 120, such that s(n=0) = s(n=25)
    - plot the signal s(n) versus n with the function 'stem'.
4. [4 pt]
    - Compute the linear convolution between b_ap(n) and one period of s(n), considering only the first samples of the result equal to the value of the period of s(n).
    - Compute also the circular convolution (exploiting the DFT properties) between s(n) and b_ap(n), considering a number of samples equal to the period of s(n).
    - plot the two convolution results in the same figure using the function 'stem'.
    - Is there any difference between the two results? If yes, in which samples? Motivate your answer.
5. [3 pt] Filter the signal s(n) with Hap(z), defining s_f(n).
    - Plot the amplitudes of the DFT of s(n) and of the DFT of s_f(n) in the same figure. As x-axis, consider the positive frequencies, starting from 0. Decide yourself if plotting the frequency axis on Hz or in normalized domain. (Hint: do we have any information about the sampling rate?)
    - Knowing |S(f0)| (the amplitude of the DFT of s(n) evaluated in f = f0), which should be the value of |S_f(f0)|, apart from small deviations? Motivate your answer.

############################ Function code to be copied to MATLAB ##############################

```
function [b_out, a_out] = find_allpass(b,a)

% Input: b, a = numerator and denominator of H(z)
% Output: b_out, a_out = numerator and denominator of the all-pass
transfer function related to H(z)

a_tilde = fliplr(conj(a));
b_tilde = fliplr(conj(b));

b_out = conv(b, a_tilde);
a_out = conv(a, b_tilde);

b_out = b_out / a_out(1);
a_out = a_out / a_out(1);

end
```

Solution:

```
%% 1.

% [1 pt] Close the opened figures, clear the workspace and clear the
command window

close all
clearvars
clc

%% 2.

% [2 pt] You are given a filter H(z) = B(z) / A(z). A(z) = 1 + 0.8z^-1,
% while B(z) = 1 - z_0 z^-1.
% Select the value z_0 among these possible values: {-4, 0.25, 0.5,
% 3exp(j*pi/3)} such that the all-pass transfer function related to H(z)
is
% real and stable.
% the function find_allpass.m receives as inputs the numerator and
% denominator of a filter and returns the numerator and the denominator of
% the all-pass transfer function related to that filter.
% use the function find_allpass.m (copy it into your code) to compute the
% numerator and denominator of the all-pass transfer function Hap(z)
related to
% the filter H(z), defining b_ap and a_ap
% Compute the zeroes and the poles and plot them in the complex plane

a = [1, 0.8];
% choose a real value of z_0 such that its conjugate reciprocal
% (that will become a pole in Hap(z)) is inside the unit circle.
z_0 = -4;
b = [1, -z_0];

[b_ap, a_ap] = find_allpass(b, a);

zeroes = roots(b_ap);
poles = roots(a_ap);
zplane(b_ap, a_ap);

%% 3.

% [2 pt] Define a sinusoidal signal s(n) = cos(2*pi*f0*n), n = [0, N-1]
% (N = 120), such that s(n=0) = s(n=25)
% plot the signal versus n with the function 'stem'

period = 25;
```

```
f0 = 1/period;
N = 120;
n = 0:N-1;
s = cos(2*pi*f0*n);

figure;
stem(n, s);

%% 4.

% [4 pt] Compute the linear convolution between b_ap(n) and one period of
% the signal s, considering only the first samples of the result equal
% to the value of the period of s(n). Then, compute also the circular
convolution
% (exploiting the fft properties)
% between s(n) and b_ap(n), considering a number of samples equal to the
% period of s(n).

% plot the two results in the same figure using the function 'stem'.
% Is there any difference between the two results? If yes, in which
% samples? Motivate your answer.

linear_conv = conv(s(1:period), b_ap);
linear_conv = linear_conv(1:period);

S = fft(s(1:period));
B_ap = fft(b_ap, period);

cyclic_conv = ifft(S.*B_ap);

figure, stem(linear_conv), hold on, stem(cyclic_conv)
% The two results differ only in the first 2 samples, which correspond to
% the length of the filter b_ap(n) - 1. This is due to periodic artifacts
% of the cyclic convolution between the two signals. This
% consideration is the rationale behind the overlap and save method.

%% 5.

% [3 pt] Filter the signal s(n) with Hap(z), defining s_f(n).
% Plot the amplitudes of the DFT of s(n) (S(f) and of the DFT of s_f(n)
% (S_f(f)) in the same figure.
% As x-axis, consider the positive frequencies, starting from 0.
% Decide yourself if plotting the frequency axis on Hz or in normalized
domain.
% (hint: do we have any information about the sampling rate?)
% Knowing |S(f0)| (the amplitude of the DFT of s(n) evaluated in f = f0),
% which should be |S_f(f0)|, apart from small deviation? Motivate your
answer.
```

```
s_f = filter(b_ap, a_ap, s);

S = fft(s);
S_f = fft(s_f);

freq_axis = 0:1/N:1 - 1/N;
figure; plot(freq_axis, abs(S));
hold on; plot(freq_axis, abs(S_f));

% the two amplitudes should be one equal to the other (apart from small
deviations),
% since the signal s(n)
% has been filtered with an all-pass filter which does not modify the
% amplitude response of the filtered signal.

%% function code

function [b_out, a_out] = find_allpass(b,a)

% Input: b, a = numerator and denominator of H(z)
% Output: b_out, a_out = numerator and denominator of the allpass transfer
function related to H(z)

a_tilde = fliplr(conj(a));
b_tilde = fliplr(conj(b));

b_out = conv(b, a_tilde);
a_out = conv(a, b_tilde);

b_out = b_out / a_out(1);
a_out = a_out / a_out(1);

end
```