Multimedia Signal Processing 1st Module and Fundamentals of Multimedia Signal Processing

date: July 20th, 2023

Ex.1 (Pt.11)

The DFT $X[k]$ of a digital signal $x[n]$ is made of 32 values, from $X[1]$ to $X[32]$. All of them are zero with the exception of $X[1]=32$, $X[3]=16j$ and $X[31]=-16j$.

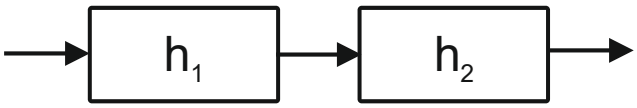
[2 pts] Describe the procedure to retrieve the signal $x[n]$ from its DFT $X[k]$.

[4 pts] Find the signal $x[n]$.

[5 pts] Downsampling the signal of an order of 2 (M=2) what will be the output in the time domain $x'[n]$ and in the frequency domain $X'[k]$?

Ex.2 (Pt.11)

A FIR filter $h_{12}[n]$ is the sequence of two filters:



$h_1[n]=\delta[n]-\delta[n-2]$ and $h_2[n]=\delta[n]-\sqrt{2}\delta[n-1]+\delta[n-2]$.

[3 pts] Describe how to obtain the expression of $h_{12}[n]$ in the time domain and provide its z transform.

[2 pts] Are $h_1[n]$ and/or $h_2[n]$ and/or $h_{12}[n]$ linear filters? Justify your answer.

[3 pts] Provide an approximate representation of the amplitude response of the filter.

An input signal $x(t)=3+\cos(2\pi 4000t)+3\cdot\sin(2\pi 8000t)$ is sampled at 16kHz and filtered with $h_{12}[n]$.

[3 pts] What components will be present at the output? With which amplitude?

CONTINUES ON THE BACK

Ex.3 (Pt.12) To be solved writing the MATLAB code on the sheet.

1) [1.5 pt] The signal x(t) is a cosine with frequency 2.5 KHz, with period 8 samples and defined over 1000 samples. Define the signal.
2) [1.5 pt] The signal x(n) is summed to a periodic sequence y(n) = [2, 0, -2, 0, 2, 0, -2, 0, …]. Define the output signal as z(n), which has the same number of samples of x.
3) [4.5 pt] Design three different filters (with real coefficients, causal and stable):
   - H1(z), which should attenuate the y component from z(n);
   - H2(z), which should completely remove the y component from z(n), without altering the rest of the signal;
   - H3(z), which should keep the magnitude of all signal components.

   The three filters should share the same angular position of their zeroes in the z-plane. If possible, use minimum phase filters. The gain of each filter at frequency = 0 should be fixed at 1.
4) [2.5 pt] By using the function 'freqz' over 2048 samples, plot the absolute value of the frequency response of each filter over the entire frequency spectrum in normalized domain.
   - For each plot, comment precisely on what you expect to see.
   - For each filter, filter the signal z, defining z_1, z_2, z_3 signals.
5) [1 extra pt] If you have available only signals z and z_3, is there a way to find an equivalent version of the signal z_2 which does not imply using the filter H2(z)? If yes, combine the two signals (z and z_3) accordingly and define the signal z_4.

Solutions

Ex.1

1st part: see the lecture related to the inverse Discrete Fourier Transform

2nd part:

$$x[n]=\frac{1}{32}\left(32+16j\cdot e^{-j2\pi\frac{2}{32}n}-16je^{-j2\pi\frac{31}{32}n}\right)=\frac{1}{32}\left(32+32\frac{-e^{-j2\pi\frac{2}{32}n}+e^{-j2\pi\frac{31}{32}n}}{2j}\right)=1-\sin\left(2\pi\frac{1}{16}n\right)$$

3rd part:

$$x'[n]=1-\sin\left(2\pi\frac{1}{8}n\right)$$

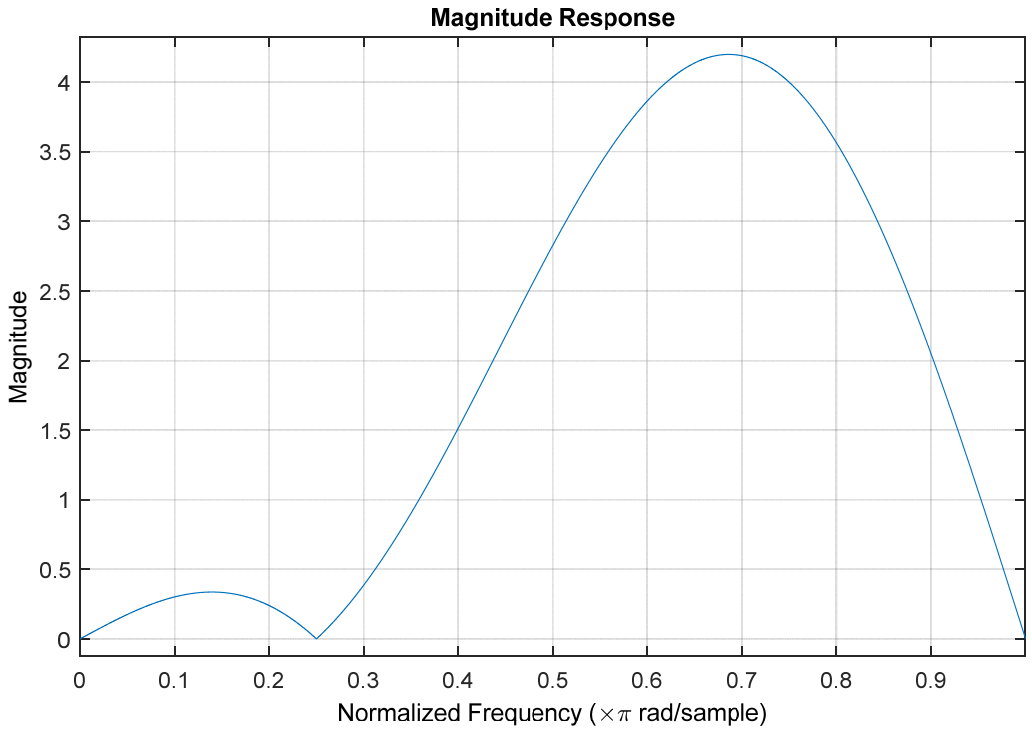
$X'[k]$ will be zero for all the values with the exception of $X'[1]=8$, $X'[3]=8j$ and $X'[1]=-8j$, i.e., $X'[k]=16\cdot\delta[1]-8j\cdot\delta[3]+8j\cdot\delta[15]$.

Ex.2

$$h_{12}[n]=h_1[n]*h_2[n]=\{1,-\sqrt{2},0,\sqrt{2},-1\}\rightarrow H(z)=1-\sqrt{2}z^{-1}+\sqrt{2}z^{-3}-z^{-4}$$

Just $h_2[n]$ is a linear phase filter due to its even symmetry.

The amplitude response of $h_{12}[n]$ is:



The sampled signal is $x[n] = 3 + \cos\left(2\pi \frac{4 \cdot 10^3}{16 \cdot 10^3} n\right) + 3 \cdot \sin\left(2\pi \frac{8 \cdot 10^3}{16 \cdot 10^3} n\right) = 3 + \cos\left(\frac{\pi}{2} n\right) + 3 \cdot \sin(\pi n)$ and the filter will remove the continuous frequency and the sinusoid at the Nyquist frequency.

The amplitude of the filter at $\frac{\pi}{2}$ will be $\left|H\left(z = e^{j\pi/2} = j\right)\right| = \left|1 + \sqrt{2}j + \sqrt{2}j - 1\right| = 2\sqrt{2}$ so that the amplitude of the input cosinusoid will be amplified by that value.

Ex.3 (MATLAB CODE)

```
close all
clearvars
clc

%% 1 [1.5 pt]

% The signal x(t) is a cosine with frequency 2.5 KHz, with period 8
samples
% and defined over 1000 samples. Define the signal.

N = 1000;
f1 = 2.5e3;
p1_samples = 8;

% we have to find Fs. To do so:
p1 = 1/f1; % period in seconds
Fs = p1_samples/p1;

time_axis = 0:1/Fs:(N-1)/Fs;

% define the signal
x = cos(2*pi*f1*time_axis);

%% 2 [1.5 pt]

% The signal x(n) is summed to a periodic sequence y(n) = [2, 0, -2, 0, 2,
0, -2, 0, …].
% Define the output signal as z(n), which has the same number of samples
of x.

% Notice that the sequence is a sinusoidal sequence with period = 4.
f2_n = 1/4;
y = 2 * cos(2*pi*f2_n*(0:N-1));
z = x + y;

%% 3. [4.5 pt]

% Design three different filters (with real coefficients, causal and
stable):
% H1(z), which should attenuate the y component from z(n);
% H2(z), which should completely remove the y component from z(n),
% without altering the rest of the signal;
% H3(z), which should keep the magnitude of all signal components.
% The three filters should share the same angular position of their
% zeroes in the z-plane. If possible, use minimum phase filters.
% The gain of each filter at frequency = 0 should be fixed at 1.
```

```
% Filter H1(z) should attenuate the y component
% --> let's consider two zeroes at phase = 2*pi*1/4 = pi/2

rho_z = 0.95; % for the minimum phase
B_1 = [1, 0, rho_z^2];
A_1 = 1;
% the gain of the filter at frequency = 0 should be fixed at 1
B_1 = B_1 / sum(B_1(:));

% Filter H2(z) should be a notch
% --> two zeros at pi/2 with abs value = 1
% --> two poles at pi/2 with abs value < 1

rho_z = 1;
rho_p = 0.95;
B_2 = [1, 0, rho_z^2];
A_2 = [1, 0, rho_p^2];
% the gain of the filter at frequency = 0 should be fixed at 1
B_2 = B_2 * sum(A_2(:)) / sum(B_2(:));

% Filter H3(z) should be an all-pass
% --> two zeros at pi/2 with abs value > 1 (for building a causal stable
% filter)
% --> two poles at pi/2 with abs value = 1/(abs value of the zeroes)
% We can build the numerator and then find the denominator such that
% denominator coefficients = conj(fliplr(numerator coefficients))

rho_z = 1.2;
B_3 = [1, 0, rho_z^2];
A_3 = [rho_z^2, 0, 1];

%% 4. [2.5 pt]

% By using the function ,freqz over 2048 samples, plot the absolute value
% of the frequency response of each filter over the entire frequency
% spectrum in normalized domain.
% For each plot, comment precisely on what you expect to see.
% For each filter, filter the signal z, defining z_1, z_2, z_3 signals.

% H1 filter behaviour
[H, omega] = freqz(B_1, A_1, 2048, 'whole');
figure,
plot(omega./(2*pi), abs(H));
title('|DTFT| of the filter H_1(f)');
grid;

% The gain at f = 0 should be 1. We expect a bell shape with minimum in f
=
% 1/4 until f = 1/2. Then, everything after f = 1/2 is symmetric.

% filter behaviour
[H, omega] = freqz(B_2, A_2, 2048, 'whole');
figure,
plot(omega./(2*pi), abs(H));
title('|DTFT| of the filter H_2(f)');
grid;
```

```
% The gain at f = 0 should be 1. We expect more or less a flat behaviour
% for all frequencies, except for f = 1/4 in which there is a strong
% localized attenuation. Everything is symmetric after f = 1/2.

% filter behaviour
[H, omega] = freqz(B_3, A_3, 2048, 'whole');
figure,
plot(omega./(2*pi), abs(H));
title('|DTFT| of the filter H_3(f)');
grid;

% We expect more or less a flat behaviour, with gain = 1, for all
frequencies

% filter the signal z
z_1 = filter(B_1, A_1, z);
z_2 = filter(B_2, A_2, z);
z_3 = filter(B_3, A_3, z);

%% 5. [1 extra pt]

%  If you have available only signals z and z_3, is there a way to find an
%  equivalent version of the signal z_2 which does not imply using the
filter H2(z)?
%  If yes, combine the two signals (z and z_3) accordingly and define the
signal z_4.

% We can build the result of the notch filter by averaging the initial
% signal and its all-pass filtered version (see previous exams)
z_4 = (z + z_3) / 2;

% Not required:
% The resulting output filter = (1 + H_3(z))/2 would be
rho_z = 1.2;
B_4 = (1 + rho_z^2)/2*[1, 0, 1];
A_4 = [rho_z^2, 0, 1];
% filter behaviour
[H, omega] = freqz(B_4, A_4, 2048, 'whole');
figure,
plot(omega./(2*pi), abs(H));
title('|DTFT| of the filter H_4(f)');
grid;
```