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

date: January 26, 2022

Ex.1 (Pt.13)

We need to apply the first derivative ($h[n]=\{1,-1\}$) in real-time to a digital signal $x[n]=\{1,2,0,-1,0,1,\ldots\}$ working in the frequency domain on blocks of 4 samples.

1. [4 pts] Describe how to get the DFT for each block of the signal and for the filter according to the requested task.
2. [7 pts] Using the Overlap and Save algorithm, describe how to find the output for the given signal, provide all the intermediate numerical results and the steps to obtain the final result.
3. [2 pts] Check that working in the time domain you would get the same result.

Ex.2 (Pt.8)

Given a filter with the following z-transform:

$$H(z)=\frac{1+2z^{-2}}{2+z^{-2}}$$

1. [2 pts] Provide its zeros-poles plot.
2. [2 pts] Provide an approximate representation of its amplitude and phase response.
3. [4 pts] Depict a logical circuit to use it as a band-stop filter: which normalized frequency will be removed?

Ex.3 (Pt.12)

To be solved with MATLAB and to be uploaded on Webeep assignments folder

1) [3 pt] The signal x1(t) = cos(2*pi*f1*t) has frequency f1 = 500 Hz. The signal x2(n) = cos(2*pi*f2*(n-5)) is sampled with sampling rate Fs, it has frequency f2 = 0.025 and it is defined for n = 0:999. Define the signal x(n) as the sum of the two signals, such that it repeats periodically every 680 samples.
    - In which time sample do you find the first maximum value of the signal x(n)? Why?
2) [3 pt] We want to convert the sampling rate of the signal x(n) to a new rate of 12750 Hz.
    - Implement the sampling rate conversion, using (if needed) a low-pass filter with 65 samples. Define the final signal as y(n).
    - Compute the DFTs of the signals x(n) and y(n) over 2048 samples and plot their absolute values as a function of the normalized frequency axis, starting from frequency 0.
    - In the DFT of x(n) we can find some oscillations: which is the reason of this behaviour?
    - Where (in terms of normalized frequencies) do you expect to see the DFT peaks of y(n)? (you can leave their values parametric).
3) [3 pt] You are given an LTI system H(z), causal stable and with real coefficients, with this finite-difference equation: y(n) = a*x(n) + b*x(n-1) + c*x(n-2) + d*y(n-1) + e*y(n-2).

Continues on the back

- Choose the values of parameters a, b, c, d, e such that the filter deletes the sinusoidal component at the lowest frequency from the signal y(n), maintaining unaltered the rest of the spectrum, and such that h(n=0) = 2.
- Verify the filter behaviour by using the function 'freqz' over 2048 samples, plotting the absolute value of the frequency response of H(z) over the entire frequency spectrum in normalized domain.

4) [3 pt] Given the same LTI system parametric definition as in 3), choose the values of a, b, c, d, e to define an all-pass filter (causal stable and with real coefficients) with the minimum possible number of coefficients for the denominator and numerator:

- In this case, is it possible to have h(n=0) > 1? If yes/no, why?
- Verify the filter behaviour by using the function 'freqz' over 2048 samples, plotting the absolute value of the frequency response and the phase response of H(z) over the entire frequency spectrum in normalized domain.
- How does the phase behave? Is it possible to realize an all-pass filter (causal and stable) with opposite phase?

Solutions

Ex.1

The DFT can be obtained using the $W^* = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$ matrix.

The DFT of the filter will be $H[k] = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1+j \\ 2 \\ 1-j \end{bmatrix}$

For the signal we have to use blocks of 4 samples, but, for the Overlap and save, in order to avoid the artifacts introduced by the circular convolution we have to add a ‘0’ before the first sample (since the length of the filter is 2 and one sample will jump back from the tail to the head of the block considered).

Therefore, we get:

$$X_0[k] = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 2 \\ 0 \end{bmatrix} = \begin{bmatrix} 3 \\ -2-j \\ 1 \\ -2+j \end{bmatrix} \quad \text{and} \quad X_1[k] = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 0 \\ -1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 2j \\ 0 \\ -2j \end{bmatrix}$$

$$Y_0[k] = X_0[k] \times H[k] = \begin{bmatrix} 0 \\ -1-3j \\ 2 \\ -1+3j \end{bmatrix} \rightarrow y_0[n] = \frac{1}{4} W \cdot Y_0[k] = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} \begin{bmatrix} 0 \\ -1-3j \\ 2 \\ -1+3j \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ -2 \end{bmatrix}$$

$$Y_1[k] = X_1[k] \times H[k] = \begin{bmatrix} 0 \\ -2+2j \\ 0 \\ -2-2j \end{bmatrix} \rightarrow y_1[n] = \frac{1}{4} W \cdot Y_1[k] = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} \begin{bmatrix} 0 \\ -2+2j \\ 0 \\ -2-2j \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ 1 \\ 1 \end{bmatrix}$$

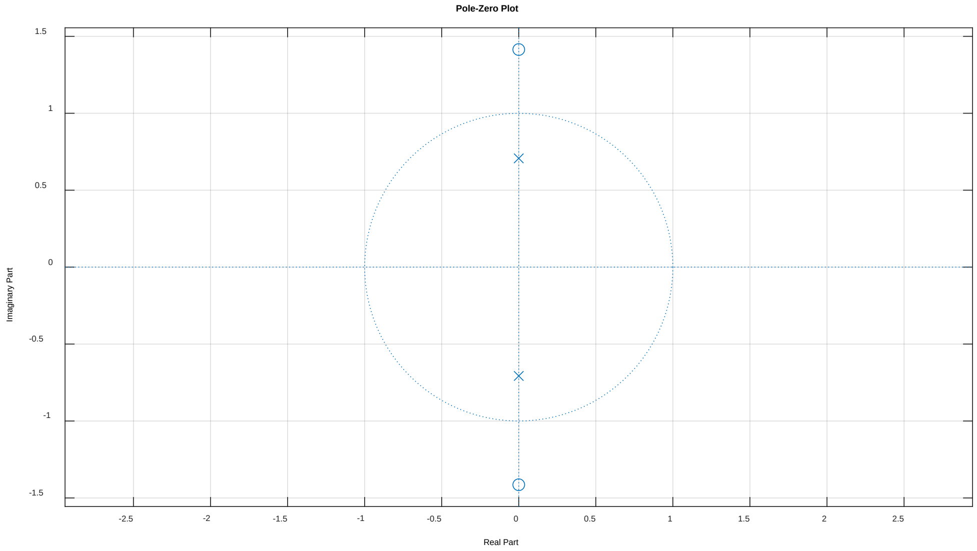
Removing the first output from every block to remove the circular convolution artifacts, we get:

$y[n] = \{1, 1, -2, -1, 1, 1\}$.

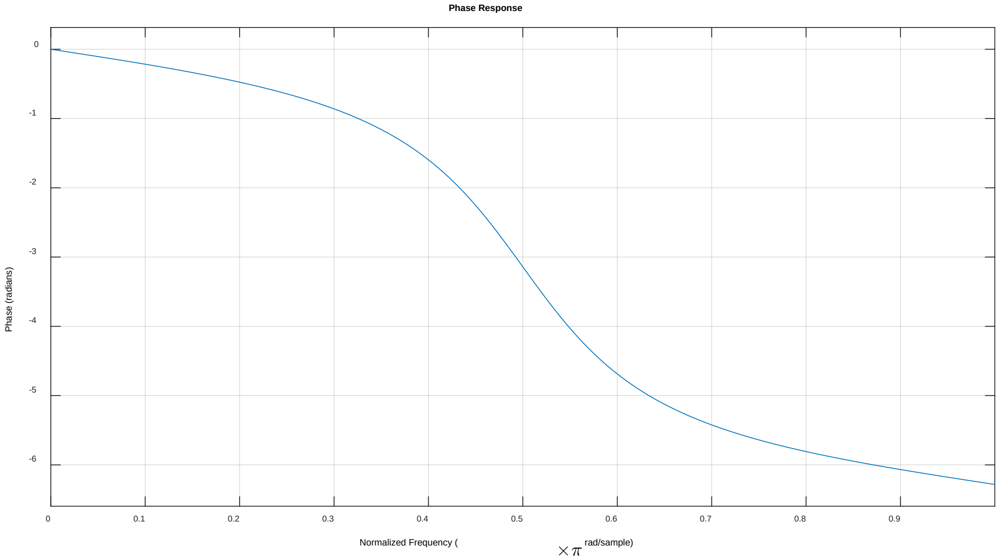
Convolving the original signal with the same filter in the time domain the result is the same.

Ex.2

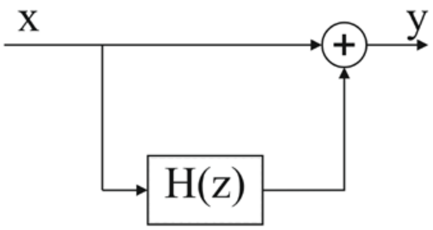
It is an all-pass filter so its amplitude will be constant and equal to 1.



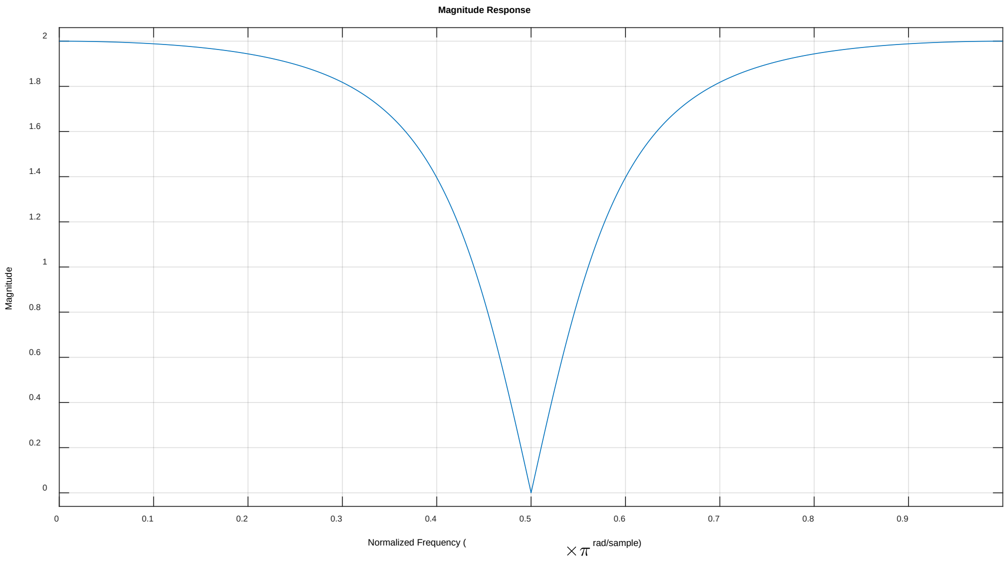
The phase response is:



The circuit will be:



The frequency that will be completely removed is $\omega = \pi/2$:



To preserve the same amplitude of the original signal a gain of ½ should be added.

Ex.3 (MATLAB CODE)

```
close all
clearvars
clc

%% 1.

% [3 pt] The signal x1(t) = cos(2*pi*f1*t) has frequency f1 = 500 Hz.
% The signal x2(n) = cos(2*pi*f2*(n-5)) is sampled with sampling rate Fs,
% it has frequency f2 = 0.025 and it is defined for n = 0:999.
% Define the signal x(n) as the sum of the two signals, such that it
% repeats periodically every 680 samples.
% In which time sample do you find the first maximum value of the signal x(n)? Why?

f1 = 500;
f2_n = 0.025;
N = 1000;
n = 0:N - 1;
P_samples = 680;

% the period of x(n) is the least common multiple between the period of
% x1(n) and the period of x2(n). We can find the period of x1(n), its
% normalized frequency and the sampling rate Fs.
P2_samples = 1/f2_n;
P1_samples = P_samples/P2_samples;
f1_n = 1/P1_samples;
Fs = f1/f1_n;

% define the signals
x1 = cos(2*pi*f1_n*n);
x2 = cos(2*pi*f2_n*n);
% shift the second signal by 5 samples by exploiting the convolution with a
% delta(n-5)
delta_5 = zeros(size(n));
delta_5(6) = 1;
```

```
x2 = conv(x2, delta_5);
% support of the convolution
n_conv = 0 : n(end) + n(end);
% the final signal is defined for n = 0:999.
x2 = x2(n_conv <= n(end));

% % another way to define the same signal can be to directly define the
% % shifted signal
% x2 = cos(2*pi*f2_n*n - 2*pi*f2_n * 5);
% % the only difference is that here you don't have zeros at the beginning

% define the final signal
x = x1 + x2;

figure;
plot(n, x);
grid;
title('x(n)');
% the first maximum value of the signal x(n) is in the sample n = 85.
% this is due to the fact that x2(n) is shifted by 5 samples, thus its
% peaks periodically repeat at n = 5, n = 45, n = 85, etc.., in general in n = 5 +
K*40
% the peaks of x1(n) periodically repeat at n = K*17.
% the first match between the peaks of the two signals will be in 85 =
% 17*5.

%% 2.

% [3 pt] We want to convert the sampling rate of the signal x(n) to a new rate of
12750 Hz.
% Implement the sampling rate conversion, using (if needed) a low-pass filter
% with 65 samples. Define the final signal as y(n).
% Compute the DFTs of the signals x(n) and y(n) over 2048 samples and plot
% their absolute values as a function of the normalized frequency axis,
% starting from frequency 0.
% In the DFT of x(n) we can find some oscillations: which is the reason of this
behaviour?
% Where (in terms of normalized frequencies) do you expect to see the
% DFT peaks of y(n)? (you can leave their values parametric).

Fs_new = 12750;
[L, M] = rat(Fs_new / Fs);

% 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
y = x_f(1:M:end);

% compute the DFTs
MM = 2048;
X = fft(x, MM);
Y = fft(y, MM);

freq_axis = 0:1/MM:1 -1/MM;
```

```matlab
figure;
leg = {};
plot(freq_axis, abs(X));
leg{1} = '|DFT| of the signal x(n)';
hold on;
plot(freq_axis, abs(Y));
leg{2} = '|DFT| of the signal y(n)';
grid;
legend(leg);

% we find oscillations in X(f) due to the zero-padding we applied when
% computing the DFT over 2048 samples. This corresponds to a windowing
% operation, thus the DTF will contain artifacts due to the convolution
% with a periodic sinc.

% we expect to see two peaks in Y(f) in the normalized frequencies f1_n*M/L
% and f2_n*M/L, together with other two peaks in symmetric position.

%% 3.

% [3 pt] You are given an LTI system H(z), causal stable and with real coefficients,
with this
% finite-difference equation: y(n) = a*x(n) + b*x(n-1) + c*x(n-2) + d*y(n-1) + e*y(n-
2).
% Choose the values of parameters a, b, c, d, e such that the filter deletes
% the sinusoidal component at the lowest frequency from the signal y(n),
% maintaining unaltered the rest of the spectrum, and such that h(n=0) = 2.
% Verify the filter behaviour by using the function ,Äòfreqz,Äô over 2048 samples,
% plotting the absolute value of the frequency response of H(z)
% over the entire frequency spectrum in normalized domain.

% in Z domain, we find these expressions:
% B = [a, b, c];
% A = [1, -d, -e];

% We want to design a notch filter that deletes the frequency f2_n*M/L
% The notch filter should have this expression:
% theta = 2*pi*f2_n/L*M;
% B = conv([1, - exp(1i*theta)], [1, - exp(-1i*theta)]);
% A = conv([1, - rho* exp(1i*theta)], [1, - rho* exp(-1i*theta)]);
% by making the computations, we will find:
% B = [1, - 2*cos(theta), 1];
% A = [1, - 2*rho*cos(theta), rho^2];

% We want also that h(n=0) = 2 --> a should be = 2.

% Therefore, we can write that:
% B = 2*[1, - 2*cos(theta), 1];
% A = [1, - 2*rho*cos(theta), rho^2];
% Then, we have to substitute these values to the parameters a, b, c, d, e

theta = 2*pi*f2_n/L*M;
rho = 0.95; % this is an option, you can choose this value
a = 2;
b = -2*2*cos(theta);
c = 2;
d = 2*rho*cos(theta);
e = - rho^2;

B = [a, b, c];
A = [1, -d, -e];

% filter behaviour
[H, omega] = freqz(B, A, MM, 'whole');
```

```
figure,
plot(omega./(2*pi), abs(H));
title('|DTFT| of the notch filter H(f)');
grid;

%% 4.

% [3 pt] Given the same LTI system parametric definition as in 3), choose
% the values of a, b, c, d, e to
% define an all-pass filter (causal stable and with real coefficients)
% with the minimum possible
% number of coefficients for the denominator and numerator:
% In this case, is it possible to have h(n=0) > 1? If yes/no, why?
% Verify the filter behaviour by using the function ,Äòfreqz,Äô over 2048 samples,
% plotting the absolute value of the frequency response and the phase
% response of H(z) over the entire frequency spectrum in normalized domain.
% How does the phase behave? Is it possible to realize an all-pass filter
% (causal and stable) with opposite phase?

% we start from a system with these expressions:
% B = [a, b, c];
% A = [1, -d, -e];

% We can realize an all-pass filter with just two coefficients for the
% numerator and two coefficients for the denominator
% Therefore, c = e = 0

% In general, an all-pass filter (with real coefficients) can be defined as
% B = [a, b];
% A = [b, a];

% In our case, b should be = 1 to fulfill the requirements in the
% denominator.
% a should be < 1 to guarantee the system stability, therefore we cannot
% have a filter h(n=0) > 1.

b = 1;
a = 1/2; % this is an option, you can choose this value
B = [a, b];
A = [b, a];

% filter behaviour
[H, omega] = freqz(B, A, MM, 'whole');
figure,
plot(omega./(2*pi), abs(H));
title('|DTFT| amplitude of the all-pass filter H(f)');
grid;


figure
plot(omega./(2*pi), angle(H));
title('|DTFT| phase of the all-pass filter H(f)');
grid;

% The filter is maximum phase, as all the causal stable all-pass filters.
% We cannot realize an all-pass filter causal and stable with minimum
% phase, because the poles should be outside the unit circle and the constraint
% on stability will not be satisfied.
```