

Multimedia Signal Processing 1st Module

Date: 18/2/2026

Ex.1 (Pt.12)

A signal in the time domain has the following expression:

$$x[n] = \left(\frac{3}{4}\right)^n \cos\left(\frac{\pi}{4}n\right) + \left(\frac{3}{2}\right)^n \sin\left(\frac{3\pi}{4}n\right)$$

Using a FIR filter $h(n)$ we want to completely remove just the unstable components preserving the stable ones.

- Define the z transform of the filter
- Plot the zeros-poles plot of the filter
- Provide the expression $H(\omega)$ of the filter response at different normalized frequencies ω and represent an approximated behavior of its magnitude $|H(\omega)|$

Ex.2 (Pt.10)

A sinusoid at 15 KHz is sampled at 45kHz and then upsampled of an order of 4 (three zeros added every sample) .

- Define an ideal low pass filter (normalized and real cut frequency) to remove artifacts in the upsampled signal.
- Represent the spectrum in normalized frequencies:
 1. before, the upsampling,
 2. after upsampling before the filter.
 3. After upsampling and after the filter.

Ex.3 (Pt. 11 – MATLAB code)

- 1) [1.5 pt] The signal $x(n)$ is composed of two sinusoidal contributions (with the same amplitude = 1) at the normalized frequencies 0.25 and 0.05. The sampling rate is 16KHz and the number of samples is 1600. Define the signal $x(n)$.
- 2) [3 pt] Design a second order FIR filter, $H_{\text{fir}}(z)$, to attenuate the frequency components at 800 Hz. Design the filter by exploiting zeros and poles (not using the windowing method), considering roots with absolute value 0.9.
 - Plot the behaviour of the filter (absolute value) in the frequency domain, considering 1024 frequency samples and the normalized frequency axis.
 - Filter the signal $x(n)$ with $H_{\text{fir}}(z)$, defining $y(n)$.
 - Which is the value of $y(n = 0)$? Define it in Matlab (do not consider writing $y_0 = y(1)$), but specify also the numerical value that you expect.
- 3) [3.5 pt] Design a second-order all-pass filter, $H_{\text{ap}}(z)$, with roots at phase = $\pi/2$, $3/2\pi$ and poles with absolute value = 0.95.
 - Apply the filter to the signal $y(n)$, defining the signal $z(n)$.
 - Apply the filter to the signal $x(n)$, defining the signal $w(n)$.
 - Define the signal $m(n)$ as the arithmetic mean between $x(n)$ and $w(n)$.
 - Find the filter $H_m(z)$ such that $M(z) = H_m(z) \cdot X(z)$.
- 4) [3 pt] Compute the DFTs of the signals $x(n)$, $y(n)$, $z(n)$, $m(n)$ and plot their absolute values as a function of the normalized frequency axis, starting from frequency 0.
 - Comment on the position/amplitude of the peaks you expect to see for every signal. Motivate your answers.

Solutions

Ex.1

From the formulas:

$$\mathcal{Z}\{a^n \cos(\omega_0 n)u[n]\} = \frac{(1 - a \cos \omega_0 z^{-1})}{1 - 2a \cos \omega_0 z^{-1} + a^2 z^{-2}}$$

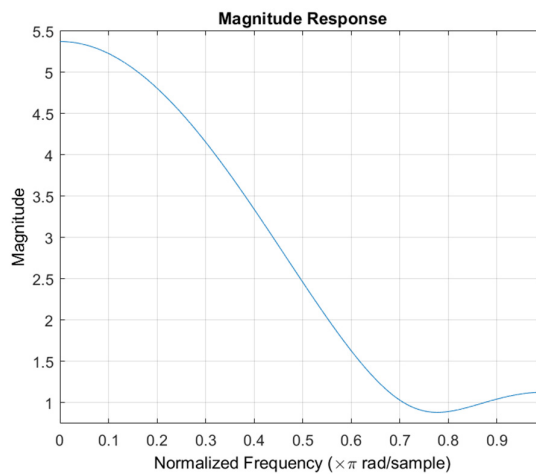
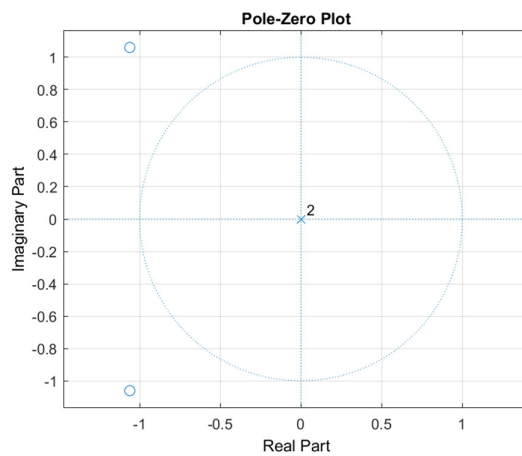
$$\mathcal{Z}\{a^n \sin(\omega_0 n)u[n]\} = \frac{a \sin \omega_0 z^{-1}}{1 - 2a \cos \omega_0 z^{-1} + a^2 z^{-2}}$$

We get the z-transform of the input signal (not requested):

$$X(z) = \frac{\left(1 - \frac{3\sqrt{2}}{8} z^{-1}\right)}{1 - \frac{3\sqrt{2}}{4} z^{-1} + \frac{9}{16} z^{-2}} + \frac{\frac{3\sqrt{2}}{4} z^{-1}}{1 + \frac{3\sqrt{2}}{2} z^{-1} + \frac{9}{4} z^{-2}}$$

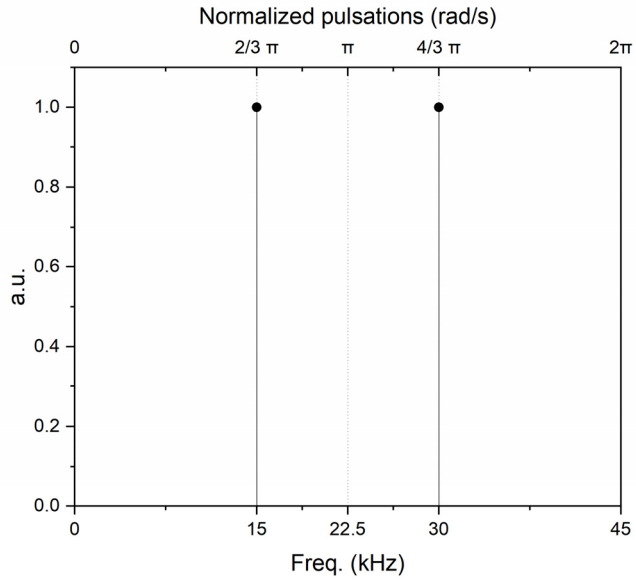
The second term presents two unstable poles and we need to remove them. The filter will then be:

$$H(z) = 1 + \frac{3\sqrt{2}}{2} z^{-1} + \frac{9}{4} z^{-2}$$

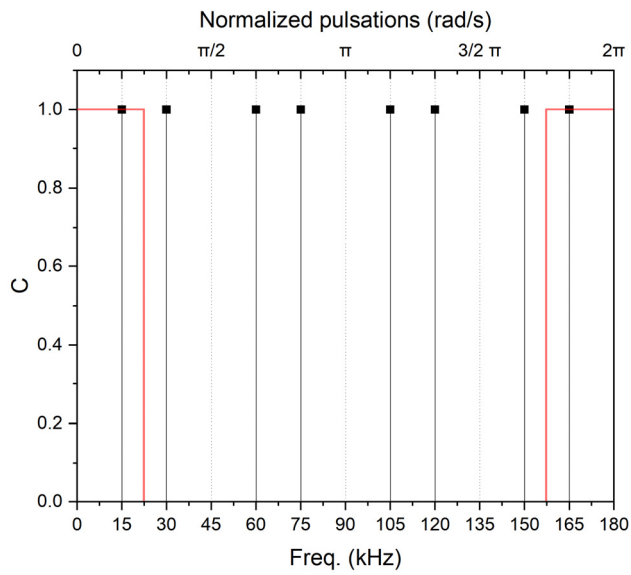


Ex.2

The original signal has a spectrum like this.

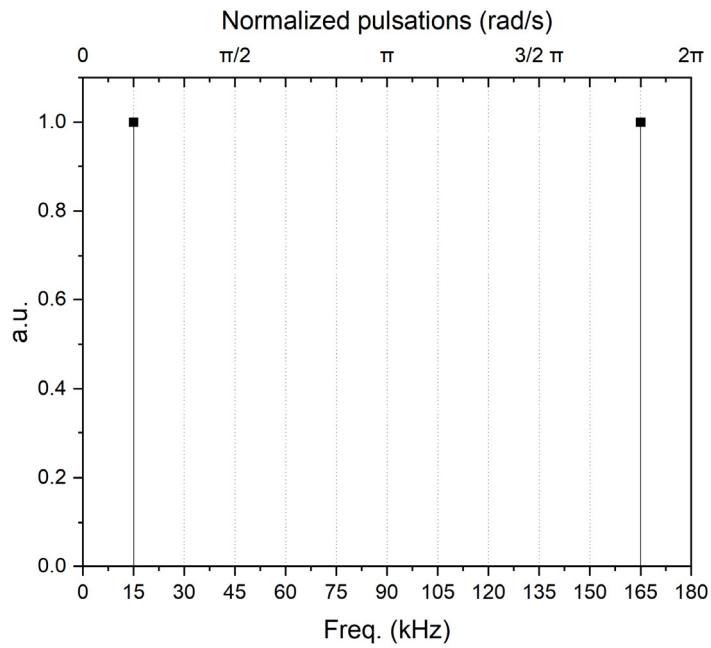


The upsampled signal will be:



The low pass filter (represented in red color) has a cut-off frequency of $\pi/4$.

The final signal will be:



Ex.3

close all

clearvars

clc

%% 1. [2 pt]

% The signal $x(n)$ is composed of two sinusoidal contributions (with the same

% amplitude = 1) at the normalized frequencies 0.25 and 0.05.

% The sampling rate is 16KHz and the number of samples is 1600.

% Define the signal $x(n)$.

f0 = 0.25;

f1 = 0.05;

N_samples = 1600;

Fs = 16e3; % up to now, we don't actually need this information

time_samples = 0:N_samples - 1;

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

```
%% 2. [3.5 pt]
```

```
% Design a second order FIR filter, H_fir(z), to attenuate the  
% frequency components at 800 Hz. Design the filter by exploiting zeros and  
% poles (not using the windowing method), considering roots  
% with absolute value 0.9.  
% Plot the behaviour of the filter (absolute value) in the frequency domain,  
% considering 1024 frequency samples and the normalized frequency axis.  
% Filter the signal x(n) with H_fir(z), defining y(n).  
% Which is the value of y(n = 0)? Define it in Matlab  
% (do not consider writing y_0 = y(1)), but specify also  
% the numerical value that you expect.
```

```
% 800 Hz corresponds to  $800/F_s = 0.05 = f_1$ 
```

```
% --> we should attenuate f1 component.
```

```
% no need for poles, just need for zeros
```

```
% -->  $H(z) = 1 - 2\rho\cos(\theta)z^{-1} + \rho^2z^{-2}$ , with  $\theta = \pi/10$ 
```

```
%  $H(z) = 1 - 1.8\cos(\pi/10)z^{-1} + 0.81z^{-2}$ 
```

```
B_fir = [1, -1.8*cos(pi/10), 0.81];
```

```
% Plot the behaviour of the filter (absolute value) in the frequency domain,  
% considering 1024 frequency samples and the normalized frequency axis
```

```
[H_fir, omega] = freqz(B_fir, 1, 1024, 'whole');
```

```
figure,
```

```
plot(omega./(2*pi), abs(H_fir));
```

```
title('|DTFT| of the filter H_{fir}(z)');
```

```
grid;
```

```
xlabel('f [norm]');
```

```
% Filter the signal x(n) with H_fir(z), defining y(n)
```

```
y = filter(B_fir, 1, x);
```

```
% Which is the value of y(n = 0)? Define it in Matlab
```

```
% (do not consider writing y_0 = y(1)), but specify also
```

```
% the numerical value that you expect.
```

```
y_0 = x(1)*B_fir(1);
```

```
% we expect that y_0 = x(n=0) * h(n=0)
```

```
% x(n=0) = 2, as we have the sum of two cosinusoidal signals being 1 in n=0.
```

```
% h(n=0) = 1 (the coefficient of the z^{0} term)
```

```
% --> y_0 = 2 * 1 = 2
```

```
%% 3. [3.5 pt]
```

```
% Design a second-order all-pass filter, H_ap(z), with roots
```

```
% at phase = pi/2, 3/2pi and poles with absolute value = 0.95.
```

```
% Apply the filter to the signal y, defining the signal z.
```

```
% Apply the filter to the signal x, defining the signal w.
```

```
% Define the signal m(n) as the arithmetic mean between x(n) and w(n).
```

```
% Find the filter H_m(z) such that M(z) = H_m(z) · X(z).
```

```
% it's easier to start from the denominator, and then build the numerator
```

```
% as its tilde-version
```

```
% poles with absolute value = 0.95 and phase pi/2, 3/2pi -->
```

```
% A_ap = 1 - 2*rho*cos(theta)z^{-1} + rho^2*z^{-2} = 1 + 0.9025*z^{-2}
```

```
A_ap = [1, 0, 0.9025];
```

```
% B_ap = fliplr(A_ap)
```

```
B_ap = [0.9025, 0, 1];
```

```
% filter the signal y
```

```
z = filter(B_ap, A_ap, y);
```

```
% filter the signal x
```

```

w = filter(B_ap, A_ap, x);

% define the signal m
m = (x + w)./2;

% Find the filter H_m such that M(z) = X(z) * H_m(z)
% M(z) = (X(z) + X(z)*H_ap(z)) / 2 = X(z) (H_ap(z) + 1)/2
% --> H_m(z) = (H_ap(z) + 1)/2.
B_m = (B_ap + A_ap)./2;
% B_m = [(1 + 0.9025), 0, (1 + 0.9025)]/2;
% --> B_m = (1 + 0.9025)/2 * [1, 0, 1];
A_m = A_ap;
% --> This is a notch filter in f0, which corresponds to pi/2.
% We can understand it by looking at the position of the zeros and poles:
% they have the same phase, but the zeros now are on the unit circle.

% to better analyze the filter (not required)
figure;
zplane(B_m, A_m);
title('Z-plane of filter H_m(z)');
grid;
[H_m, omega] = freqz(B_m, A_m, 1024, 'whole');
figure,
plot(omega./(2*pi), abs(H_m));
title('|DTFT| of the filter H_{m}(z)');
grid;
xlabel('f [norm]');

%% 4. [3 pt]

% Compute the DFTs of the signals x(n), y(n), z(n), m(n)
% and plot their absolute values as a function of the normalized frequency
% axis, starting from frequency 0.

```

```
% Comment on the position/amplitude of the peaks you expect to see
% for every signal.
```

```
X = fft(x);
```

```
Y = fft(y);
```

```
Z = fft(z);
```

```
M = fft(m);
```

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

```
figure;
```

```
stem(freq_axis, abs(X));
```

```
title('Absolute value of the DFT of the signal x(n)');
```

```
grid;
```

```
xlabel('f [norm]');
```

```
% We find four peaks in f0, f1, 1-f0, 1-f1. They have the same amplitude.
```

```
figure;
```

```
stem(freq_axis, abs(Y));
```

```
title('Absolute value of the DFT of the signal y(n)');
```

```
grid;
```

```
xlabel('f [norm]');
```

```
% We find four peaks in f0, f1, 1-f0, 1-f1.
```

```
% The peaks in frequency f1 are attenuated with respect to those of x(n),
```

```
% because the filter has zeros in f1 with absolute value = 0.9.
```

```
% The peaks in frequency f0 are not attenuated by the filter.
```

```
% Their amplitude differs from that of x(n) because the filter introduces
```

```
% a gain in f = f0 that is different from 1.
```

```
figure;
```

```
stem(freq_axis, abs(Z));
```

```
title('Absolute value of the DFT of the signal z(n)');
```

```
grid;
```

```
xlabel('f [norm]');
```

```
% We find basically no differences with respect to  $y(n)$ , because  $z(n)$  has  
% been defined as  $y(n)$  filtered through an all-pass filter, that does not  
% modify the absolute value of the signal.
```

```
figure;
```

```
stem(freq_axis, abs(M));
```

```
title('Absolute value of the DFT of the signal  $m(n)$ ');
```

```
grid;
```

```
xlabel('f [norm]');
```

```
% The filter is a notch in  $f = f_0$ . Therefore, we find only the  
% contributions at  $f_1$ , with an amplitude which is similar to that of  $x(n)$ .
```