Middle East Technical University

Dept. of Electrical and Electronics Engineering

EE301 (2018-2019 Fall Semester)

HW4-MATLAB Question (Due 30 December 2018)

A discrete time (DT) real sequence $x[n]$ is defined to represent the samples of a given bandlimited analog speech signal $x_c(t)$, where the sampling rate is F_s (Hz) (each sample shows the amplitude level of the real signal). First, we read data from the audio file named 'speech.wav', and obtain sampled data sequence, $x[n]$ in MATLAB:

```
close all, clear all;
[voice, Fs] = audioread('speech.wav');   % Read voice data from given file
                                         % should be in the same directory with code
x = voice / max(abs(voice)).';           % Normalize voice data
% sound(x, Fs);     % Listen to the original signal we will work on
Sampling_rate = Fs;     %44.1 kHz sampling rate
voice_len = size(x,1);
w_axis = (-Fs/2:Fs/(voice_len-1):Fs/2)/1e3;
figure, plot(w_axis, fftshift(abs(fft(x)))/Fs); % Observe magnitude response of the sample signal
                                               % by calculating DFT of the sequence
% fftshift(X), which is a Matlab built-in function, is used to rearrange a Discrete Fourier Transform X by shifting the zero-frequency component to the center of the array
title('Magnitude Response of the Original Signal');
xlabel('Frequency, kHz'), ylabel('Amplitude');
```

Throughout this question, we investigate the magnitude spectrum of related signals in MATLAB via DFT. The signal $x[n]$ above has length N= 441234, thus we can obtain the samples of continuous time Fourier Transform (CTFT) of $x_c(t)$ by just computing the N-point DFT of $x[n]$. It is important to note that the normalized frequency $\Omega = \pm\pi$ corresponds to $\omega = \pm\pi F_s$ for continuous time signals.

Before processing the signal $x[n]$ further, we need to increase the sampling rate by integer factor:

```
voice_dur = voice_len / Fs;
upsmp_rate = 10;
Fs = Fs * upsmp_rate;  %441 kHz sampling rate
t = 0 : voice_dur / (voice_len * upsmp_rate - 1) : voice_dur;
x = interp(x, upsmp_rate);
%A Matlab built-in function y = interp(x,r) increases the sampling rate of x, the input signal, by a factor of r. The interpolated vector, y, is r times as long as the original input, x
```

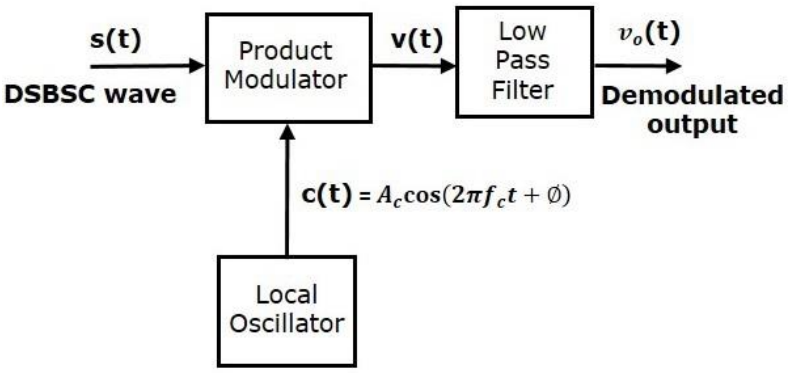
Based on the above settings, we investigate the magnitude spectrum of **Double-sideband suppressed-carrier transmission (DSB-SC)** in which frequencies produced by **amplitude modulation (AM)** are symmetrically spaced above and below the carrier frequency and the carrier level is reduced to the lowest practical level, ideally being completely suppressed.

1) ***DSB-SC Transmitter:*** Mathematically, we can represent the **equation of DSBSC** wave as the product of modulating and carrier signals:

$$s(t) = x_c(t)\cos(2\pi f_c t)$$

For f_c=100 kHz, first compute the DFT of $s[n] = s(n/F_s)$. Then, plot the magnitude spectrum of the DFT of $s[n]$. Note that this spectrum is closely related with the spectrum of $s(t)$, namely $|S(j\omega)|$ (see above Matlab commands).

2) ***DSB-SC Receiver:*** The process of extracting an original message signal from DSBSC wave is known as detection or demodulation of DSBSC. The same carrier signal (which is used for generating DSBSC signal) is desired to be used to detect the message signal. Hence, this process of detection is called as coherent or synchronous detection. Following is the block diagram of the coherent demodulator:



In this process, the message signal can be extracted from DSBSC wave by multiplying it with a carrier, having the same frequency and the phase of the carrier used in DSBSC modulation. The resulting signal is then passed through a Low Pass Filter. Output of this filter is the desired message signal. However, in practice, there is always some non-zero offset present both in phase and frequency of the local oscillator, which we call as the carrier phase offset ($\emptyset$) and carrier frequency offset (Δf) respectively. In that case, the demodulated signal has perceptional distortion that we would like to observe in the following steps.

From the above figure, we can write the output of product modulator as

$$v(t) = 2\,\mathrm{s(t)}\cos[2\pi(f_c + \Delta f)t + \emptyset]$$

a) For $\Delta f = 0$, and $\emptyset = 0, \frac{\pi}{4}, \frac{\pi}{2}, \pi$, first compute the DFT of $v[n] = v(n/F_s)$. Then, plot the magnitude spectrum of the DFT of $v[n]$. Note that this spectrum is closely related with the spectrum of $v(t)$, namely $|V(j\omega)|$.

b) For $\emptyset = 0$, and $\Delta f = 0, 1, 10, 100, 1000$ Hz, first compute the DFT of $v[n] = v(n/F_s)$. Then, plot the magnitude spectrum of the DFT of $v[n]$. Note that this spectrum is closely related with the spectrum of $v(t)$, namely $|V(j\omega)|$.

c) It is assumed that the signal $v(t)$ passes through an ideal low pass filter, whose cutoff frequency is below f_c. Then, find the demodulator output $v_o(t)$ in terms of $x_c(t)$, Δf, and $\emptyset$. Verify that $v_o(t) = x_c(t)$ if there is no offset in phase and frequency of the local oscillator.

d) The following Matlab script can be used to obtain the demodulated signal $v_o[n] = v_o(n/F_s)$:

```
fir_order = 128;   % The low-pass filter for demodulation will be an FIR filter of order 128
downsmp_rate = 10;
Fs = Fs / downsmp_rate;
```

```
demodulated_signal = decimate(v, downsmp_rate, fir_order, 'fir');
w_axis = (-Fs/2:Fs/(voice_len-1):Fs/2)/1e3;
figure, plot(w_axis, fftshift(abs(fft(demodulated_signal)))/Fs);
title('Magnitude Response of Demodulated Signal Under Phase and Frequency Offset');
xlabel('Frequency, kHz'), ylabel('Amplitude');
% A Matlab built-in function y = decimate(x,r,n,'fir') reduces the sampling rate of  x, the input signal, by a factor of r. The decimated vector, y, is shortened by a factor of r so that length(y) = ceil(length(x)/r). Here, decimation is used to reduce the sampling rate of a sequence to a lower rate. It is the opposite of interpolation. This operation also carries out low-pass filtering (FIR filter of order n) before down-sampling the data.
```

Then, obtain and plot the magnitude spectrum of the demodulated signal for different values of $\emptyset$ and Δf given in part (a) and (b). By using the following Matlab command, listen to the demodulated signal for different values of $\emptyset$ and Δf. Compare the demodulated signal $v_o[n]$ you obtain and listen to with the original message signal $x[n]$, and comment.

```
demodulated_signal = demodulated_signal / max(abs(demodulated_signal)); % Normalize demodulated message signal
sound(demodulated_signal, Fs);
```