AM Radio Modulation / Demodulation

1 Overview

In this project, you will develop an understanding of how AM radio signals are modulated and demodulated. The goal is to understand AM radio in both the time and frequency domain.

When working on the project, please follow the instructions and respond to each item listed. Your project grade is based on: (1) your Matlab scripts,(2) your report (plots, wave files, explanations, etc. as required), and (3) your final results. The project report should include all your scripts and all requested plots. It is often easier to combine these using Microsoft Word or Powerpoint. For example, you can copy/paste figures from Matlab into these applications. You must clearly display the associated problem number and label the axes and on your plots to get full credit.

The following should be uploaded to Moodle: the report in PDF format, the wave file of generated by your synthesizer, and a zipfile containing all your Matlab code. The files should be named: “lastname-project-2-part-ii.pdf”, “lastname-project-2-part-ii.wav”, and “lastname-project-2-part-ii.zip”.

2 Exercises

2.1 Fourier Transforms and Frequency Shifts

The following Matlab code loads a discrete-time waveform and plots its discrete Fourier transform (this is a sampled version of the DTFT). The plot represents the frequency domain content of the signal. The frequency axis is labeled by the frequency normalized by the sample frequency. Therefore, it ranges from -0.5 to 0.5 as the frequency ranges from the negative Nyquist frequency to the positive Nyquist frequency. It is equally common to normalize the frequency axis by Nyquist frequency and this results in a range from -1 to 1 (e.g., the Matlab filter design function `fir1` uses this normalization).

```
load proj2_wave1                % Load data
X = fft(x);                     % Compute fft
n = (length(X)-1)/2;
f = (-n:n)/n/2;                 % Freq/Fs range
subplot(2,2,1);                 % Use subplots save paper
plot(f,fftshift(abs(X)));
xlabel('Frequency / Sample Frequency');
ylabel('Fourier Magnitude');
title('X');
```

(a) Listen to this waveform with `soundsc(x,8000)`.

(b) Add a subplot showing `x1=exp(2i*pi*0.2*(1:length(x))).*x` in the frequency domain. Explain the result in terms of Fourier transforms.

(c) Add a subplot showing `x2=real(exp(2i*pi*0.2*(1:length(x))).*x)` in the frequency domain. Listen to this waveform with `soundsc(x2,8000)`. Explain the result in terms of Fourier transforms.

(d) Add a subplot showing `x3=cos(2*pi*0.2*(1:length(x))).*x` in the frequency domain. Listen to this waveform with `soundsc(x3,8000)`. Explain the result in terms of Fourier transforms.

2.2 Demodulation and Filtering

Starting with `x3` from the previous exercise, we will now demodulate the test signal.

(a) Make new figure (e.g., use `figure(2)`) and add a subplot for `x4=cos(2*pi*0.2*(1:length(x))).*x3`. Listen to this waveform with `soundsc(x4,8000)`. Explain the result in terms of Fourier transforms. At what center frequencies do the "images" appear? What must be done next to recover the original signal `x`?

(b) Type `help fir1` and consider the discrete-time filter `h=fir1(40,0.2)`. This lowpass filter has a cutoff frequency of 0.2 times the Nyquist frequency. In DT, it is common to define the *normalized frequency* as a fraction of the Nyquist frequency. This allows one to discuss filter properties without defining the sampling frequency. The frequency response $H(e^{j\omega})$ associated with the impulse response `h` can be displayed in a new figure using `figure(3);freqz(h)`.

(c) Next, we pass the signal `x4` through thew filter `h` with `x5=filter(h,1,x4)`. Plot `x5` in the frequency domain using the method from Section 2.1. Listen to this waveform with `soundsc(x5,8000)`. Explain the result in terms of Fourier transforms.

2.3 Tuning In

The following Matlab code loads a discrete-time waveform `x` sampled at 64 KHz that contains two 8 KHz AM radio channels. Then, it displays the signal in the frequency domain.

```
load proj2_wave2              % Load data
x = double(x)/32768;          % Convert from 16 bit to double
X = fft(x);                   % Compute fft
n = (length(X)-1)/2;
f = (-n:n)/n/2;               % Freq/Fs range
subplot(2,2,1);               % Use subplots save paper
plot(f,fftshift(abs(X)));
xlabel('Frequency / Sample Frequency');
ylabel('Fourier Magnitude');
title('X');
```

(a) Use this plot to estimate the center frequencies of the two AM channels.

(b) Using what you learned from the previous exercises, demodulate and filter each channel down to signals `y1` and `y2`. Plot Fourier spectrum after each operation.

(c) Don't forget that all the signal processing is taking place at 64 KHz. To listen to these waveforms, you must downsample by 8. If `y1` is your demodulated and filtered waveform, then use `soundsc(real(y(1:8:end)),8000)`.

(d) Finally, output the two signals to 8 KHz single channel wave files.