ECEG 470/670 Communication and Information Systems Fall 2025

Project #1: NBFM and WBFM Spectra

In this project you will use Matlab to generate sampled FM signals using single-frequency message signals with varying frequencies and then plot their spectra using the fast Fourier transform.

***Assignment*:**

1. Write a function in Matlab named "generateFM_lname.m," where lname should be replaced with your last name, that generates a vector that contains a sampled tone-modulated FM signal. Recall that an FM waveform can be expressed in the following form:

   $$\phi_{FM}(t) = A\cos\left[\omega_c t + k_f a(t)\right],$$

   where $a(t)$ is the time integrated from of the message signal $m(t) = A_{msg} \cos \omega_{msg}t$, A_{msg} is the amplitude, and ω_{msg} is the radian frequency of the modulating tone. Thus, $a(t) = (A_{msg}/\omega_{msg}) \sin \omega_{msg}t$. The Matlab function should accept values for A, ω_c, k_f, A_{msg}, ω_{msg}, and f_{samp} as inputs and produce the vector 'phiFM' as the output.

2. Write a Matlab script named "eceg470project1_lname.m" that calls the function from part 1 above to generate sampled NBFM and WBFM waveforms with two different modulating tones (not two tones at the same time). Name the vector generated by the function 'phiFM' to maintain consistency between m-files. The carrier frequency should be 1.0 MHz. Use a sampling frequency of f_{samp} = 4.0 MHz to minimize the effects of aliasing, and sample the waveform for 0.25 seconds. The message frequencies (f_{msg}) that you select should be widely separated and lie somewhere in the range of 300 Hz to 10 kHz. The goal is to compare NBFM to WBFM spectra for low and high-frequency audio. Specific values of k_f are not specified, but they should be chosen to generate reasonable NBFM and WBFM waveforms for each audio tone, and they should be appropriate values given the amplitude A_{msg} that you choose. Assign values to the various parameters at the top of the script and use the variables throughout the script. That is, none of the parameter values should be "hard-wired." You might add a list of definitions at the beginning like the following:

   ```
   fsamp = 4e6;                % sample frequency (Hz)
   tstart = 0;                 % sample start time (sec)
   tend = 0.25;                % sample stop time (sec)
   Amsg = 1.0;                 % message tone amplitude (V)
   fmsg = 1000;                % message tone frequency (Hz)
   A = 1.0;                    % FM carrier amplitude (V)
   fc = 500e3;                 % FM carrier frequency (Hz)
   kf = 1e5;                   % FM deviation constant (rad/V)
   fstart = 450e3;             % spectrum plot start freq. (Hz)
   fend = 550e3;               % spectrum plot stop freq. (Hz)
   ```

   Of course, you are encouraged to (and should) assign values other than the ones shown above.

Within the script, calculate the FFT of each waveform and then plot the magnitude spectrum of the waveform in decibels. You may ignore the redundant upper half of the output vector of the FFT (i.e., you do not have to plot the upper half). Your spectrum plot should be zoomed in on the signal; that is, your final plot should not show the entire spectrum from 0 to $0.5f_{samp}$. There is likely to be a lot of low-amplitude "noise" in the FFT output consisting of spikes that are many tens of decibels below the signal of interest. You should therefore plot only those parts of the FFT that are within a few tens of dB of the peak value (say, within 40–60 dB of the peak). One way to accomplish this is to set the limits of the vertical axis using code similar to the following (variable `spectrumpk_dB` is the maximum magnitude of the spectrum expressed in decibels):

```
ymax = 10*ceil(spectrumpk_dB/10);
ylim([(ymax - 60) ymax]);
```

Be sure to include appropriate axis labels. Use an appropriate unit for the frequency axis so that the axis labels do not have to be displayed in scientific notation. If you can, adjust the font sizes and change to bold face as appropriate to make the plot more readable. Consider adding a distinctive title to each plot.

You should end up with four plots that illustrate the following cases:

NBFM with low-frequency tone
NBFM with high-frequency tone
WBFM with low-frequency tone
WBFM with high-frequency tone

3. Copy the plots into a document and add enough annotations above each one to distinguish them from each other. The annotations should include the values of A, f_c, k_f, A_{msg}, and f_{msg}, and any other information that you think would be helpful. Also add the peak frequency deviation Δf_{FM} and bandwidth B_{FM} given by Carson's rule to the plot information.

   Write a short paragraph that briefly explains the reasons for the important differences between the various plots. That is, write a short reflection that interprets your results and shows that you understand the difference between NBFM and WBFM and how their spectra differ for various modulating tones.

   You might notice what appears to be a discrepancy between the apparent bandwidth of the signal and the bandwidth predicted by Carson's rule. Examine closely the amplitudes of the sidebands and how the amplitudes of the ones close to f_c compare to the amplitudes of the ones farther out. Think about where you might define the threshold between "significant" and "negligibly small" sidebands. Add your reflections on the bandwidth issue to your comments.

4. Scan the document into PDF format and submit it to the course Moodle site by **11:59 pm on Friday, October 10, 2025**. Also, e-mail to me the two Matlab scripts that your wrote, "eceg470project1_lname.m" with the main code and "generateFM_lname.m" with the FM waveform sampling code. Again, the text "lname" should be replaced with your last name.