Lab 2 - Communication signals

Goal

Basic communication waveform simulation, generation, and detection.

Background

In addition to instrument control, *LabView* has a powerful MatLab like interface called *MathScript* that can be used for simulation, generation, and analysis. We will use of *MathScript* because it allows us to couple the widely used simulation capabilities of *MatLab* with *LabView's* instrument control and real-time data analysis capabilities.

Prelab

We will generate some Matlab simulations of the time waveforms and power spectra of simple communication signals. We will transfer the *.m* files in MatLab that created the waveforms to a *MathScript* node in *LabView* to generate and analyze the waveforms.

Reading

1. Material to be reviewed: All of Chapter 5 and 6 from Ziemer and Tranter (ZT), ECE 109, and ECE 153.
2. Read pages 3-10 through the section titled noise marker in the Application Note on passband noise measurements with an SA.
3. Some experience with MatLab is assumed. A link to a tutorial is here.
4. Read through the basics of creating a *MathScript* node within *LabView* using the following tutorial. In particular, note how to load a *.m* file (MatLab code) into a *MathScript* node. Additional information is here on how to create a waveform graph in *LabView* from the data generated by *MathScript*.
5. Read the description of white gaussian noise generation VI. You will use this in lab.

Simulations and Problems The baseline MatLab code for AM and FM signal generation is given in Computer Examples 3.1 and 3.2 in ZT (5th edition) The source code is here.

1. *AM waveforms.*
   Plot the time waveform and the power spectra for each following signals where $f_1 = 440$ Hz, f_2 =880 Hz, f_c =5.40 kHz, and $m = 0.5$. Plot the time waveforms over 20 cycles of the carrier. Plot the spectra centered on the carrier over twice the maximum frequency of the baseband signal. Label both the time and frequency plots changing the y-axis for the frequency plots to correspond to (one-sided) signal power since that is what a spectrum analyzer measures. Be sure that the sampled sequences that are used to determine the power spectra satisfy the sampling theorem to prevent aliasing. For a)-c) set $A = 1$.

   a) $A\cos(2\pi f_1 t)\cos(2\pi f_c t)$ (Single audio tone.)
   b) $A\,[1 + m\cos(2\pi f_1 t)]\cos(2\pi f_c t)$ (Large carrier single tone).
   c) $A\,[1 + \cos(2\pi f_1 t) + \cos(2\pi f_2 t)]\cos(2\pi f_c t)$ (two tones w/large carrier).
   d) Determine the constant A so that the voltage waveform in part (b) produces -20 dBm of power at the carrier frequency if the load resistance is 50 Ω. What is the power in each sideband as a function of m? What is the total power in the signal in dBm?
   e) Save the generation part for each of the waveforms in (a-c) to *.m* files. These files will be loaded into a *MathScript* node in Lab to generate the waveform in *LabView*.

2. *FM waveforms*
   Repeat for the following FM waveforms. For the power spectrum plots, use Table 3.2 to determine the frequency range that contains $> 98\%$ of the total power. Remember to save the generation part of each simulation to a separate *.m* file for use in Lab.

   a) Plot Eq. 3.88 in ZT with f_c =10 kHz, $f_m = 440$ Hz, and $\beta = Af_d/f_m = 10$. Set $A_c = 1$. Again use 20 cycles of the carrier for the time plot and a frequency range that contains $> 98\%$ of the total power.
   b) Same as (a), but with $\beta = 0.1$.

3. *Music File*
   To qualitatively test the communication systems we will use music files. A music file (*Music1*) in a pulse coded modulation format or PCM format which is a variation of the *.wav* or *.aiff* formats used on PCs and Mac is on the class Web site.

   a) **Optional** If you would like to do this with one of your own ***legal*** music files do the following:
      i. Convert a stereo music file that you own on a CD into *.wav* format. (iTunes can do this by setting the Import preferences.) Use mono for the import as we will only listen to one channel.
      ii. Re-sample the file using a sampling rate of 8 ksamples/s with 16 bits/sample. Edit the file down to about 30 seconds. There are many tools to do this.
   b) **Required** Import the *.wav* music file (either your own or *Music1*) into MatLab using **wavread**. You can hear the file in Matlab using **wavplay**. Matlab help is here.

4. *Synthesis of Noise*
All of the statistical data collected in lab are samples of an underlying random distribution. Here we discuss the relationship between the sample statistics used in simulation or collected in lab and the underlying random distribution that generated the statistics.

   a) N samples of a zero-mean Gaussian random variable X with expectation $E(X) = \mu_X$, and variance σ_X^2 are measured. The sample average and sample variance are defined as

$$\overline{X} = \frac{1}{N} \sum_{i=1}^{N} X_i$$

$$s_X^2 = \frac{1}{N-1} \sum_{i=1}^{N} \left(X_i - \overline{X} \right)^2$$

   Show that:

      i. $E(\overline{X}) = \mu_X$ (Sample mean $E(\overline{X})$ equals mean of underlying distribution μ_X.)
      ii. $\mathrm{var}(\overline{X}) = \sigma_X^2/N$. (Variance of the sample mean reciprocally scales with the number of samples. More samples produces less variance in estimate of the mean value.)
      iii. $E(s_X^2) = \sigma_X^2$ which states that the mean of the sample variance equals the variance of the underlying distribution. (Note: that is why s_X^2 is defined with $N - 1$ in the denominator.)

   b) Using these results, determine the number of samples required for the rms error σ in $\mathrm{var}(\overline{X})$ to be less than 10% of rms σ_X of the underlying distribution.

   c) Using the **data=normrnd(0, sigma,[1 N])** function in Matlab where N is determined in part (a), generate N samples of a zero-mean gaussian random variable with $\sigma = 0.5$. This will be our basic method to simulate thermal noise. Plot the histogram of these values using **hist(data,nbin)** where **nbin**=100 and calculate the sample mean (**mean(data)**) and sample variance (**var(data)**). Compare the values to those determined in part (a). Matlab help is here.

5. *Correction factor for averaged passband noise power measurements with an SA*
Following the example on p. 6 of the Agilent Application Note on noise measurements, derive the 2.51 dB correction factor stated on page 8 which is the correction factor when you average the log instead of taking the log of the average. Do this in three steps:

   a) Transform the Rayleigh pdf (Figure 5) setting $\sigma = 1$ and $R = 1$ to the pdf of the log of the noise using the random variable transformation $\mathrm{dB} = 20\log_{10}(v)$. Be careful with scaling when you manipulate the logs.

   b) Plot out the pdf. It should agree with Figure 6 in the Agilent notes. Estimate the peak amplitude of the pdf. (It should be greater than 0.05)

   c) Using the fact that the mean of the transformed distribution is 0.5035, and following the example on p. 6, show that the correction factor is 2.5068 which is rounded to 2.51 dB in the notes.

Lab

1 AM Signals

1. Following the *MathScript* node tutorial, create a VI called *AM.vi* and load the *.m* file from Prelab Part (b) into a *MathScript* node. Set the sampling rate, $f_s = 10^6$ samples/s. Using input terminals for the *MathScript* node, create front panel input sliders for f_1, f_2, f_c, m, and A. Create output terminals for f_s, the AM waveform data, and the AWG gain where $AWG = 1$.

2. Using f_s and the AM data, create a waveform data type in *LabView* (details). To do this you must take the AM data, and $dt = 1/f_s$ and insert them into the *Build Waveform* VI. You can find the VI under the "Waveform" icon under the functions palette. By default, the build waveform VI will only list "Y" for the input which is the AM data. Expand the VI (pull down the bottom) so that it displays dt (the terminal you will need to wire $1/f_s$ to). Connect the output terminal of the *Build Waveform* VI to a time graph VI, run the VI, and display the same simulated waveform in *LabView* as the simulated waveform in *MatLab*. Change the values of the input parameters on the front panel to see the effect on the simulated waveform.

3. Now connect the output terminals f_s, AWG gain, and the AM data to the inputs terminals of the *ArbINIT* sub VI. For now, we will treat *ArbINIT* as a "black box" that given the input parameters will generate a waveform at the output of the 5441 AWG. Later in Lab, we will open up *ArbGen* and look inside. Set the parameters of the of the *MathScript* VI to generate the waveform from part (b) of the Prelab changing the carrier frequency f_c= 540 kHz and only displaying *exactly* one period of the modulating wave. **IMPORTANT**: the 5441 AWG does not support any voltage inputs GREATER than (+/- 1V) so make sure the voltage will be within that range. The amplitude A of the waveform should be set so that the carrier power is -20 dBm into a 50 Ω load which was determined in Prelab.

4. Connect the output of the AWG to the AM loop antenna. Turn on the AM radio and set the channel to 540 kHz. If everything is connected properly, you should hear the "A" note above middle C over the radio at 440 Hz.

5. Now connect the output of the AWG to the digitizer. Use *AM_SA,* with a resolution bandwidth of 100 Hz to measure the peak power in the carrier and each sideband. Also measure the noise power. Dump the measured waveform to the your key drive.

6. Change the resolution bandwidth to 20 Hz and repeat. Which values change? Which stay the same? Why?

7. Load the *.m* file corresponding to part (a) from the AM part of Prelab which is suppressed carrier AM (with the tone still being at 440 Hz). Can you hear the signal on the radio? Is the tone at 440 Hz? Compare this sound with large carrier AM where the tone is at 880 Hz. (You will need to explain what is happening for the lab write-up.)

8. Repeat (1-4) with the music data file you generated or use the file *Music1* that is in the Class Folder on the computer. (You can also record your own voice into a file in lab.) The same commands are used in the *MathScript* node as in *MatLab*. Import the *.wav* file using **wavread**. (You will need to use the full path of the file to import it.) Listen to the file in

MathScript before connecting to the AWG using **wavplay**. You should be able to hear the music file on the AM radio at the same carrier frequency. For the power measurements, use a channel spacing of 10 kHz.

9. Load the *.m* file corresponding to part (c) of the AM section of Prelab which is large carrier two-tone modulation at 440 and 880 Hz. Save spectrum to your key drive being sure the spectrum shows both the noise floor and the maximum signal. There should be additional frequency components in the measured spectrum that are not in the simulation. These components are nonlinear distortion. Repeat using the stand-alone SA with the attenuation set to 40 dB. By comparing the spectrum of the stand-alone SA and the soft SA, determine if the nonlinear distortion is due to the generation of the signal or the detection of the signal.

2 FM signals

We will now repeat for FM waveforms using the upconverter to translate the FM carrier from 10 kHz used in the simulations to the FM band at 87.9 MHz. For this Section, we will be using National's Instruments VI for FM generated named *FM.vi*. This VI has inputs for f_m and f_d that can be used to set β from Prelab. (See ZT Section on FM on how to use the inputs on *FM.vi* to generate β.)

1. Open *FM.vi* and input the appropriate parameters. Set the carrier so that the modulated waveform is at 87.9 MHz, and use the resource name *"5610"* for the upconverter.

2. If set correctly, you should be able to hear the tone on the FM radio tuned to 87.9 MHz.

3. Display the spectrum using *FM_SA* using a frequency range that contains 98% of the power and dump the trace of the spectrum to your key drive.

4. Repeat for β=0.1.

3 Generation and Measurement of Noise Waveforms

Here we will generate white Gaussian noise and examine the temporal, spectral and statistical characteristics of noise-like signals. A general goal of modern communications is to make the transmitted waveforms as "noise-like" as possible as it can be shown that these types of waveforms can carry the most information in finite bandwidth.

Synthetic noise waveforms are generated by first generating an array of normally distributed random numbers with an rms voltage value σ. Ideally, the noise waveform generated from this array has a power density spectrum of $\sigma^2/(f_s/2)$ (W/Hz) where f_s is the sampling rate in Hz. The noise power over a noise bandwidth B_N is then

$$P = \frac{2B_N}{f_s}\sigma^2 \tag{1}$$

where the noise bandwidth is defined in Section 5.4.3 of ZT. In practice, the bandwidth of the resolution filter in the SA is not equal to the noise bandwidth and there is a correction factor of 0.25-0.5 dB depending on the type of SA used. This is discussed on pages 8-9 of the Agilent notes on measuring noise.

1. Create a new VI called *Noise.vi* and load the *Gaussian white noise* VI from the signal generation palette in *LabView.*

2. Create front panel controls for **standard deviation**, **seed**, **samples** and the sampling rate f_s which will used to determine the time between noise samples which will set the bandwidth of the generated noise waveform. Using the front panel, set the rms amplitude (standard deviation) of the noise source to 50 mV, the **seed** to -1, **samples** to 100k, and the sampling rate to $f_s = 50 \times 10^6$.

3. Connect the data array to a histogram (under statistics) and display a histogram of the noise values along with mean and standard deviation. Is the value of the histogram, which is the sample mean, consistent with the Pre-lab questions on the variance of the sample mean?

4. Build a noise waveform using the noise data array following the same procedure as in Part 1. Connect the output waveform to both a time and power spectral density (PSD) waveform display. The TA will show you which VIs to use for each of these functions. Using the measured value for the PSD noise power, determine the total noise power. Is this consistent the value of σ? Why or why not?

5. Create a front panel for the AWG gain. Now connect the output terminals f_s, to the inputs terminals of the *ArbINIT* sub VI. Also create a front panel for "filters enabled". This will turn and turn off real-time spectral shaping filters in the AWG. The TA will show how to connect this switch to the *ArbINIT* sub VI.

6. The AWG will generate a noise waveform with a bandwidth of $\approx$ 25 MHz. Open "*Cal SA*" VI, and view the noise waveform over a bandwidth of 32 MHz.

7. Using the yellow cursors measure the total power neglecting the enhanced response near DC (less than 1 MHz). This is caused by the fact that the digitizer is AC-coupled. Is this power consistent with the power measured using the PSD VI? Is it consistent the the power of the noise waveform?

8. Repeat with the stand-alone SA averaging 100 waveforms and using the same resolution bandwidth. Use the noise marker function **NOISE/Hz** which is under the Marker menu. This marker expresses the noise power density spectrum (dBm/Hz) and should account for both the difference between the resolution bandwidth and the noise bandwidth as well as the 2.51 dB difference between averaging the log and the log of the average.

9. Turn off the "filters enabled" switch on the front panel and view the noise waveform over a frequency range of about 100 MHz. Comment on the difference between the noise waveforms with and without the filters enabled.

Post Lab

1. Provide a quantitative explanation of the observations in Part 6 of Section 1.

2. Are the measured powers in the carrier and the sidebands of the AM signal the same as determined in Prelab? Why?

3. Provide a quantitative explanation of the observations in Part 9 of Section 1 by comparing the spectrum to that of single tone modulation using 440 and 880 Hz. Does superposition hold?

4. Compare and contrast the modulated AM signals generated from the music files with the corresponding commercial signals measured in Lab 1.

5. Compare the single-tone FM spectrum recorded in lab with the simulated spectrum. Discuss any discrepancies.

6. Using the results of Prelab, determine the variance of the estimated mean noise power and compare it to the measured value.

7. Discuss the results of Parts 6 and 7 of the noise generation section. What general conclusions can you draw about using sampled noise in place of a real noise source?

8. Are the noise powers measured using the stand-alone SA and the *Cal_SA* VI consistent? If not, provide possible reasons and how you would test to verify if your hypothesis is true?