Lab 6 Digital Transmission

Objectives:

- Understand process for encoding information into signals for transmission through base-band analog channels.
- Observe the impact of typical sources of corruption that create errors in the communication process, namely channel noise and bandwidth limits.
- Perform spectral estimation to analyze the relationship between encoded signal bandwidth requirements, channel noise, and bandwidth.

1. Background:

This lab considers encoding a digital data stream into analog signals for transmission through a base-band analog channel. Various line coding methods for digital base-band modulation will be implemented and the spectral properties of the codes will be examined. Simulation studies will be used to establish a relationship between bit error rates and signal corruption.

The classical model for communication channel corruption is a band-limiting filter (low-pass or band-pass filter) with additive Gaussian noise as shown in Fig. 1.

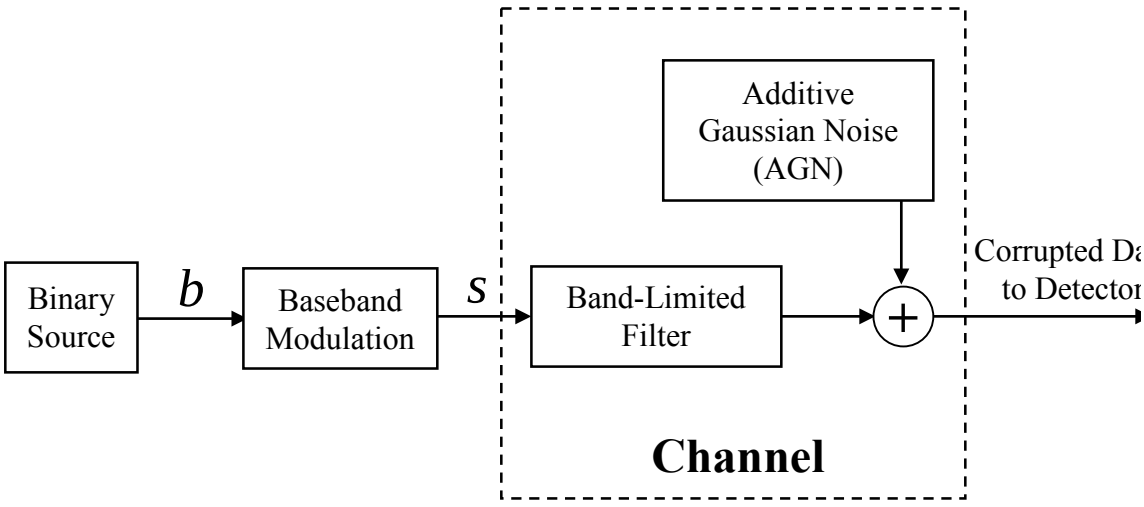

Figure 1. Block diagram of communication system showing the key components that interfere with the information transfer, such as limited bandwidth and noise.

Figure 1 illustrates the main components of a communication source and transmission. The information to be transmitted is converted into a binary sequence (i.e. ASCII codes for text or sequences of quantized signal amplitudes from sampled analog waveforms). The binary data source emits a series of 1's and 0's (bits) representing the *source*. The *baseband modulation* codes each bit into a continuous waveform for sending the source sequence over a physical channel, which is the medium the signal propagates through. Typical signals are formed by changing voltages and currents over a wire, light over an optical fiber, or electromagnetic energy through the atmosphere. No medium is totally free of noise (random perturbations on transmitted signal), therefore white or colored Gaussian noise is added to the transmitted signal to simulate this corruption. Noise is an irreversible corruption (i.e. permanent loss of information). While the signals can be filtered to exploit signal redundancies and reduce the impact of noise, there will always be a level of uncertainty in the received signal. Parameters extracted from signals denoting 1s and 0s must be sufficiently separated (i.e. amplitude, frequency, phase …) to limit the impact of the ambiguities introduced by the system noise.

In addition to noise, channels have limited bandwidth that can distort the signals by reducing the frequency content non-uniformly over its spectrum. So if the coded waveforms are not bandlimited to a value less than the channel bandwidth, the waveform will be distorted. Distortion is a deterministic change in the signal and can be reversed in some cases. If the distortion can be modeled as a linear filter, it can be reversed (the operation of undoing this distortion is referred to as deconvolution). In most cases of nonlinear distortion, such as clipping or quantization, the distortion cannot be reversed.

The baseband channel is simulated with a low-pass filter. *Baseband* and *low-pass* essentially means the same thing when describing a signal. When signals are modulated with a high frequency oscillator, their baseband spectrum is shifted up in frequency. If the modulated signal contains no significant DC energy, it is referred to as a passband signal. This lab considers only baseband signals and channels.

In order to send information over the channel, the information must be encoded into a sequence of binary digits and these digits are converted into analog signals for transmission using a variety of signaling formats called *line codes*. To help examine their spectral properties, a Matlab function, *modulb*(), was written to create various line codes from bit sequences. The function syntax is:

```
>>  [y,t] = modulb(binary_sequence, Fd, Fs, line_code_name);
```

where *binary_sequence* is a vector of 1's and 0's denoting the source binary sequence, *Fd* corresponds to the binary data rate in bits per second (b/s), *line_code_name* is a special string indicating the particular line code to be used (see help file for options), and *Fs* is the sampling frequency of the line code waveform used by the simulation. Note that sampling rate *Fs* should be much higher than the bit rate to simulate the analog signal (***at least*** by a factor of 10, 100 is preferred if it does not excessively slow down the system). The *modulb* function outputs the waveform as vector *y* and corresponding time axis *t*. The command

supports the following codes: '*unipolar_nrz*', '*bipolar_nrz*', '*bipolar_rz*', '*ami*', '*manchester*', '*miller*', '*unipolar_nyquist*', and '*bipolar_nyquist*'.

The type of line coding is selected to meet various system criteria, such as power requirements, bit timings (additional transitions of the line code signals within the bit interval can help in timing recovery), bandwidth efficiency (excessive transitions may require more bandwidth than available), low frequency content (some channels block low frequency), error detection, and complexity. Figure 2 shows analog waveforms for various line coding examples.

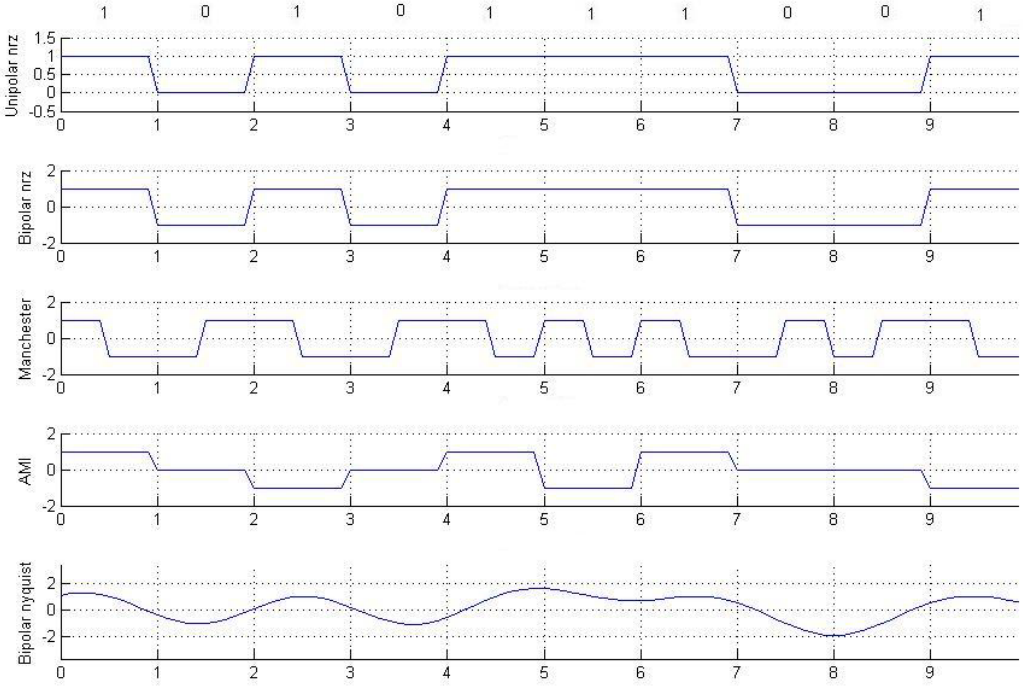

Figure 2. Line code waveform examples. Original binary sequence listed across the top is at a bit rate of 1, and applies to all codes.

2. Pre-Laboratory Assignment

1. Use the *modulb*() function to plot waveforms (similar to those in Fig. 2) representing the binary sequence $seq = \{0,1,0,0,1,1,0,0,0,1,1,1,0,0\}$ using the following line codes at a bit rate of $R_d = 1\,\text{kb/s}$ (choose a reasonable sampling rate for plotting the waveforms):

   a) unipolar NRZ (on-off signaling, NRZ= non-return to zero)
   b) bipolar NRZ (binary antipodal signaling)
   c) bipolar RZ (binary antipodal signaling RZ = return to zero)
   d) AMI (Alternate Mark Inversion)
   e) Manchester (split-phase);

f) Bipolar Nyquist (bipolar transmission where the pulse shape is a sinc function with main lobe extending to fill bit interval and side lobes extending beyond).

Turn in a hard copy of the plots with correctly labeled axes and a title for each plot including the name of the line code. Also include a hard copy of the code you used to generate and plot the waveform. For each plotted waveform describe in words how each line code encodes 0's and 1's into an analog waveform. Focus on the differences (i.e. unique features) between the codes in your descriptions.

2. Consider a simple communication system using bipolar NRZ, where a binary one is mapped into a 1 volt level and a binary zero is mapped into a -1 volt level. Assume the channel has no distortion and zero mean additive white Gaussian noise with a standard deviation of 0.7 volts. Sketch/plot the probability density function (pdf) for a line code sample corresponding to the case when a binary 1 was sent $p_v(v|b=1)$ and the pdf for a line code sample given a binary 0 was sent $p_v(v|b=0)$ (where v is the voltage level associated with the line code sample). Plot both on same axis. Clearly label the means of each distribution and the point where they intersect. Compute the expected probability of error when a threshold is used that corresponds to the point of intersection between the 2 pdfs. If line code sample is greater than this voltage, it is detected as a binary 1, and if less than this voltage, it is detected as a binary 0. Describe qualitatively what will happen to the probability of error (*i.e.* up, down, no difference) if the standard deviation of the Gaussian distribution decreases.

3. Laboratory Exercise

1. Generate a random binary sequence of 4000 bits by rounding off a set of uniformly distributed random numbers between 0 and 1 with the command:

```
>> seq = round(rand(1,4000));
```

Encode the sequence for each of the line codes listed below using the *modulb*() function.

a) unipolar NRZ
b) bipolar NRZ
c) bipolar RZ
d) AMI
e) Manchester;
f) Bipolar Nyquist

Set the data rate to R_d = 1 kb/s, the waveform sampling rate to F_s = 50 kHz, and compute the PSD for each encoded sequence. Plot the PSD in 2 ways. In both cases use a linear frequency scale, and in one case plot the PSD magnitudes in dB and in other case plot the PSD magnitudes on a linear scale. The parameters you select for

the *pwelch*() function used for computing the PSD (*i.e.* window length, overlap, number of FFT points) are important for seeing critical features of their spectra. Make sure you select a small enough window so that the spectrum is reasonably smooth. If jagged noise-like fluctuations exist over the spectrum, smooth them out by using a smaller window. On the other hand, if the window is too small, then the characteristic/important peaks will merge together due to lack of resolution. So this must be adjusted properly for best results. (Note that the window length should never be smaller than the bit interval so you do not distort the bit waveforms, this is a lower limit). You will need to identify major peaks and nulls from these plots. **Present spectra to the TA for each of the line codes, with correct axes labels and name of the line code in the title of the plot!**

The PSD plots indicate how efficiently each line code utilizes channel bandwidth. The more the spectral energy concentrates near zero, the less bandwidth the signal requires for transmission without distortion. To measure the bandwidth requirements for each line code, record the locations of the major peaks and nulls. In most cases this will be easier from the logarithmic scaled (dB) plots rather than the linearly scaled plot.

From the PSD plots, identify the locations of the first and second major spectral peaks (denoted by f_{p1} and f_{p2}) and the first and second spectral nulls (f_{n1} and f_{n2}). A spectral peak may exist at $f = 0$; however, the first peak for this measurement (f_{p1}) should not be counted at f=0. The first spectral null must also be selected for a non-zero frequency (i.e. a null at $f = 0$ is not included in this characterization). The location of the peaks and nulls should be entered in the table below. If a secondary peak and null cannot be located in the spectral range you are viewing, simply indicate *NA* for f_{p2} and f_{n2}. Create a table as shown in Fig 1 and include these number for each waveform.

Consider 3 metrics for computing the minimum required channel bandwidth for a given line code. Denote the minimum required channel bandwidth as *W*, and call the first metric *null 1* given by:

$$W_{n1} = f_{n1} \tag{1}$$

Call the second metric *peak average* given by:

$$W_{pa} = \frac{f_{p1} + f_{p2}}{2} \tag{2}$$

($f_{p2} = f_{p1}$ when f_{p2} does not exist). And call the third metric *null 2* given by:

$$W_{n2} = f_{n2} \tag{3}$$

($f_{n2} = F_s/2$ when f_{n2} does not exist). **Add these metrics to your table. Show the TA the completed table. Based on the metric you think best reflects the spectral distribution, indicate which line code would be most robust to distortion from limited bandwidth channels.**

Table 1: Spectral characteristics of line codes at 1 kb/s

Line Code	f_{p1}	f_{p2}	f_{n1}	f_{n2}	W_{n1}	W_{pa}	W_{n2}
Unipolar NRZ							
Bipolar NRZ							
Bipolar RZ							
Manchester							
AMI							
Bipolar Nyquist							

2. This next exercise examines the relationship between the line code PSD and the data rate F_d. For this part just consider the Manchester encoding and produce a PSD for different values of F_d , taken to be 250, 500, 1000, and 2000 b/s. Compute and plot the PSD (**just save the dB plots for later analysis**). Compute the *best* metric you determined in part 1, for minimum required bandwidth and present them in a plot versus F_d in the results section. **Propose a formula or approximate rule for the minimum required bandwidth as a function of F_d. This formula or rule should be based on the trends observed in the last plot. Show the TA this plot and your proposed formula or rule.**

In the next set of exercises, you will simulate the characteristics of an ideal baseband communication channel including additive white Gaussian noise (AWGN). The channel is modeled as an ideal low-pass filter whose output is summed with the output of a white Gaussian noise generator (see conceptual diagram of Fig. 1).

3. Now consider the effects of the channel on the waveforms. In particular, channel distortion is modeled as a low-pass filter, and Gaussian noise is added to the signal (after filtering) to simulate random amplitude fluctuations of the channel. Create a 14 bit binary sequence b shown below and encode it into an analog waveform using bipolar NRZ at a data rate of 2kb/s and sampling frequency of 40 kHz.

   >> $seq = \{0,1,0,0,1,1,0,0,0,1,1,1,0,0\}$

   From your observations in Part 1, determine the required transmission bandwidth W in Hz and apply a low-pass Butterworth with a cut-off at this frequency:

```
>> [b,a] = butter(4, W/(fs/2));
>> yf = filtfilt(b,a,y);
```

   In the above example *filtfilt*() was used instead of *filter*(). This filters the signal forward and backwards to result in effectively doubling the filtering order and

removing the delay and phase distortion. So when comparing plots, the filter delay does not confuse the comparison. Plot the original and the filtered version on the same graph and observe on differences.

Repeat the above procedure for a channel limited to $W/6$ (i.e. reduce the bandwidth by a factor of 6). **Show the TA the resulting plots. Describe the general effects of limited bandwidth on the waveforms and how this hinders accurate detection.**

4. Generate the waveform and a channel with bandwidth W, as used in Exercise 3. Add white Gaussian noise (AWGN) with noise power of 0.06 W (note power is equivalent to variance for a zero-mean Gaussian random number). Plot channel input and output and observe the changes. Also plot the PSD of the signal. Could you accurately estimate the original sequence based on visual inspection of the plots? Repeat the previous procedure with increasing the noise power. Find the noise power at which it is no longer possible to correctly estimate all of the bits correctly by visual inspection. **Show the the plots of this waveform with noise and the PSD for this final condition to the TA. Comment on the features of the PSD that suggests this signal has been corrupted with white noise.**