Binary Communication Transceiver Design with Alphabet Analysis

A. Simulation of a binary communication system

BLOCK DIAGRAM

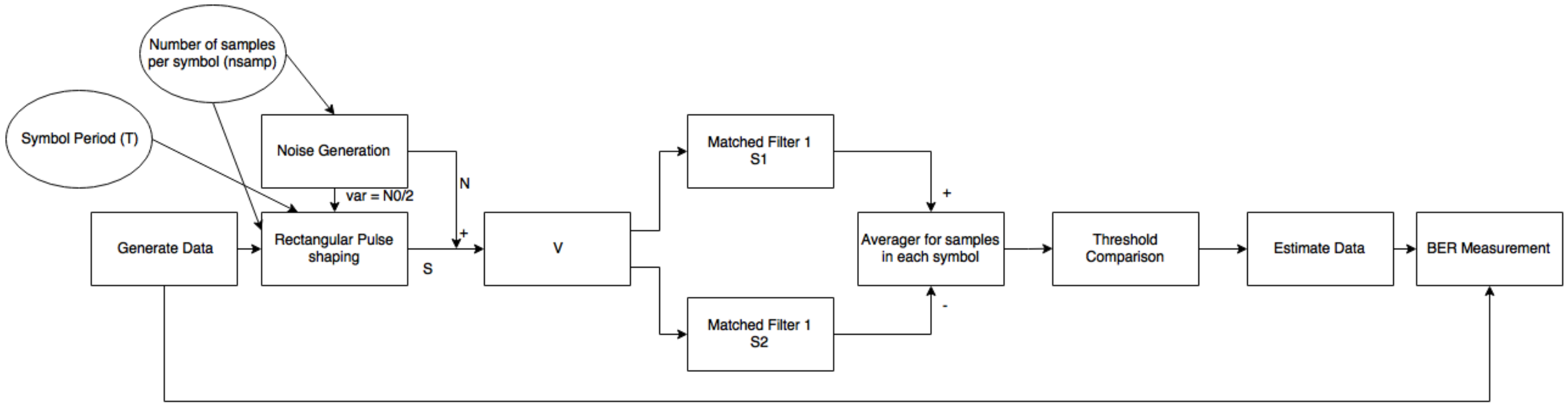

Figure 1. Block diagram of a binary communication system with matched filter receiver.

Some parameters are left for us to design. At first, I pick some random numbers for these parameters and start to implement each function. Symbol period T is default as 0.25s in pulse shaping. Amplitude of each pulse is determined by the function A = sqrt(SNR.*N0./T); Here N0 is determined by the noise we generate. Since we generate white Gaussian noise with zero mean and unit variance, and the variance of such distribution equals to N0/2, N0 in linear scale equals to 2. Amplitude is therefore determined by SNR and T. With less symbol period, the amplitude is higher. Another parameter is the number of samples for each symbol. The default value of nsamp is 6. Obviously, when the length of signal is L the length of noise generated is L*nsamp.

After pulse shaping and noise generation, we add up noise and signal to get V. This is supposed to be the received signal. In order to correctly decode the signal, we first need to use matched filters. We implement two matched filters because we have two rectangular pulse signals from our pulse shaping. Therefore, the matched filters are simply the time reversed and conjugate version of two rectangular pulse signals. Since the values are always real, we only need to time reverse two pulse signals. We then use filter.m to filter out signals with two matched filters. The numerator is the real values of the time reversed version of the two pulse signals and the denominator is 1. Since the matched filter returns large positive values when two signals matches and negative values when they don't. We subtract the filtered result of pulse signal 2 (S2) from that of S1. In my implementation, S1 is the positive rectangular pulse and S2 is the negative one. Therefore, after subtraction, positive value means bit 1 and negative value means bit 0. The threshold in this case is easily determined as 0 with E2-E1/2, because the energy of two pulse signals are equal to each other.

Before using threshold 0 to determine the estimate data, we need to average samples from each symbol to get a more accurate result. At first, I try to downsample from these sample values but the result is not promising. We need to average these samples because it is highly possible to sample an error value. Finally, we have an averaged value for each symbol, what's left is simply to use a threshold to determine the estimate data for each symbol.

In the end, we compare generate data and estimate data to measure the BER. Then theory BER in this case is calculated with BER = Q(sqrt(2*SNR)). This is because, we use antipodal instead of orthogonal rectangular pulse signal.

BER vs SNR

The figures show the BER measurement with different T and nsamp values. The length of signals L is set to be 10^6 but it is not enough to cover BER less than 10^ (-6). This is due to the limitation of computer memory. As you can see from Figure 2, small symbol period actually yields better BER rate. This is because the amplitude is inversely proportional to the symbol period with a fixed SNR value. With higher amplitude, the noise has less probability to change positive signals to negative and vice versa. Figure 3 shows that with fewer samples per symbol, we have better BER. This is, however, not true in the reality since we cannot rely on only a small number of sample from analog signal. In this project, the noise is relatively small and two samples are sufficient to estimate the data. If we have noise with large variance, we should not rely on only 4 samples per symbol and there will be an optimal number of samples based on noise levels. For this project, the default nsamp is set to 6 in order to approximate the theory curve and have better noise resistance.

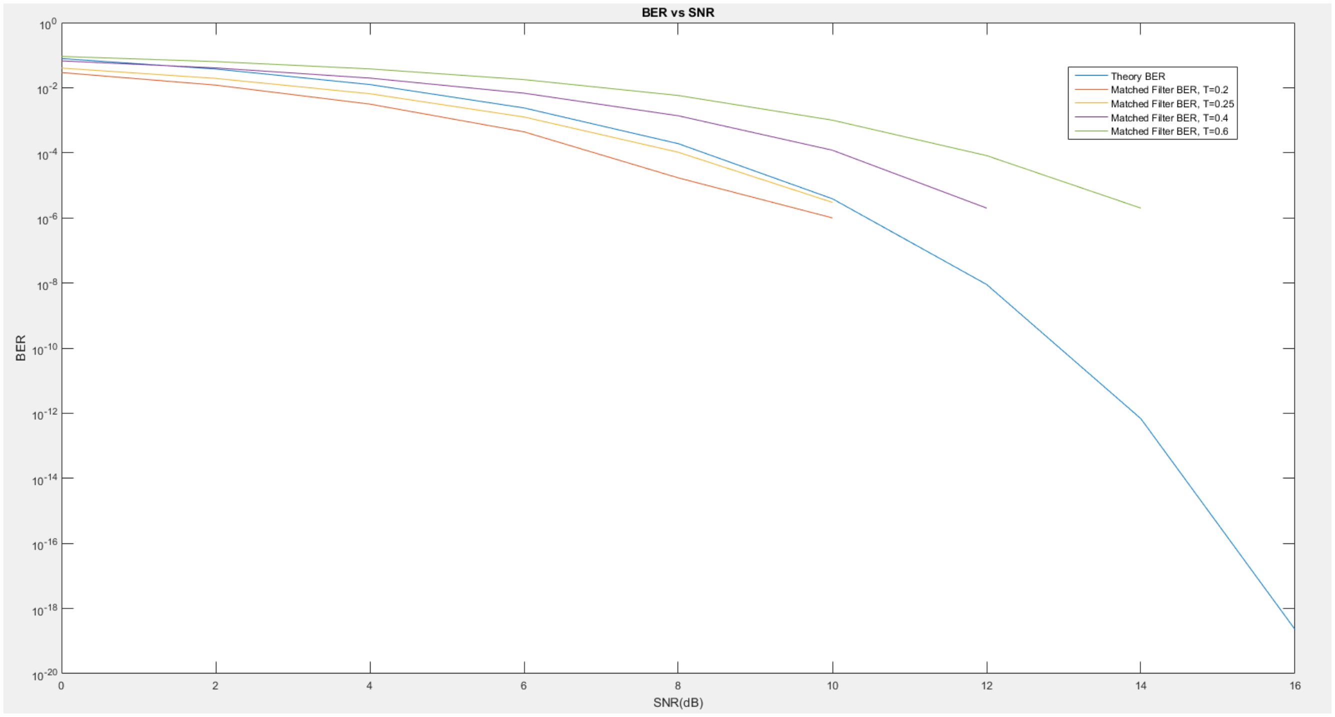

Figure 2. BER vs SNR with various symbol period (T).

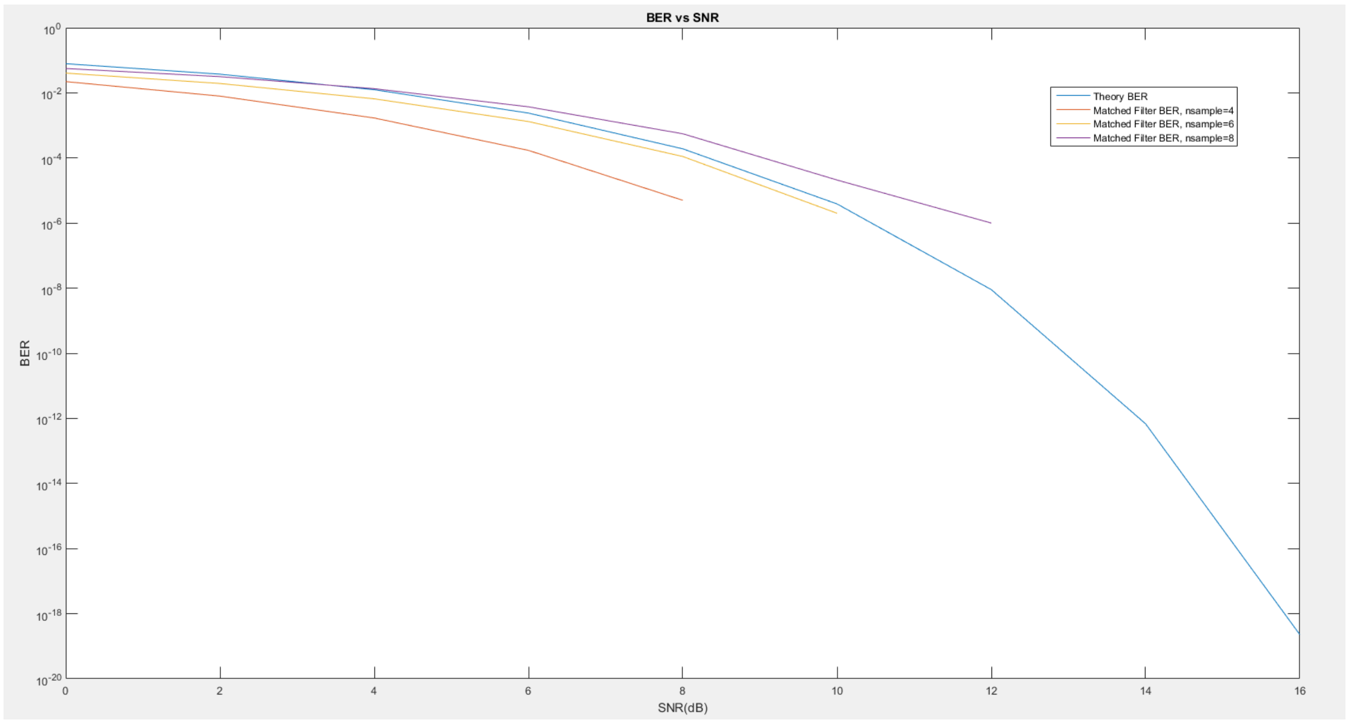

Figure 3. BER vs SNR with various numbers of samples per symbol (nsamp).

Bandwidth

Given a BER, the bandwidth efficiency is determined by the bit rate and bandwidth it consumes. With BER, we can calculate the SNR and symbol period T. The number of bits that each symbol transmitted is 1 and the symbol period is T. Therefore the bit rate is 1/T. The bandwidth of each symbol is determined by the signal pulse in our implementation. The baseband bandwidth of a rectangular pulse is from 0Hz to 1/T Hz. In passband, the bandwidth will be 2*1/T Hz. Since bandwidth efficiency = R/BW (R = bit rate). In the baseband, the bandwidth efficiency is a constant due to our binary modulation scheme. The bandwidth efficiency = 1/T (bps)/(1/T) (Hz) = 1 bit/s/Hz.

B. QPSK encoding and decoding with statistical analysis

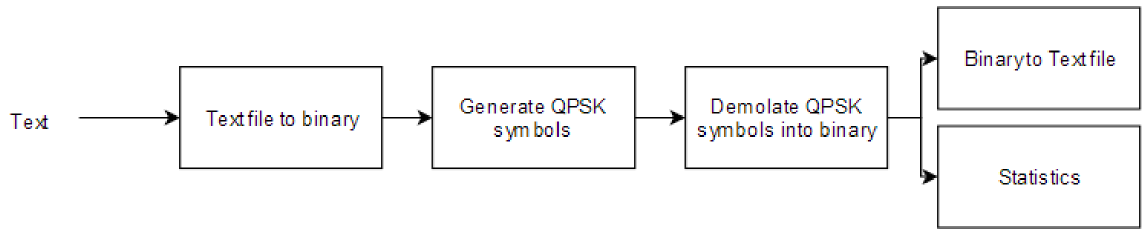

Figure 4. Block diagram of QPSK encoding and decoding scheme.

The flow of my implementation just follows the block diagram shown in Figure 4. All uppercase letters are considered as lowercase. Calculations in statistics only take letters into account. Since there will be possible noise of QPSK symbols. The QPSK symbol could be like 0.9 + j1.1. We decide 0 or 1 by checking if the real and imaginary values are greater or smaller than zero. If the binary sequence has odd length, the last QPSK symbol will have 0 imaginary value and therefore not output any bits. However, if the QPSK symbol is not the last one, even though the real or imaginary value is zero, one bit will still be decoded in order to maintain the order of the decoded binary sequence.

Figure 5 shows the letter counts and letter frequencies of testFile.txt and Table 1 is from the statistics result of my QPSK program.

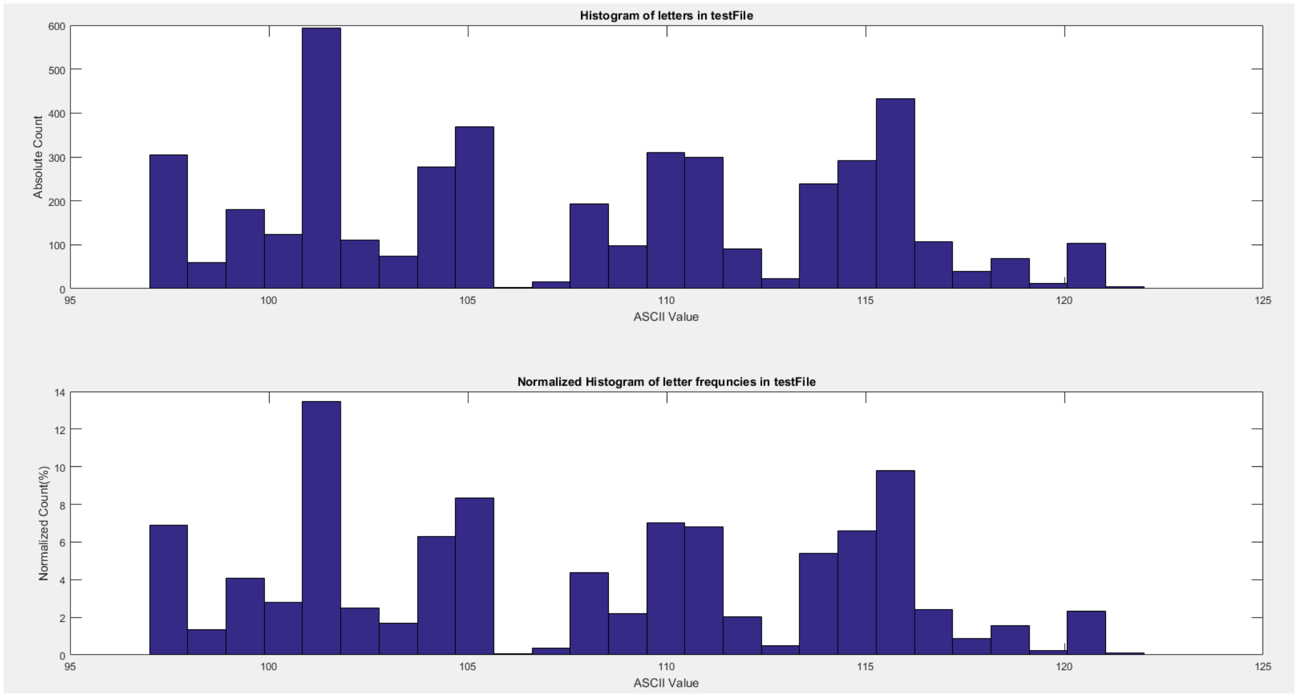

Figure 5. Histogram of letters and letter frequencies in testFile.

TABLE I

Letter frequencies of TestFile

Letters	Frequencies (%)
a	6.905139
b	1.335748
c	4.075164
d	2.807335
e	13.44804
f	2.490378
g	1.697985
h	6.293865
i	8.331447
j	0.04528
k	0.362237
l	4.369482
m	2.2187
n	7.018338
o	6.79194
p	2.037582
q	0.498076
r	5.388273
s	6.588182
t	9.780394
u	2.422459
v	0.882952
w	1.539506
x	0.249038
y	2.331899
z	0.090559