EE512 Project 1 (5% of grade): Three bit Binary $\pi/8$ Phase Shift Keying with BandPass Channel and Additive White Gaussian Noise: Modulation and Demodulation

by Laurence G. Hassebrook
4-3-2015, updated 4-8-2015

Given the MATLAB simulation structure used in V6 and the same channelxx.m file, the student is asked to create a 3 bit BPSK modulator and demodulator. The student is graded on the amount of noise used in the channel for 8192 bits in each sequence. The student will need to email the instructor the noise level to be used in the channelxx.m program. The system must send and receive with 0 error. NO WRITEUP is required but there are two demodulator designs. Use your student alias that you used in V6 but for the first version, make the alias end in A and in the second version, end your alias in B. The instructor will run your code and if it does not have any errors, the instructor will accept it and post the results using the alias. Remember Nseq=3.

I would also like to see the figures come up analogous to the ones in V6 for the modulator and demodulator. Figure 12 in the demodulator should show the output of the in-phase and quadrature mixers as shown below for part A at noiseCoef=0.01:

![In-Phase Signal](image1)

![Quadrature Signal](image2)

Figure 1: In-phase and quadrature phase response before decoding.
A. Modulation for 3 Bit PSK

The 3 bit modulation is done differently than the 2 bit system in V6. The phase shifting function is based on a weighted sum of the 3 bit sequences such that

\[ m(t) = 4b_3(t) + 2b_2(t) + b_1(t) \]  \hspace{1cm} (1)

Where the message signal \( m(t) \) has values \{0,1,2,3,4,5,6,7\}. The message signal modulates the phase from 0 to 2\( \pi \) such that

\[ s(t) = \cos(2\pi f_c t - 2\pi m(t)/8) \]  \hspace{1cm} (2)

B. Demodulator based on V6 demodulator

The student should use the in-phase and quadrature mixers used in V6 but followed by a phase detector based on \( \text{atan2}(y,x) \). The phase angle will indicate the bit sequence and bit value that is detected.

Figure 2: (left) Quadrature Mixer. (right) Phase detector.

Maximize the noiseCoef in channel...m program such that the result still has zero probability of error. Indicate studentname by ending the student’s alias with “A”. The decoding of the detected phase requires the consideration of the signal space constellation as shown below.
The phase detector output is \(-\pi\) to \(\pi\) where the 0 value is centered at 0 phase. So the 0 values is any phase value between \(-\pi/8\) to \(\pi/8\). So if we change the negative phase values such that if a phase value is less than \(-\pi/8\) we add \(2\pi\) to the value. Then if we multiply the result by \(8\) and divide by \(2\pi\) and floor the result, we will get values 0, 1,\ldots,7 which are the decoded number values of the phase. Anytime you are converting from decimal values to integer values, you may also need to add 0.5 in order to get the nearest integer value mapped by the floor function. Using the MATLAB “find” function to locate values of 0 through 7, we can encode the 3 bit streams. For example: \(J1=\text{find}(\text{theta}==1)\). The encoding of the 3xN sequence matrix, is done by mapping the find indices such as: \(Bs(3,J1)=0; Bs(2,J1)=0; Bs(1,J1)=1\); encodes 001 for the number 1 symbol.

**C. Demodulator based on Pulse detector**

The student should take the demodulator sequences from part A based on mixer cutoffs equal to the channel cutoff frequencies and correlate with a rectangular pulse of 1 bit length. To make sure the peak of the correlation corresponds to the bitcheck location, the pulse should be centered at the origin as the function \text{irect()} does.
The pulse length should be the closest odd number to the bit length. This way, it will be symmetric when used in \texttt{irect()} and the correlation will give a peak value at the bit location centers where the bitcheck is defined. Maximize the \texttt{noiseCoef} in \texttt{channel.m} such that the result still has zero probability of error. Indicate studentname by ending the student’s alias with “B” and send the \texttt{noiseCoef} value that the system should be run at.

Sample Code for pulse Correlation:

```matlab
b=[0,0,1,0];
u=ones(1,64);
btt=kron(b,u);
noise=1*(rand(size(bt))-0.5);
size(noise)
s=bt+noise;
h=irect(1,63,1,256);
figure(1);
plot(h);
axis([1,256,-0.1,1.1]);
figure(2);
n=1:256;
plot(n,bt,n,s);
figure(3);
y=real(ifft(fft(h).*fft(s)));
plot(y);
```