ENEL 529 - Lab 4

Kyle Derby MacInnis - 10053959

November 12, 2014

Abstract

The purpose of this lab is to explore how MODulation and DEModulation (MODEM) is affected by additive white Gaussian noise (AWGN) and Rayleigh fading within wireless channels, and to look at the statistical performance of MODEM techniques as they pertain to bit error rates.

Exercise Preambles

Preamble Objective

The objective of the preamble section is to prepare a random bit stream to act as our data to be transmitted, as well as the system parameters we will be assuming in this lab. The following steps apply to the upcoming exercises being explored in the lab:

1. Setup the # of bits to be transmitted and an empty bit vector:

```
%% -------- ENEL 529 - Lab 4 -------- %%
%
% Author:    Kyle Derby MacInnis
% Date:      November 3, 2014
%
% -------------------------------------- %

% Clear all System Variables
clear all;

%% Step 0 - Preambles
% # of bits
M = 1000;
% Setup Empty Bit Stream Vector
b = zeros(1,M);
```

2. Setup the bit rate period for 0.125ms:

```
% Setup Bit Rate Period (0.125 ms)
Tb = 0.125 * 10^(-3);
```

3. Setup the carrier frequency to be 10 kHz:

```
% Setup the Carrier Frequency (10 KHz)
fc = 10*10^(3);
```

4. Setup the # of Samples per bit interval, and the corresponding sample time:

```
% Setup # of samples in one bit interval
N = 15;
% Calculate the Sample Time
dt = Tb/N;
```

Exercise 1

Exercise Objective

The objective of this exercise is to simulate BPSK modulation over an AWGN channel using a perfect coherent received.

Procedure

1. The first step in this exercise is to generate a random bit stream to act as our data. The following generates random 1's or 0's, which in the second part are then converted to 1's and -1's (our digital data):

```
%% Step 1 - Generate Random Bit Stream
% Setup Random Binary Data Stream (0's and 1's)
binary_data = rand(1,M) > 0.5;
% Convert to Data Stream (-1's and 1's)
b = 2.*binary_data - 1;
```

2. The next phase of the exercise is to generate the BPSK transmission waveform using the generated data stream:

```
%% Step 2 - Generate the BPSK Transmitted Waveform
% Setup empty Signal Vector
s_i = zeros(1, M*N);
% Setup Energy of Symbol
E = Tb/2;
% Loop through Time Index, n (1-15000)
for i = 1:M
    for j = 1:N
        % Calculate Time Index, n
        n = i*N - (N-j);
        % Calculate the BPSK Signal at Time index n
        s_i(1, n) = b(1, i) * sqrt((2*E)/Tb)*cos(2*pi*fc*n*dt);
    end
end
```

3. Next, we generate the received signal under the assumption that AWGN is present in the channel. For the first part of this exercise, the AWGN is set to have a mean of 0, and a Variance of 0.01:

```
%% Step 3 - Generate the Received Signal Waveform
% Create empty Signal Vector
r = zeros(1, M*N);
% Loop through Time Index, n
for i = 1:M
    for j = 1:N
        % Calculate the Time Index, n
        n = i*N - (N-j);
        % Calculate the Received Signal at Time, n
        r(1,n) = s_i(1, n) + normrnd(0, 0.1);
    end
end
```

4. Following the generation of the received signal, the signal is then normalized against its maximum value and stored separately:

```
%% Step 4 - Normalize the Received Signal Vector
% Generate Empty Vector
r_n = zeros(1, M*N);
% Calculate the Max of the Received Signal
r_max = max(abs(r));
% Normalize Vector
for i=1:M*N
    r_n(1,i) = r(1,i)/r_max;
end

% Plot Normalized Received Signal
t = dt*(1:M*N);
figure(2);
plot(t, r_n(1,:), 'r-');
xlabel('Time: (sec)');
ylabel('Received Signal Envelope');
title('Normalized Received Signal Envelope vs. Time');
```

The Normalized signal waveform generated in the simulation is as follows:

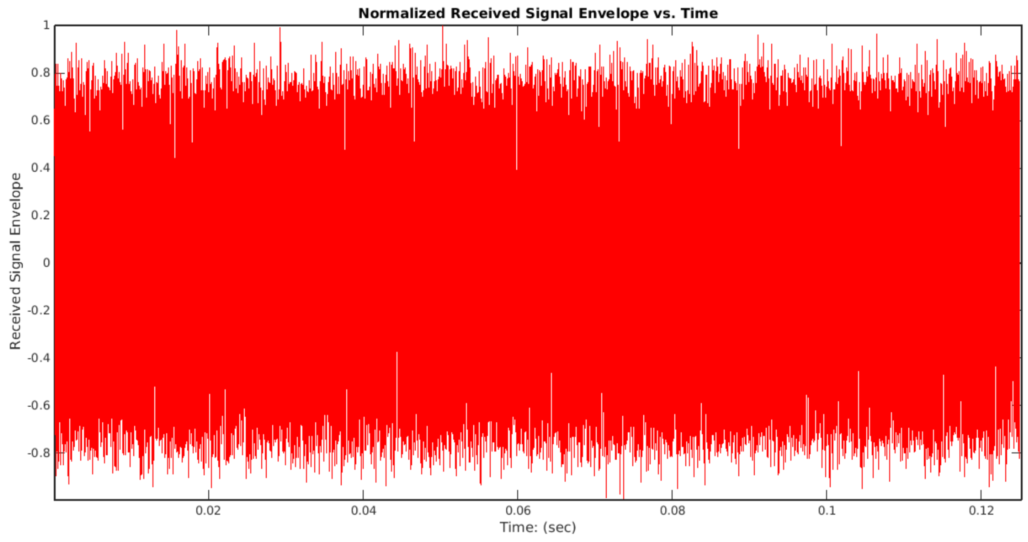

Figure 1: The graph shows the normalized BPSK signal after it has been subject to AWGN from the simulated channel.

5. For this first exercise case, the lab looks at coherent demodulation in which the MODulator and the DEModulator are in sync without any phase difference present. The next step was performing this Coherent demodulation:

```
%% ## CASE 1 - IDEAL COHERENT DETECTOR

%% Step 5 - Perform Coherent Demodulation of the Normalized Waveform
```

```
% Generate Empty Vector for Demodulated Signal
x = zeros(1, M*N);
% Loop through Time Indices , n
for i = 1:M
    for j = 1:N
        % Calculate the time Index , n
        n = i*N - (N-j);
        % Demodulate the Signal
        x(1, n) = r_n(1, n)*(2*cos(2*pi*fc*n*dt));
    end
end
```

6. Following the coherent detection, the next part of this lab passes the demodulated signal through a lowpass butterworth filter and the result is stored in another vector:

```
%% Step 6 - Perform Low-pass Filtering
% Generate Butterworth Filter Parameters
[bb, aa] = butter(10, 0.2);
% Filter Signal with Butterworth Filter
filter_output = filtfilt(bb, aa, x);
```

7. The filtered signal is then passed into a symbol detector to extract the data stream within it. This Data is then known as the received data:

```
%% Step 7 - Perform Signal Detection
% Generate Empty Signal Vector
recovered_data = zeros(1,M);
% Loop Through Symbols
for i = 1:M
    % Calculate Middle Time Index of Symbol
    k = ceil(N/2 + (i-1)*N);
    % Recover Bit Data from Signal
    if filter_output(k) > 0
        recovered_data(1,i) = 1;
    else
        recovered_data(1,i) = -1;
    end
end
```

8. Once the received data has been recovered, the original data stream is then compared with the received data stream, and all differences (errors) are calculated, and the corresponding Bit-Error Rate is computed:

```
%% Step 8 - Compare the Transmitted Signal with the Recovered Signal
% Initialize # of Errors to zero
BE = 0;
% Loop and Compare Bit-by-Bit
for i = 1:M
    if recovered_data(1,i) ~= b(1,i)
        BE = BE + 1;
    end
end
% Calculate the Bit Error Rate
BER = BE/M;
display('For a Standard Deviation of 0.1 ( Variance of 0.01):');
display(BER);
```

9. The final step of this first exercise was then to repeat the above mentioned steps, but this time to adjust the Variance of the AWGN for three different values (see Step 3):

   (a) For a Variance of 0.25:

   ```
   %% Step 3 - Generate the Received Signal Waveform
   % Create empty Signal Vector
   r = zeros(1, M*N);
   % Loop through Time Index, n
   for i = 1:M
       for j = 1:N
           % Calculate the Time Index, n
           n = i*N - (N-j);
           % Calculate the Received Signal at Time, n
           r(1,n) = s_i(1, n) + normrnd(0, 0.5);
       end
   end
   ```

   (b) For a Variance of 1.00:

   ```
   %% Step 3 - Generate the Received Signal Waveform
   % Create empty Signal Vector
   r = zeros(1, M*N);
   % Loop through Time Index, n
   for i = 1:M
       for j = 1:N
           % Calculate the Time Index, n
           n = i*N - (N-j);
           % Calculate the Received Signal at Time, n
           r(1,n) = s_i(1, n) + normrnd(0, 1.0);
       end
   end
   ```

   (c) For a Variance of 5.00:

   ```
   %% Step 3 - Generate the Received Signal Waveform
   % Create empty Signal Vector
   r = zeros(1, M*N);
   % Loop through Time Index, n
   for i = 1:M
       for j = 1:N
           % Calculate the Time Index, n
           n = i*N - (N-j);
           % Calculate the Received Signal at Time, n
           r(1,n) = s_i(1, n) + normrnd(0, sqrt(5.0));
       end
   end
   ```

Results

Following the execution of this exercise, and adjusting the values of the AWGN accordingly the following Bit Error Rates were calculated along with their corresponding AWGN Variances:

1. AWGN Variance: 0.01 $\rightarrow$ BER: 0
2. AWGN Variance: 0.25 $\rightarrow$ BER: 0.0020
3. AWGN Variance: 1.00 $\rightarrow$ BER: 0.0640
4. AWGN Variance: 5.00 $\rightarrow$ BER: 0.2480

Exercise 2

Exercise Objective

The objective of this exercise is to explore the effects of imperfect coherent detection whereby the detector has a constant phase error present within it.

Procedure

1. Please see exercise 1 - Step 1
2. Please see exercise 1 - Step 2
3. Please see exercise 1 - Step 3 with the small variation on the AWGN parameters. For this exercise the variance is set to 1.00.
4. Please see exercise 1 - Step 4
5. Whereas previously in exercise 1, the case being explored was that of ideal perfect coherent detection, in this exercise the case being explored is that of imperfect coherent detection with a constant phase error present in the detector. The constant phase error is initially set to 5.0°:

```
%% ## CASE 2 - IMPERFECT COHERENT DETECTOR (CONSTANT ERR)

%% Step 10 - Repeat Steps 3-8 Using Constant Phase Error
% Generate Empty Vector for Demodulated Signal
x = zeros(1, M*N);
% Generate Phase Error
phaseErr = (5/180)*pi;
% Loop through Time Indices, n
for i = 1:M
    for j = 1:N
        % Calculate the time Index, n
        n = i*N - (N-j);
        % Demodulate the Signal
        x(1, n) = r_n(1, n)*(2*cos(2*pi*fc*n*dt + phaseErr));
    end
end
```

6. Please see exercise 1 - Step 6
7. Please see exercise 1 - Step 7
8. Please see exercise 1 - Step 8
9. For this exercise, rather than varying the AWGN variance, the exercise was instead repeated by varying the phase error present in the detector. In addition to the original 5° error, the following errors were also tested:

   (a) Phase Error of 35°

```
%% Step 10 - Repeat Steps 3-8 Using Constant Phase Error
% Generate Empty Vector for Demodulated Signal
x = zeros(1, M*N);
% Generate Phase Error
phaseErr = (35/180)*pi;
% Loop through Time Indices, n
for i = 1:M
    for j = 1:N
        % Calculate the time Index, n
        n = i*N - (N-j);
        % Demodulate the Signal
        x(1, n) = r_n(1, n)*(2*cos(2*pi*fc*n*dt + phaseErr));
    end
end
```

(b) Phase Error of 85°

```
%% Step 10 - Repeat Steps 3-8 Using Constant Phase Error
% Generate Empty Vector for Demodulated Signal
x = zeros(1, M*N);
% Generate Phase Error
phaseErr = (85/180)*pi;
% Loop through Time Indices, n
for i = 1:M
    for j = 1:N
        % Calculate the time Index, n
        n = i*N - (N-j);
        % Demodulate the Signal
        x(1, n) = r_n(1, n)*(2*cos(2*pi*fc*n*dt + phaseErr));
    end
end
```

Results

Following the execution of this exercise, and adjusting the values of the phase error accordingly the following Bit Error Rates were calculated using an AWGN variance of 1.00 along with their corresponding phase errors:

1. Phase Error: 5° → BER: 0.0550
2. Phase Error: 35° → BER: 0.0760
3. Phase Error: 85° → BER: 0.4390

Exercise 3

Exercise Objective

The objective of this exercise is to explored the effects of having an imperfect coherent detector whereby there is a random phase error present.

Procedure

1. Please see exercise 1 - Step 1
2. Please see exercise 1 - Step 2
3. Please see exercise 1 - Step 3 with the small variation on the AWGN parameters. For this exercise the variance is set to 1.00.
4. Please see exercise 1 - Step 4
5. Whereas previously in exercise 2, the detector used was designed to simulate having a constant phase error, this exercise looks at having a random phase error present in respect to time. The original setup for this random phase error is that of a Gaussian random variable with a mean of 0° and a standard deviation of 0°:

```
%% ## CASE 3 - IMPERFECT COHERENT DETECTOR (RANDOM ERR)

%% Step 10 - Repeat Steps 3-8 Using Random Phase Error
% Generate Empty Vector for Demodulated Signal
x = zeros(1, M*N);
% Loop through Time Indices, n
for i = 1:M
    for j = 1:N
        % Calculate the time Index, n
        n = i*N - (N-j);
        % Generate Random Phase Error
        phaseErr = normrnd(0,(0/180)*pi);
        % Demodulate the Signal
        x(1, n) = r_n(1, n)*(2*cos(2*pi*fc*n*dt + phaseErr));
    end
end
```

6. Please see exercise 1 - Step 6
7. Please see exercise 1 - Step 7
8. Please see exercise 1 - Step 8
9. For this exercise, rather than varying a constant phase error, the exercise was instead repeated by varying the phase error present in the detector by adjusting the variance of the random variable. In addition to the original 0° standard deviation, the following errors were also tested:

   (a) Phase Error Std. Dev. of 10°

```
%% ## CASE 3 - IMPERFECT COHERENT DETECTOR (RANDOM ERR)

%% Step 10 - Repeat Steps 3-8 Using Random Phase Error
% Generate Empty Vector for Demodulated Signal
x = zeros(1, M*N);
% Loop through Time Indices, n
for i = 1:M
    for j = 1:N
        % Calculate the time Index, n
        n = i*N - (N-j);
        % Generate Random Phase Error
        phaseErr = normrnd(0,(10/180)*pi);
        % Demodulate the Signal
        x(1, n) = r_n(1, n)*(2*cos(2*pi*fc*n*dt + phaseErr));
    end
end
```

   (b) Phase Error Std. Dev. of 20°

```
%% ## CASE 3 - IMPERFECT COHERENT DETECTOR (RANDOM ERR)

%% Step 10 - Repeat Steps 3-8 Using Random Phase Error
% Generate Empty Vector for Demodulated Signal
x = zeros(1, M*N);
% Loop through Time Indices, n
for i = 1:M
    for j = 1:N
        % Calculate the time Index, n
        n = i*N - (N-j);
        % Generate Random Phase Error
        phaseErr = normrnd(0,(20/180)*pi);
        % Demodulate the Signal
        x(1, n) = r_n(1, n)*(2*cos(2*pi*fc*n*dt + phaseErr));
    end
end
```

(c) Phase Error Std. Dev. of 30°

```
%% ## CASE 3 - IMPERFECT COHERENT DETECTOR (RANDOM ERR)

%% Step 10 - Repeat Steps 3-8 Using Random Phase Error
% Generate Empty Vector for Demodulated Signal
x = zeros(1, M*N);
% Loop through Time Indices, n
for i = 1:M
    for j = 1:N
        % Calculate the time Index, n
        n = i*N - (N-j);
        % Generate Random Phase Error
        phaseErr = normrnd(0,(30/180)*pi);
        % Demodulate the Signal
        x(1, n) = r_n(1, n)*(2*cos(2*pi*fc*n*dt + phaseErr));
    end
end
```

Results

Following the execution of this exercise, and adjusting the values of the phase error Std. Dev. accordingly the following Bit Error Rates were calculated using an AWGN variance of 1.00 along with their corresponding phase errors parameters:

1. Phase Error Std. Dev.: 0° → BER: 0.0440
2. Phase Error Std. Dev.: 10° → BER: 0.0600
3. Phase Error Std. Dev.: 20° → BER: 0.0630
4. Phase Error Std. Dev.: 30° → BER: 0.0730

Exercise 4

Exercise Objective

The objective of this exercise is to explore the effects of Rayleigh fading on MODulation and DEModulation techniques and the effect that it has on bit error rate.

Procedure

1. See Exercise 1 - Step 1
2. For this exercise, we are going to assume that the phase of the signal is shifted uniformly between 0 and 2π due the presence of multipath:

```
%% Step 2 - Generate the BPSK Transmitted Waveform (Rayleigh)
% Setup empty Signal Vector
s_i = zeros(1, M*N);
% Setup Energy of Symbol
E = Tb/2;
% Loop through Time Index, n (1-15000)
for i = 1:M
    for j = 1:N
```

```
        % Calculate Time Index, n
        n = i*N - (N-j);
        % Calculate the BPSK Signal at Time index n with Uniform Random Phase Shift
        s_i(1, n) = b(1, i) * sqrt((2*E)/Tb)*cos(2*pi*fc*n*dt + random('uniform', 0, 2*
    pi));
    end
end
```

3. Whereas the previous exercises have worked under the principle that only AWGN is present alongside the transmitted signal, this exercise looks at how Rayleigh fading can affect transmission quality and MODEM techniques. For this, the variance of the AWGN is originally set to 0.01.

```
%% ## CASE 4 - PERFECT COHERENT DETECTOR (RAYLEIGH FADING)

%% Step 3 - Generate the Received Signal Waveform (Rayleigh)
% Create empty Signal Vector
r = zeros(1, M*N);
% Loop through Time Index, n
for i = 1:M
    for j = 1:N
        % Calculate the Time Index, n
        n = i*N - (N-j);
        % Calculate the Received Signal at Time, n
        r(1,n) = raylrnd(1)*s_i(1, n) + normrnd(0, 0.1);
    end
end
```

4. Please see exercise 1 - Step 4 with the minor addition that this is now a Rayleigh faded signal, and the following plot was generated from the calculations:

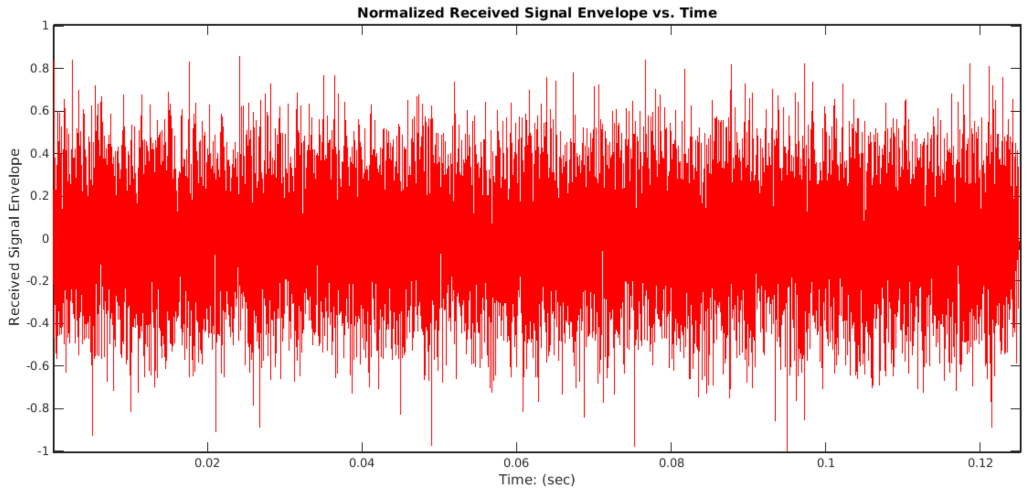

Figure 2: The graph shows the normalized BPSK signal under Rayleigh fading conditions, along with AWGN present in the channel.

5. Please see exercise 1 - Step 5

6. Please see exercise 1 - Step 6

7. Please see exercise 1 - Step 7

8. Please see exercise 1 - Step 8

9. For this exercise, the test was repeated by varying the Variance of the AWGN and applying it to the Rayleigh faded signal. In addition to the original Variance of 0.01, the following values were also used in the simulation:

   (a) AWGN Variance of 0.25:

```
%% ## CASE 4 - PERFECT COHERENT DETECTOR (RAYLEIGH FADING)

%% Step 3 - Generate the Received Signal Waveform (Rayleigh)
% Create empty Signal Vector
r = zeros(1, M*N);
% Loop through Time Index, n
for i = 1:M
    for j = 1:N
        % Calculate the Time Index, n
        n = i*N - (N-j);
        % Calculate the Received Signal at Time, n
        r(1,n) = raylrnd(1)*s_i(1, n) + normrnd(0, 0.5);
    end
end
```

   (b) AWGN Variance of 1.00:

```
%% ## CASE 4 - PERFECT COHERENT DETECTOR (RAYLEIGH FADING)

%% Step 3 - Generate the Received Signal Waveform (Rayleigh)
% Create empty Signal Vector
r = zeros(1, M*N);
% Loop through Time Index, n
for i = 1:M
    for j = 1:N
        % Calculate the Time Index, n
        n = i*N - (N-j);
        % Calculate the Received Signal at Time, n
        r(1,n) = raylrnd(1)*s_i(1, n) + normrnd(0, 1.0);
    end
end
```

   (c) AWGN Variance of 5.00:

```
%% ## CASE 4 - PERFECT COHERENT DETECTOR (RAYLEIGH FADING)

%% Step 3 - Generate the Received Signal Waveform (Rayleigh)
% Create empty Signal Vector
r = zeros(1, M*N);
% Loop through Time Index, n
for i = 1:M
    for j = 1:N
        % Calculate the Time Index, n
        n = i*N - (N-j);
        % Calculate the Received Signal at Time, n
        r(1,n) = raylrnd(1)*s_i(1, n) + normrnd(0, sqrt(5.0));
    end
end
```

Results

Following the execution of this exercise and adjusting the values of the AWGN accordingly, the following Bit Error Rates were calculated along with their corresponding AWGN Variances for the Rayleigh faded channel:

1. AWGN Variance: 0.01 $\rightarrow$ BER: 0.4630
2. AWGN Variance: 0.25 $\rightarrow$ BER: 0.5430
3. AWGN Variance: 1.00 $\rightarrow$ BER: 0.5000
4. AWGN Variance: 5.00 $\rightarrow$ BER: 0.4820

Additional Questions and Exercises

1.1 - In Case 1, what is the effect of the AWGN variance on the simulated bit error rate?

As the AWGN variance was increased, the corresponding bit error rates began to increase as well. The effect of the noise on the bit error rate however, was not as extreme as one may originally think. The effect only became quite high when the variance was at its maximum value of 5.00. Even at a variance of 1.00 the bit error rate was less than 10%.

1.2 - In Case 2, what is the effect of the constant phase error on the bit error rate?

The effect of the phase error was slightly more noticeable than the effect of AWGN. For a slight phase error, there was less than 1% of the bits transmitted incorrectly, but as the value increase it began to have a greater impact. At 35° the error began to approach 10%, and once it was tested with a phase error 85° there was almost a 50% error rate. This goes to show that small phase errors are not much trouble, but as the phase error begins to approach 90° lots of data is lost.

1.3 - In Case 3, what is the effect of the Random Phase error on the bit error rate?

As far as things are concerned in this lab, having random phase errors do not cause too much trouble. Provided the phase errors are Gaussian, the overall bit error rate never seems to go past 10%, however, the variance on the phase error did little to affect the result. Having a small variance or a larger variance did little to change how much the bit error rate varied between the runs.

1.4 - In Case 4, what is the effect of the Rayleigh fading on the bit error rate?

Under Rayleigh fading conditions, the bit error rate of the system was considerably higher than other factors in the system. All of the simulated runs resulted in a bit error rate approaching 50% regardless of the noise present in the system. Even under small AWGN variance, the bit error rate was close to 40%, where as at a variance of 5.00, there was only a change of approximately an additional 10% to the bit error rate as it was approximately 50%. Another thing to note was that under the simulation, the highest BER was found at around an AWGN variance of 0.25 which resulted in a BER of over 50%. This suggests that the Rayleigh fading is the principle factor affecting the performance of the BER of the system, and that even under low noise channels, performance is greatly affected.

1.5 - Comment on the relative impacts of Rayleigh fading and imperfect coherent receivers on bit error rate performance.

After analyzing the results from the simulations, imperfect coherent receivers are going to result in some form of bit error between transmission and reception. If Rayleigh fading is also present within the system, than the resulting signal will have a much larger BER than a system without it. The next largest factor results from having large phase errors between the receiver and the transmitter MODEMs and following that, the next largest factor would be the presence of noise. To maximize performance, techniques should be implemented to minimize the effects of multipath, and then adjust for noise, as multipath fading will result in phase errors within the system.

Remarks on the Lab

The lab was completed as necessary, and following completion, an understanding of how Rayleigh fading and noise present within a system given imperfect coherent receivers can result in less-than-ideal bit error rates. The most difficult part of the lab was generating the signal such that each individual symbol interval was stored accordingly into a vector such that each individual bit could be sampled specifically using the Time index. Other than the point mentioned previously, the rest of the lab was expanded gradually from the previous lab exercises, and this allowed for the new material to be absorbed effectively. The only argument I can think to suggest is that a relationship between signal outage and bit error rate be expanded on to help cement the concepts of channel propogation and how they individually can impact the received signal.