ChanggangZheng_Laboratory_Report_2289258Z_2016200302027.pdf

by Changgang Zheng

Glasgow College, UESTC

Wireless and Optical Transmission Systems

Laboratory Report

Author: Changgang Zheng
UoG ID: 2289258Z
UESTC ID: 2016200302027
E-mail: chnaggangzheng@std.uestc.edu.cn

LABORATORY REPORT

CHANGGANG ZHENG[1]

November 19, 2019

CONTENTS

1Glasgow College, University of Electronic Science and Technology of China, ChengDu, China

LIST OF FIGURES

INTRODUCTION

This Laboratory report is about Lab report of the course Wireless and Optical Transmission Systems. The experiment includes information from generate signal, modulation [1, 2], design and add channels, and analysis system via Simulink, MATLAB toolbox by calculating the BER, plotting the constellation diagram and the relationship between BER and SNR.

The first four experiments are about using MATLAB codes to realize the environments and do the analysis. The following experiments are mainly based on the Simulink, as all the implementation and tests are on Simulink.

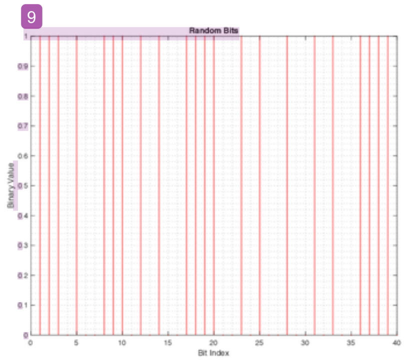
From this experiment, I understand the necessary information about how to use the MATLAB to construct communication models, includes different kinds of inputs, modulation techniques, and how we can test these models. The relationship between bit error rate and the signal to noise ratio as well as the eye-diagram are compared to evaluate the quality of the signal or the modulation techniques.

The above techniques include all the essential methods in communication system design, modulation techniques design, channel design, signal analysis, communication system evaluation techniques, and so on, which provides an excellent technical basis for wireless communication simulation and testing.

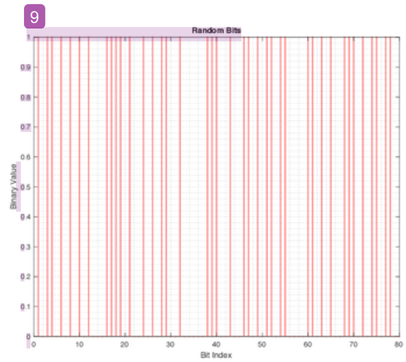
1 EXERCISE 1

1.1 Introduction

This experiment introduces basic techniques about how to use MATLAB codes to generate the random signals, Binary Data Stream for example, and implement modulation to these generated signals. Then, add the proper channel with noise. Finally, evaluate the system by using different techniques.



Random signal with length 40



Random signal with length 80

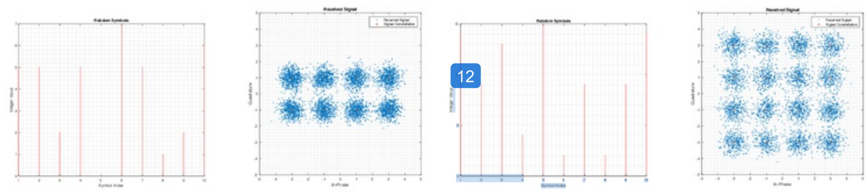
Figure 1: Random signals

1.2 System Design

First of all, Bernoulli random codes are generated to simulate the input data of the system. The random bits are generated and stored in and fixed-length array. After this, the code is used to construct symbols. Then the symbols are modulated and transmitted in a channel with wite gaussian noize. Then the received signals are extracted by the demodulator and the symbol to bit converter. Finally, compare it with the transmitting signal to calculate the total number of bit errors and bit error rate(BER).

1.3 Results

The Random binary signal is generated firstly. And the random symbols are constructed by these binary signals with a group of four. Then, those signals are modulated by either the 8-QAM or 16-QAM before going into the channel where the AWGN is added. Finally, after the signal is received, it is shown on the constellation, as shown in Figure2. Besides, the total number of errors(NER) and



Random Symbols 8 QAM Random Symbols 16 QAM

Figure 2: Random Symbols and Their Constellation

the bit error rate of the communication are calculated and displayed by using the MATLAB. The output of NER and BERm is shown below.

```
number_of_errors =

   172


bit_error_rate =

    0.0057

>>
```

1.4 Conclusion and Learning Outcomes

From this experiment, I can conclude that the MATLAB function can be used to generate Bernoulli signals and symbolize it. AWGN be added to the signal to simulate the random noise. Finally, the received symbols are displayed on a constellation diagram. We can see that the point of symbols is centered but not specifically at the Theoretical place, which is caused by the AWGN.

For the Additive white Gaussian noise, the nature of the it is random jitter around its mean value, where any sufficient length of signal would have a zero average. Plus, the variance of AWGN follows the normal distribution. To the AWGN, it has mainly three properties. Firstly, the variance of AWGN is its power. AWGN is one of two primary sources of signal deterioration (the other source is attenuation). The transmitting signal would not change its power after passing through an AWGN channel, which reduces the SNR of the transmitting signal.

1.5 Code

The code [3] is typed and revised by Changgnag Zheng.

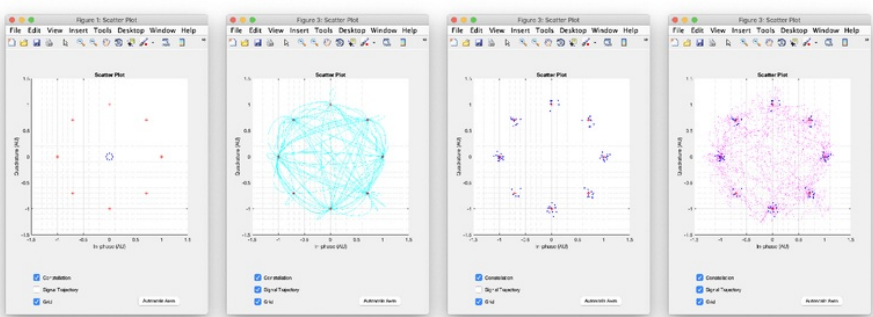
```matlab
% Type and revised by Changgnag Zheng
% A student from Glasgow College, UESTC
% UoG matrix: 2289258z UESTC matrix: 2016200302027

%% Define parameters
M = 16;
k = log2(M);
n = 3e4;
nsamp = 1;
iMin = 0;
iMax = 1;
x = randi([iMin,iMax],1,n);
EX1_plot1 = figure;
stem(x(1:40),'.','filled','LineWidth',1.5,'color',[1,0.5,0.5]);
title('Random Bits');
xlabel('Bit Index');
ylabel('Binary Value')
grid minor
set(EX1_plot1, 'PaperPosition', [0.05 0.05 9 7]);
set(EX1_plot1, 'PaperSize', [9.05 7.05]);
saveas(EX1_plot1,['EX1_plot1.pdf'],'pdf')

%% Bit to symbol mapping
xsym = bi2de(reshape(x,k,length(x)/k).','left-msb');
EX1_plot2 = figure;
stem(xsym(1:10),'.','filled','LineWidth',1.5,'color',[1,0.5,0.5]);
title('Rabdom Symbols');
xlabel('Symbol Index');
ylabel('Integer Value')
grid minor
set(EX1_plot2, 'PaperPosition', [0.05 0.05 9 7]);
set(EX1_plot2, 'PaperSize', [9.05 7.05]);
saveas(EX1_plot2,['EX1_plot2.pdf'],'pdf')

%% Modulation
y = modulate(modem.qammod(M), xsym); % using the 16 QAM

%% Transmissitted Signal
ytx = y;

%% Channel
% Send signal over an AEGN channel
EbNo = 9;
snr = EbNo + 10*log10(k) - 10*log10(nsamp);
ynoisy = awgn(ytx,snr,'measured');

%% Receive channel
yrx = ynoisy;

%% Scatter plot
h = scatterplot(yrx(1:nsamp*5e3),nsamp, 0,'.');
hold on;
EX1_plot3 = scatterplot(ytx(1:5e3),1,0,'+',h);
```

```
title('Received Signal')
legend('Received Signal','Signal Constellation');
axis([-5 5 -5 5]);
grid minor
set(EX1_plot3, 'PaperPosition', [0.05 0.05 9 7]);
set(EX1_plot3, 'PaperSize', [9.05 7.05]);
saveas(EX1_plot3,['EX1_plot3.pdf'],'pdf')
hold off;

%% Demodulation
% Demodulate signal using 16-QAM.
zsym = demodulate(modem.qamdemod(M), yrx);

%% Symbol-to-Bit Mapping
% Undo the bit-to-symbol mapping performed earlier.
z = de2bi(zsym, 'left-msb'); % Convert integers to bits.
% Convert z from a matrix to a vector.
z = reshape(z.', prod(size(z)),1);

%% BER Computation
% Compare x and z to obtain the number of errors and the bit
% error rate.
[number_of_errors, bit_error_rate] = biterr(x, z.')
```

2 EXERCISE 2

2.1 Introduction

This experiment aims to display the transmitting data, and to receive data on a constellation diagram. The theoretical position of each star is also plotted on the diagram. The AWGN is also added to the channel.

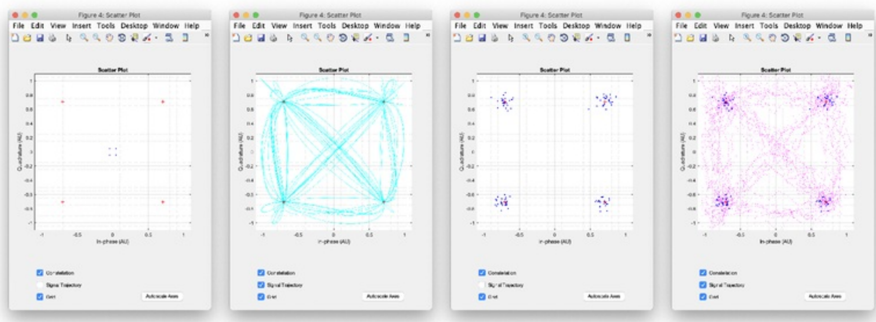


Ideal Constellation | Trajectory | Results | Result Trajectory

Figure 3: Scatter Plots of 8-PSK

2.2 System Design

At the very beginning, the modulation of Q-PSK is defined. After which, the already defined modulator process the data, which are random digits, and send them to the channel. When passing through the channel, AWGN is added to the signal. Finally, after data are handled by the receiver, the data position is plotted on a constellation diagram where the ideal position of each point is displayed.



Ideal Constellation   Trajectory   Results   Results Trajectory

Figure 4: Scatter Plots of Q-PSK

2.3 Results

The above two figures with eight subfigures shows the ideal constellation. Then, the moving trajectory of received data are plotted on the constellation diagram. Meanwhile, the received data point is finally added to the constellation diagram.

2.4 Conclusion and Learning Outcomes

From this experiment, we can conclude that the MATLAB functions can be used to generate the ideal constellation diagram and the received signal. These signals are distorted and attenuated due to the transmission channel. Due to the same reason, there are jitters on the signal trajectory. Finally, the received signal is not at the theoretical position on the diagram but has a slight shift.

2.5 Code

The code [3] is typed and revised by Changgnag Zheng.

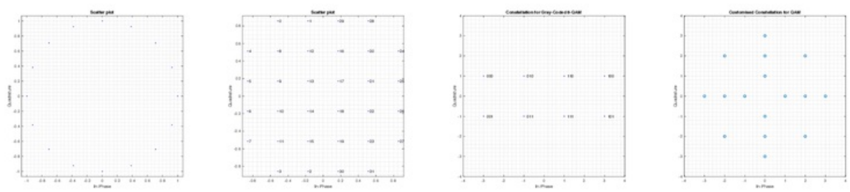
```
% Type and revised by Changgnag Zheng
```

```
% A student from Glasgow College, UESTC
% UoG matrix: 2289258z UESTC matrix: 2016200302027

%% create a QPSK modulator object.
hMod = modem.pskmod('M', 8, 'PhaseOffset', pi/4);
%% create an upsampling filter
Rup = 16; % up sampling rate
hFilDesign = fdesign.pulseshaping(Rup, 'Raised Cosine','Nsym,Beta', ...
    Rup,0.50);
hFil = design(hFilDesign);
%% create the transimit signal
d = randi([0 hMod.M-1], 100, 1); % Generate data symbols
sym = modulate(hMod, d); % Generate modulated symbols
xmt = filter(hFil, upsample(sym, Rup));
%% create a scatter plot and set the samples per symbol to ghe ...
    upsampling rate
hScope = commscope.ScatterPlot
grid minor
hScope.SamplesPerSymbol = Rup;
%% set the constellation value
hScope.Constellation = hMod.Constellation;
%% groupdelay
groupDelay = (hFilDesign.NumberOfSymbols/2);
hScope.MeasurementDelay = groupDelay /hScope.SymbolRate;
update(hScope, xmt)
hScope.PlotSettings.Constellation = 'on';
%%
hFil.Numerator = hFil.Numerator / max(hFil.Numerator);
%%
xmt = filter(hFil, upsample(sym, Rup));
%%
reset(hScope)
update(hScope, xmt)
hScope.PlotSettings.SignalTrajectory = 'on';
hScope.PlotSettings.SignalTrajectoryStyle = ':m';
autoscale(hScope)
rcv = awgn(xmt, 20, 'measured'); % Add AWGN
%%
reset(hScope)
update(hScope, rcv)
hScope.PlotSettings.SignalTrajectory = 'off';
hScope.PlotSettings.Constellation = 'on';
hold off;
```

3 EXERCISE 3

3.1 Introduction

This experiment requires us to display the theoretical constellation includes16-PSK, 32-QAM, and 8-QAM. The experiment also requires us to design and plot the new type of constellation pattern.

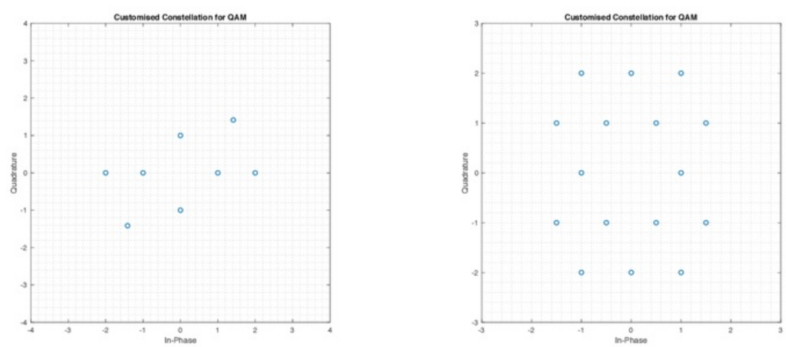


16-PSK 32-QAM 8-QAM Special Pattern

Figure 5: Constellations

3.2 System Design

First of all, use the scatterplot to display the constellation of 16-PSK, 32-QAM, and 8-QAM. Similarly, the additional selected type of constellation pattern is required to be plotted. To doing this, some vectors are generated to store the position of each point in the new constellation. Then, use the MATLAB function to plot it just as the traditional ones.



New Constellation Type 1 New Constellation Type 2

Figure 6: New Constellations

3.3 Results

Some of the commonly used constellation diagram is plotted, as shown in Figure 5. The final sub-figure, a new type of constellation diagram, is also designed and displayed. Besides, Based on the instruction of the lab manual, the other two type of the constellation diagram is also plotted in Figure 6.

3.4 Conclusion and Learning Outcomes

This experiment shows how to use MATLAB to construct constellation diagrams. Including the already existing types, new types of diagrams can also be used and analyzed by MATLAB. More than that, the quality of new constellation can also be calculated by using other MATLAB tools.

3.5 Code

The code [3] is typed and revised by Changgnag Zheng.

```
% Type and revised by Changgnag Zheng
% A student from Glasgow College, UESTC
% UoG matrix: 2289258z UESTC matrix: 2016200302027

% input bertool in the commond line
% constellation for 16-psk
M = 16;
x = [0:M-1];
EX3_plot1 = scatterplot(modulate(modem.pskmod(M), x));
grid minor
set(EX3_plot1, 'PaperPosition', [0.05 0.05 9 7]);
set(EX3_plot1, 'PaperSize', [9.05 7.05]);
saveas(EX3_plot1,['EX3_plot1.pdf'],'pdf')

% constellation for 32-QAM
M = 32;
x = [0:M-1];
y = modulate(modem.qammod(M), x);
scale = modnorm(y, 'peakpow', 1);
y = scale*y; % Scale the constellation.
EX3_plot2 = scatterplot(y); % Plot the scaled constellation.
% Include text annotations that number the points.
hold on; % Make sure the annotations go in the same figure.
for jj=1:length(y)
text(real(y(jj)), imag(y(jj)), [' ' num2str(jj-1)]);
end
hold off;
grid minor
set(EX3_plot2, 'PaperPosition', [0.05 0.05 9 7]);
set(EX3_plot2, 'PaperSize', [9.05 7.05]);
```

```
saveas(EX3_plot2,['EX3_plot2.pdf'],'pdf')

%% gray-coded signal constellation
M = 8;
x = [0:M-1];
y = modulate(modem.qammod('M', M, 'SymbolOrder', 'Gray'), x);
% Plot the Gray-coded constellation.
EX3_plot3 = scatterplot(y, 1, 0, 'b.'); % Dots for points.
% Include text annotations that number the points in binary.
hold on; % Make sure the annotations go in the same figure.
annot = dec2bin([0:length(y)-1], log2(M));
text(real(y)+0.15, imag(y), annot);
axis([-4 4 -4 4]);
title('Constellation for Gray-Coded 8-QAM');
hold off;
grid minor
set(EX3_plot3, 'PaperPosition', [0.05 0.05 9 7]);
set(EX3_plot3, 'PaperSize', [9.05 7.05]);
saveas(EX3_plot3,['EX3_plot3.pdf'],'pdf')

%% customised constellation 1 for QAM
% Describe constellation.
inphase = [0 -2 0 2 0 -3 -2 -1];
quadr = [3 2 2 2 1 0 0 0];
inphase = [inphase; -inphase];
inphase = inphase(:);
quadr = [quadr; -quadr];
quadr = quadr(:);
const = inphase + j*quadr;
% Plot constellation.
EX3_plot4 = scatterplot(const, 1, 0, 'o');
hold on;
axis([-4 4 -4 4]);
title('Customised Constellation for QAM');
hold off;
grid minor
set(EX3_plot4, 'PaperPosition', [0.05 0.05 9 7]);
set(EX3_plot4, 'PaperSize', [9.05 7.05]);
saveas(EX3_plot4,['EX3_plot4.pdf'],'pdf')

%% customised constellation 2 for QAM
inphase = [0 1 2 sqrt(2)];
quadr = [1 0 0 sqrt(2)];
inphase = [inphase; -inphase];
inphase = inphase(:);
quadr = [quadr; -quadr];
quadr = quadr(:);
const = inphase + j*quadr;
% Plot constellation.
EX3_plot5 = scatterplot(const, 1, 0, 'o');
hold on;
axis([-4 4 -4 4]);
title('Customised Constellation for QAM');
```

```
hold off;
grid minor
set(EX3_plot5, 'PaperPosition', [0.05 0.05 9 7]);
set(EX3_plot5, 'PaperSize', [9.05 7.05]);
saveas(EX3_plot5,['EX3_plot5.pdf'],'pdf')

% Manual constellation.
inphase = [1/2 -1/2 1 0 3/2 -3/2 1 -1];
quadr = [1 1 0 2 1 1 2 2];
inphase = [inphase; -inphase]; inphase = inphase(:);
quadr = [quadr; -quadr]; quadr = quadr(:);
const = inphase + 1i*quadr;
% Plot constellation.
EX3_plot6 = scatterplot(const, 1, 0, 'o');
hold on;
axis([-3 3 -3 3]);
title('Customised Constellation for QAM');
hold off;
grid minor
set(EX3_plot6, 'PaperPosition', [0.05 0.05 9 7]);
set(EX3_plot6, 'PaperSize', [9.05 7.05]);
saveas(EX3_plot6,['EX3_plot6.pdf'],'pdf')
```

4 EXERCISE 4

4.1 Introduction

This experiment requires us to use BERTool to simulate the connection between the signal to noise ratio (SNR) and the bit error rate (BER) of any selected communication system. The simulated result of each system is also compared with the ideal situation. The results are as shown as follows.

4.2 System Design

First of all, writing codes in MATLAB to construct the model of communication. For the system, the size of constellation, number of bits per symbol, number of bits to process, oversampling rate, and other relating parameters are assigned to the system. After which, the model is evaluated by the BERTool in the following steps. After all the parameters are assigned properly, Use the BERTool to simulated the above model. Then operate the tool to plot the ideal result of the relation between the BER and the SNR. The result are drawn and compared in figures.

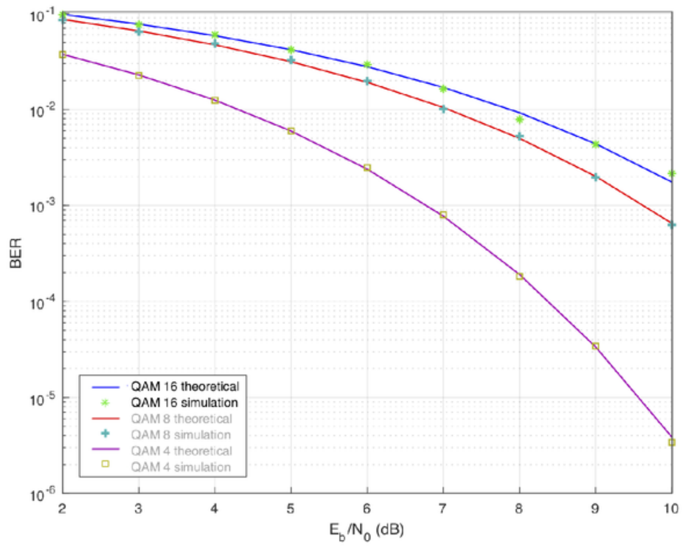


Figure 7: Simulation and Theoretical Results

4.3 Results

The bit error rates and the signal to noise ratio relationship of 4-QAM, 8-QAM, and 16-QAM is displayed as Figure 7. The Simulated and the ideal curve are all plotted on it.

4.4 Conclusion and Learning Outcomes

Form Figure 7, we can see that the 4-QAM technology is better than the 8-QAM. 16-QAM is the worst modulation techniques under this simulation environment. The reason is if we see these three lines vertically, which means we see the BER in the same SNR value, the 4-QAM's error rate is the lowest one, the second-lowest would be the 8-QAM and16-QAM has the highest BER. This implies that the 4-QAM has the lowest amount of errors when the signal power is very similar. By using this method, we can use this kind of relationship to evaluate the power efficiency of the modulation techniques.

4.5 Code

The code [3] is typed and revised by Changgnag Zheng.

```
% Type and revised by Changgnag Zheng
% A student from Glasgow College, UESTC
```

```
% UoG matrix: 2289258z UESTC matrix: 2016200302027

function [ber, numBits] = my_commdoc_bertool(EbNo, maxNumErrs, ...
  maxNumBits)
%BERTOOLTEMPLATE Template for a BERTool simulation function.
%   This file is a template for a BERTool-compatible
%   simulation function.  To use the template, insert your
%   own code in the places marked "INSERT YOUR CODE HERE"
%   and save the result as a file on your MATLAB path.
%   Then use the Monte Carlo panel of BERTool to execute
%   the script.
%
%   For more information about this template and an example
%   that uses it, see the Communications System Toolbox documentation.
%
%   See also BERTOOL, VITERBISIM.

% Copyright 1996-2010 The MathWorks, Inc.

% Import Java class for BERTool.
import com.mathworks.toolbox.comm.BERTool;

% Initialize variables related to exit criteria.
totErr  = 0; % Number of errors observed
numBits = 0; % Number of bits processed

% --- Set up parameters.
% --- INSERT YOUR CODE HERE.
% Setup
% Define parameters.
M = 4; % Size of signal constellation
k = log2(M); % Number of bits per symbol
n = 1000; % Number of bits to process
nsamp = 1; % Oversampling rate

% Simulate until number of errors exceeds maxNumErrs
% or number of bits processed exceeds maxNumBits.
while((totErr < maxNumErrs) && (numBits < maxNumBits))
  % Check if the user clicked the Stop button of BERTool.
  if (BERTool.getSimulationStop)
    break;
  end
% --- Proceed with simulation.
% --- Be sure to update totErr and numBits.
% --- INSERT YOUR CODE HERE.

%% Incorporating Gray Coding
% This example, described in the Getting Started chapter of the
% Communications Toolbox documentation, aims to solve the following
% problem:
%
% Modify the modulation example (COMMDOC_MOD) so that it uses
% a Gray-coded signal constellation.
```

```
% Copyright 1996-2009 The MathWorks, Inc.

%% Setup
% Define parameters.
M = 16;                        % Size of signal constellation
k = log2(M);                   % Number of bits per symbol
n = 3e4;                       % Number of bits to process
nSamp = 1;                     % Oversampling rate

%% Create Modulator and Demodulator
hMod = modem.qammod(M);            % Create a 16-QAM modulator
hMod.InputType = 'Bit';            % Accept bits as inputs
hMod.SymbolOrder = 'Gray';         % Accept bits as inputs
hDemod = modem.qamdemod(hMod);     % Create a 16-QAM based on the ...
   modulator

%% Signal Source
% Create a binary data stream as a column vector.
x = randi([0 1],n,1); % Random binary data stream

%% Modulation
% Modulate using 16-QAM.
y = modulate(hMod,x);

%% Transmitted Signal
yTx = y;

%% Channel
% Send signal over an AWGN channel.
%EbNo = 10; % In dB
SNR = EbNo + 10*log10(k) - 10*log10(nSamp);
yNoisy = awgn(yTx,SNR,'measured');

%% Received Signal
yRx = yNoisy;

%% Scatter Plot
% Create scatter plot of noisy signal and ideal constellation points
hScatter = commscope.ScatterPlot;                  % Create a scatter ...
   plot scope
hScatter.Constellation = hMod.Constellation; % Set expected ...
   constellation
hScatter.SamplesPerSymbol = nSamp;                 % Set oversampling rate
hScatter.PlotSettings.Constellation = 'on';  % Display ideal ...
   constellation
update(hScatter, yRx(1:5e3))                       % Send received signal ...
   to the scope
%title('Received Signal');

%% Demodulation
% Demodulate signal using 16-QAM.
z = demodulate(hDemod,yRx);
```

```
%% BER Computation
% Compare x and z to obtain the number of errors and
% the bit error rate.
[number_of_errors,bit_error_rate] = biterr(x,z)

%% Update totErr and numBits.
totErr = totErr + number_of_errors;
numBits = numBits + n;
end % End of loop
% Compute the BER.
ber = totErr/numBits;
```

5 EXERCISE 5

5.1 Introduction

This experiment is very similar to the previous one, which requires us to use Simulink to make the model for further analysis by the BERTool. Use the simulated output of the system to compare with the theoretical results. This comparison contributes to the verification of our design. The results are as shown as follows.

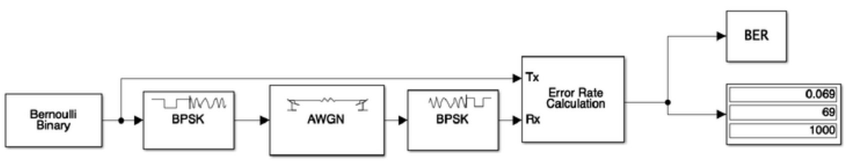
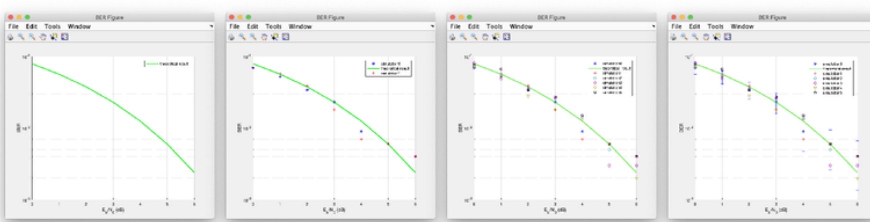


Figure 8: Prepared Model for Communication



Theoretical   Two Simulations   Six Simulations   90 % confidence

Figure 9: Simulation and Theoretical Result of Above Model

5.2 System Design

First of all, the model of our communication system is built for further evaluation by using the BERTool in the following steps. For designing the system, The system begins with a Bernoulli binary generator, after which a BPSK modulate block and an AWGN channel block are connected. After the demodulator, the number of errors and the bit error rate would be calculated by a monitor block. When we got the system, use the BERTool to simulate the above model. Plot the simulated result with the theoretical result of the relation between the BER and the SNR. The result are drawn and compared in figures.

5.3 Results

From the above system, as shown in the Figure 8, we use the BERtoolbox, which is the same as the previous exercise, to generate the curve of BER-SNR relation, as shown in Figure 9. The first subplot of Figure 9 is the theoretical result of the system. The next two subplots include more simulations for which the AWGN is added to the channel. The forth subplot shows the range of BER in any SNR position with a 90 percent confidence area.

5.4 Conclusion and Learning Outcomes

The models from Simulink can be used to generate models which can be tested in BERtool. The designed system can be analysis by firstly generate the theoretical results from the saved model. Then, the designed system with noise channel are run and tested. This method can figure out if the BER-SNR curve of our model is heavily influenced by the noize. It can also verify if the simulated result is stick to the real result.

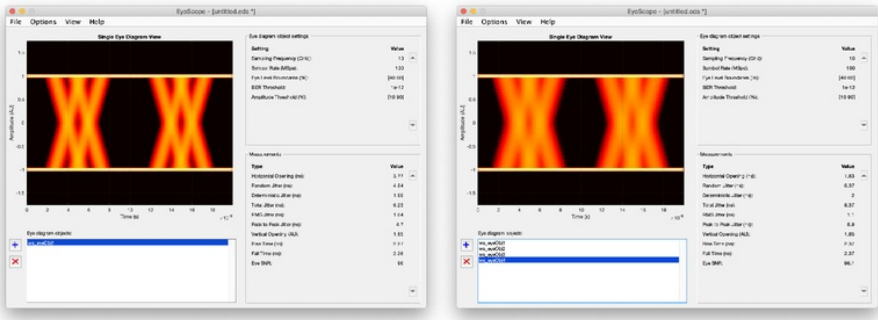
6 EXERCISE 6

6.1 Introduction

This experiment asks us to use MATLAB toolbox to analysis the eye diagram. The meaning and information carried by the eye diagram should be understood.

6.2 System Design

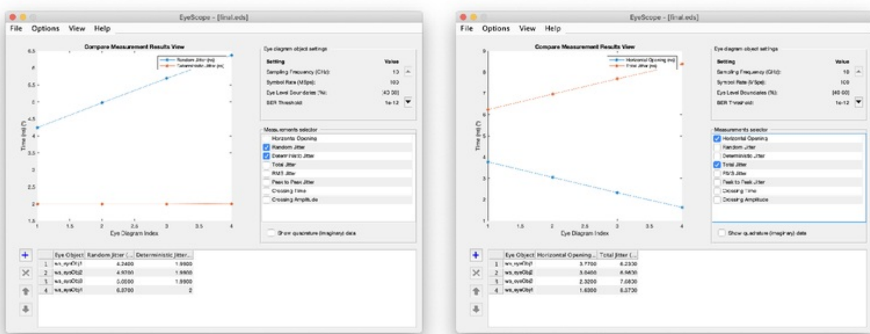
At the very beginning, loaded the stored signal data to the MATLAB, which include five more eye-diagram samples with different signal qualities. The toolbox is used to display these eye-diagram. Some of the appearance of the signal



Relatively higher quality of data Relatively lower quality of data

Figure 10: The sample eye diagram of two signals

and the quality of the signal are extracted and analyzed. The analyzing result is shown in Figure 11.



Setting and Property Setting and Property

Figure 11: Use toolbox to investigate ptopertoes of the signal via eye diagram

6.3 Results

The eye-diagram of the loaded data is shown by using the toolbox, which is shown in Figure 10, where the two sub-plot shows how well the quality is for different signals. Then, the curve between random and deterministic jitters and horizontal jitters and the eye-opening, are extracted by the toolbox and plotted in Figure 11.

6.4 Conclusion and Learning Outcomes

From the graph, we can see that the signal from the first subplot of Figure 10 has a better quality comparing with the second sub-plot. The reason is that, for the second sub-plot, the eye is not open as large as the first sub-plot, which means the horizontal time jitter of this signal is larger than the first plot, which reduces the signal quality.

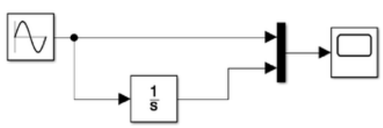
7 EXERCISE 7

7.1 Introduction

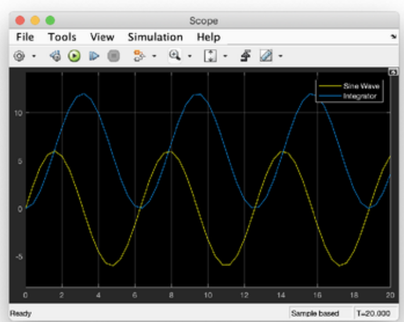
The experiment requires us to use Simulink to construct and simulate a simple model. We need to learn how to generate the Simulink model, find blocks, connect each block, and run it.

7.2 System Design

The system includes a sine wave generator, an integration block, and finally added and displayed on a scope.



The Simulink block of the system

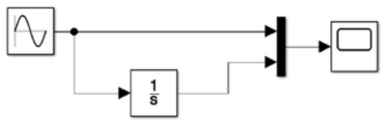


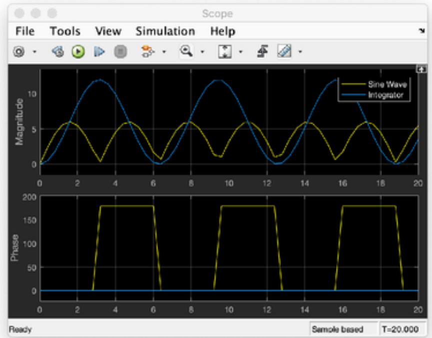
The wave form of the generated signals

Figure 12: System and output signals

7.3 Results

The block diagram of a system, shown in Figure 12 sub-plot one, is simulated during this experiment. The output of the system includes one original sine





The Simulink block of the system

The wave form of the generated signals

Figure 13: System and output signals

wave and an integrated sine wave (The above sine wave). For integration, it is generated by the block $\frac{1}{s}$. The phase and magnitude of these two signals is also shown in Figure 13.

7.4 Conclusion and Learning Outcomes

The Simulink Block $\frac{1}{s}$ is used to do the integration of the sine input wave because of the integration property of the Fourier Transform. We can conclude that the Simulink blocks can be used to generated considerably complex systems, which includes integration and differentiation.

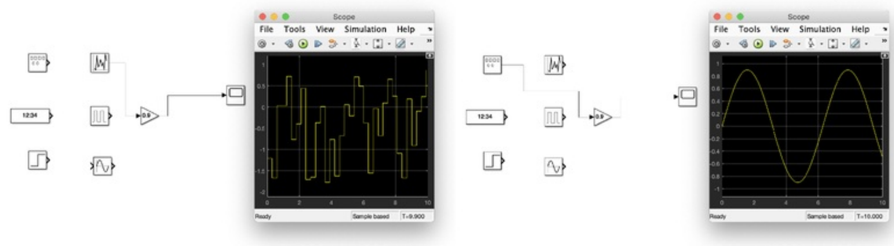
8 EXERCISE 8

8.1 Introduction

This experiment requires us to use Simulink blocks to simulate different data inputs, which need to use the different signal generator. The scope is used to help to explore the different signal input block.

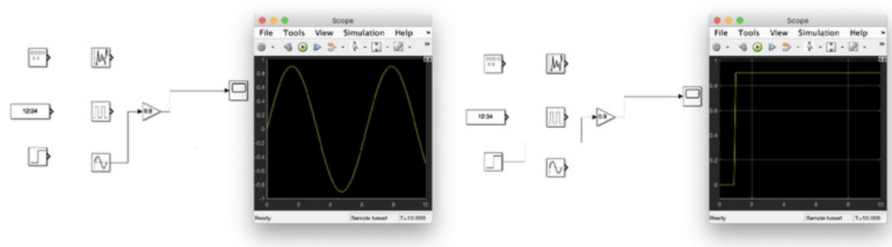
8.2 System Design

The system includes different types of signal inputs and a scope, which is used to test each input block. For example, a sine wave generator, a rectangular wave generator, a clock, a step signal generator, a random signal generator as well as a multiple-use signal generator, which can generate different types of signal. A scope is connected to the block, which is needed to test.



Random signal input Output of the signal Any wave form input Output of the signal

Figure 14: Signal inputs and the sample outputs



Sine signal input Output of the signal Step signal input Output of the signal

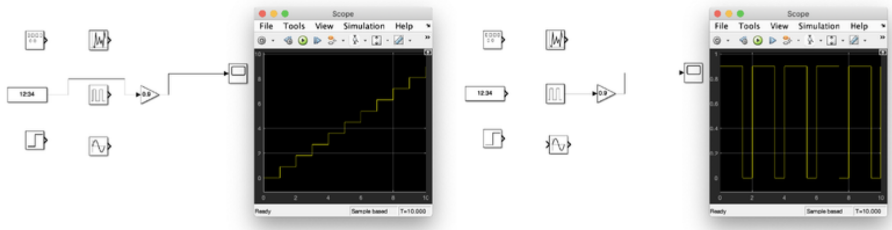
Figure 15: Signal inputs and the sample outputs

8.3 Results

All the input modules were first placed on Simulink, which includes the sine wave, square wave, random signal, step function, time signal, and multiple-use signal generator. Each of the signal generators connects to a scope, which is used to display the testing results after their parameters are set properly, as shown in Figure 14 15 16.

8.4 Conclusion and Learning Outcomes

The Simulink can be used to generate different types of signals, which includes all the commonly used signals by using different blocks. Plus, even the needed



Time signal input Output of the signal Square wave input Output of the signal

Figure 16: Signal inputs and the sample outputs

block are not predefined; we can still use the MATLAB code to set the needed block by ourselves.

9 EXERCISE 9

9.1 Introduction

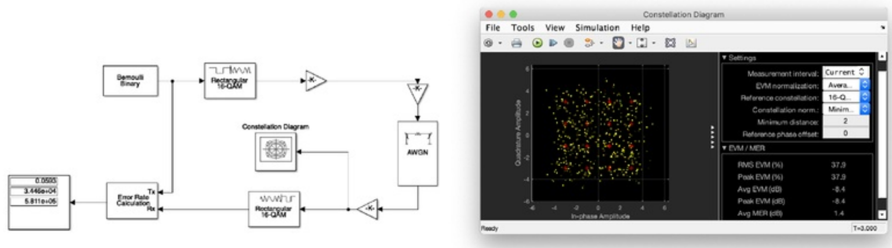
This experiment asked me to simulate a digital communication system by using the rectangular QAM under the AWGN. The number of errors, error rate, and total number of bits are required to gain from the simulation.

9.2 System Design

Under the instruction of the lab manual, all the blocks are found in Simulink and appropriately connected. The binary codes are generated by the Bernoulli Binary Generator Block, which is connected by a QAM modulator, sent and amplified by gain transmitter. Then it is attenuated and distorted by the AWGN channel and amplified in the receiver by using the Gain receiver block. The received data are demodulated and analyzed by comparing them with the original data.

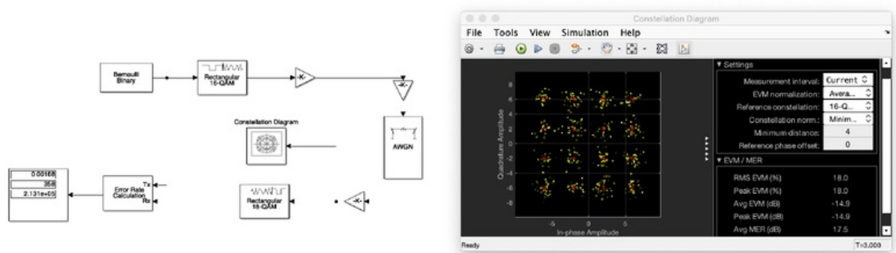
9.3 Results

From the Figure 17, the channel and the QAM modulation system is constructed in the Simulink. The block located at the lower-left quarter is used to calculate the BER and total numbers of bit errors. The results are directly displayed on the block. Besides, the constellation block is used to plot the received data is a constellation diagram. Comparing with these two figures. The only difference between these two experiments is the minimum distance between each star, where Figure 17 has a distance of 2, and Figure 18 has a distance of 4.



Bolck diagram of the system   Receiving constellation

Figure 17: AWGN channel with 16-QAM modulation of a point distance of 2



Bolck diagram of the system   Receiving constellation

Figure 18: AWGN channel with 16-QAM modulation of a point distance of 4

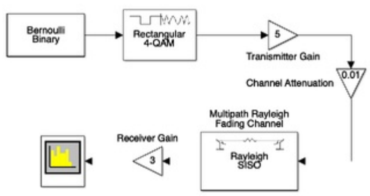
9.4 Conclusion and Learning Outcomes

From this experiment, we can conclude that Simulink can be used to construct and test communication systems. It can calculate the BER as well as display the constellation diagram. Besides, the distance between each star on the constellation, which is a direct representation od different SNR or signal power, have a direct influence on the BER. Figure 17 18 shows that the after distance rise from2 two 4 between each stars, the BER drop from 5.93% to 0.168%.

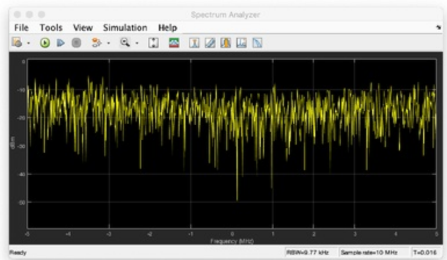
10 EXERCISE 10

10.1 Introduction

This experiment simulates a close to reality transmission model by designing a multipath channel with different noise, distortion, and attenuation on each path. The experiment verifies how the different levels of the Doppler effect on diverse paths would influence the quality of communication.

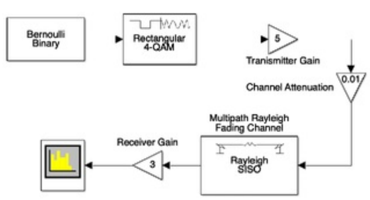


Block diagram of a MRFC channel

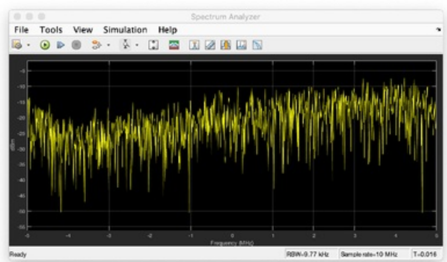


The Rayleigh fading of maximum 2 Hz

Figure 19: Graphs of the message signal



Block diagram of a MRFC channel



The Rayleigh fading of maximum 70 Hz

Figure 20: Simulation of data transfer in a Multipath Rayleigh Fading Channel

10.2 System Design

The Bernoulli Data Generator Block is used to generate random data, which is then passing through a QAM modulator. The following block is a multipass channel, which can delay, attenuate, and add the Doppler effect to each pass of the signal. The receiver received all the signal and added them together before the demodulator. Finally, the received data is used to calculate the bit error rate by comparing it with the original data.

$$f(z) = \frac{z}{\sigma^2} \exp\left(-\frac{z^2}{2\sigma^2}\right), \quad z \geqslant 0$$

10.3 Results

A 16-QAM or a 4-QAM modulation signal is generated in the blocks, as shown in the first subplot of Figure 19 and Figure 20. Then, a Multipath Rayleigh fading channel is used to simulate the environment. After the channel, the received signal is displayed in the second subplots in the above figures.

10.4 Conclusion and Learning Outcomes

For this experiment, we can conclude that the MRFC is used to simulate the multipath effect, which is a common problem for communication. The multipath effect mainly caused by the different propagation path of signals. As the path is different, at the same time, the received signal could travel in different distance and result in the time jitter. The signal from different paths could have different types and levels of distortion and attenuation. This effect results in the signal distortion, and if we use the eye diagram to analysis the received signal, we can see this jitter. More than that, for each path, the signal would have different attenuations, distortion, and the doppler effect, which is varied in this experiment. This effect could be caused by the different angles of transmission and the different numbers transmission between the cloud and land. Another example of multipath could happen in the room, formed by the different transmitting direction and different numbers of reflections (just like echos).

11 SUMMARY

For doing these ten experiments. I have learned how to use MATLAB to generate various signals, modulation techniques, different types of channel, demodulation techniques, and finally, BER calculation. Briefly speaking, how to operate this powerful tool, MATLAB, to construct and evaluate communication systems. Besides, the evaluation techniques like BER calculation, BER-SNR curve, and the eye-diagram are realized and tested. All of the above help me to construct a solid foundation in the field of signal processing, system designing, and modulation implementation techniques.

REFERENCES

[1] Upamanyu Madhow. *Introduction to Communication Systems*. Cambridge University Press, 2014.

[2] Dale Louise Robert, Kenneth. *Software Defined Radio MATLAB and Simulink*. Strathclyde Academic Media, 2015.

[3] Imran Shafique Ansari. *Wireless and Optical Transmission Systems Lab Manual*. Glasgow College, UESTC, 2019.

ChanggangZheng_Laboratory_Report_2289258Z_20162003020.

ORIGINALITY REPORT

20%	18%	7%	16%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.tankonyvtar.hu Internet Source	4%
2	e2evts.blogspot.com Internet Source	4%
3	www.matlabanalysis.com Internet Source	2%
4	Submitted to University of Bradford Student Paper	2%
5	xuyuann.blog.163.com Internet Source	1%
6	Submitted to University of Bath Student Paper	1%
7	www.slideserve.com Internet Source	1%
8	www.mathworks.com Internet Source	1%
9	www.dip.ee.uct.ac.za Internet Source	1%
10	rabaska.net Internet Source	<1%

11	www.mathworks.co.kr Internet Source	<1%
12	Submitted to University of Michigan, Dearborn Student Paper	<1%
13	Submitted to Kingston University Student Paper	<1%
14	download.micron.com Internet Source	<1%
15	www.nexoncn.com Internet Source	<1%
16	Submitted to University of Glamorgan Student Paper	<1%
17	Submitted to Bilkent University Student Paper	<1%
18	Submitted to Coventry University Student Paper	<1%
19	journals.sagepub.com Internet Source	<1%
20	nrl.northumbria.ac.uk Internet Source	<1%
21	Submitted to University of Bedfordshire Student Paper	<1%
22	infostore.saiglobal.com Internet Source	<1%

23	oxpedia.org Internet Source		<1%
24	studentsyllabus.wordpress.com Internet Source		<1%
25	willame.francois.free.fr Internet Source		<1%
26	Submitted to University of Strathclyde Student Paper		<1%
27	hdl.handle.net Internet Source		<1%
28	Submitted to Higher Education Commission Pakistan Student Paper		<1%
29	Submitted to University of Wolverhampton Student Paper		<1%
30	Submitted to De Montfort University Student Paper		<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		