Urkund Analysis Result

Analysed Document:	ChanggangZheng_Laboratory_Report_2289258Z_2016200302027.pdf (D59127507)
Submitted:	11/18/2019 6:58:00 PM
Submitted By:	2289258Z@student.gla.ac.uk
Significance:	22 %

Sources included in the report:

2289228W_Wu_UESTC4024_Lab_Report.pdf (D58105271)
2289231X_2016200301035_XUYANG.pdf (D59097678)
MaoJianqiao-2289247M-2016200302016-Lab_Report.pdf (D59033413)
Lab Report_Yang Yuhang_2289224Y.pdf (D58882170)
Wireless and Optical Transmission Systems Report 2289314y PUXI YU.docx (D57987001)
Wangpan_2289295_2016200303029.pdf (D58877452)
https://jp.mathworks.com/matlabcentral/answers/4509-index-exceeds-matrix-dimensions
https://www.tankonyvtar.hu/en/tartalom/tamop412A/2011-0041_introduction_matlab/ch08s09.html
https://core.ac.uk/download/pdf/53187569.pdf

Instances where selected sources appear:

18

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

 introduction This Laboratory report is about Lab report of the course Wireless and Opti- cal Transmission Systems. The experiment includes information from gener- ate 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 envi- ronments 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 rela- tionship 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, commu- nication system evaluation techniques, and so on, which provides an excellent technical basis for wireless communication simulation and testing.

1 exercise 1 1 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. 0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1 Binary Value Random Bits 0 5 10 15 20 25 30 35 40 Bit Index Random signal with length 40 0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1 Binary Value Random Bits 0 10 20 30 40 50 60 70 80 Bit Index 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

conclusion and learning outcomes 2 0 1 2 3 4 5 6 7 Integer Value Rabdom Symbols 1 2 3 4 5 6 7 8 9 10 Symbol Index Random Symbols -5 -4 -3 -2 -1 0 1 2 3 4 5 In-Phase -5 -4 -3 -2 -1 0 1 2 3 4 5 Quadrature Received Signal Received Signal Signal Constellation 8 QAM 0 5 10 15 Integer

Value Rabdom Symbols 1 2 3 4 5 6 7 8 9 10 Symbol Index Random Symbols -5 -4 -3 -2 -1 0 1 2 3 4 5 In-Phase -5 -4 -3 -2 -1 0 1 2 3 4 5 Quadrature Received Signal Received Signal Signal Constellation 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. 1 number_of_errors = 2 3 172 4 5 6 bit_error_rate = 7 8 0.0057 9 10 << 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.

code 3 1 % Type and revised by Changgnag Zheng 2 % A student from Glasgow College, UESTC 3 % UoG matrix: 2289258z UESTC matrix: 2016200302027 4 5 %% Define parameters 6

grid minor 19 set(EX1_plot1, 'PaperPosition', [0.05 0.05 9 7]); 20 set(EX1_plot1, 'PaperSize', [9.05 7.05]); 21 saveas(EX1_plot1,['EX1_plot1.pdf'],'pdf') 22 23 %%

Bit

to symbol mapping 24

filled','LineWidth',1.5,'color',[1,0.5,0.5]); 27 title('Rabdom Symbols'); 28 xlabel('Symbol Index'); 29 ylabel('Integer Value') 30

grid minor 31 set(EX1_plot2, 'PaperPosition', [0.05 0.05 9 7]); 32 set(EX1_plot2, 'PaperSize', [9.05 7.05]); 33 saveas(EX1_plot2,['EX1_plot2.pdf'],'pdf') 34 35 %%

0: https://jp.mathworks.com/matlabcentral/answers/4509-index-exceeds-matrix-dimensions 66%

Modulation 36 y = modulate(modem.qammod(M), xsym); % using the 16 QAM 37 38 %% Transmissitted

Signal 39 ytx = y; 40 41 %%

Channel 42 % Send signal over an AEGN channel 43

EbNo = 9; 44

snr = EbNo + 10 * log10(k) - 10 * log10(nsamp); 45 ynoisy = awgn(ytx,snr,'measured'); 46 47 %%

Receive channel 48 yrx = ynoisy; 49 50 %%

Scatter plot 51

h = scatterplot(yrx(1:nsamp * 5e3),nsamp, 0,'.'); 52

hold on; 53

EX1_plot3 =

scatterplot(ytx(1:5e3),1,0,'+',h);

2 exercise 2 4 54 title('Received Signal') 55 legend('Received Signal','Signal Constellation'); 56

axis([-5 5 -5 5]); 57

grid minor 58 set(EX1_plot3, 'PaperPosition', [0.05 0.05 9 7]); 59 set(EX1_plot3, 'PaperSize', [9.05 7.05]); 60 saveas(EX1_plot3,['EX1_plot3.pdf'],'pdf') 61

0: https://core.ac.uk/download/pdf/53187569.pdf 99%

hold off; 62 63 %%

Demodulation 64 % Demodulate signal using 16-QAM. 65 zsym = demodulate (modem.qamdemod(M),

yrx); 66 67 %%

> Symbol-to-Bit Mapping 68 %
>
> Undo the bit-to-symbol mapping performed earlier. 69
>
> z = de2bi(zsym, 'left-msb'); %
>
> Convert integers to bits. 70 %
>
> Convert z from a matrix to a vector. 71 z =
>
> reshape(z.', prod(size(z)),1); 72 73 %%
>
> BER Computation 74 %
>
> Compare x and z to obtain the number of errors and
>
> the bit 75 % error rate. 76 [
>
> 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

system design 5 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. 1 % Type and revised by Changgnag Zheng

code 6 2 % A student from Glasgow College, UESTC 3 % UoG matrix: 2289258z UESTC matrix: 2016200302027 4 5 %%

create a QPSK modulator object. 6

0: MaoJianqiao-2289247M-2016200302016-Lab_Report.pdf 81%

hMod = modem.pskmod('M', 8, 'PhaseOffset', pi/4); 7 %% create an upsampling filter 8

Rup = 16; % up sampling rate 9

hFilDesign = fdesign.pulseshaping(Rup, 'Raised Cosine','Nsym,Beta', ... Rup,0.50); 10 hFil = design(hFilDesign); 11 %%

create the transimit signal 12

d = randi([0 hMod.M-1], 100, 1); % Generate data symbols 13 sym = modulate(hMod, d); % Generate modulated symbols 14 xmt = filter(hFil, upsample(sym, Rup)); 15 %%

create a

scatter plot and set the samples per symbol to

ghe ... upsampling rate 16

hScope =

commscope.ScatterPlot 17 grid minor 18 hScope.SamplesPerSymbol = Rup; 19 %% set the constellation value 20

hScope.Constellation = hMod.Constellation; 21 %%

groupdelay 22 groupDelay = (hFilDesign.NumberOfSymbols/2); 23 hScope.MeasurementDelay = groupDelay /hScope.SymbolRate; 24

update(

hScope, xmt) 25

0: MaoJianqiao-2289247M-2016200302016-Lab_Report.pdf 81%

hScope.PlotSettings.Constellation = 'on'; 26 %% 27

hFil.Numerator = hFil.Numerator / max(hFil.Numerator); 28 %% 29 xmt = filter(hFil, upsample (sym, Rup)); 30 %% 31

reset(hScope) 32 update(hScope, xmt) 33

hScope.PlotSettings.

SignalTrajectory = 'on'; 34

hScope.PlotSettings.SignalTrajectoryStyle = ':m'; 35 autoscale(hScope) 36

> rcv = awgn(xmt, 20, 'measured'); % Add AWGN 37 %% 38 reset(
>
> hScope) 39 update(hScope, rcv) 40

hScope.PlotSettings.

SignalTrajectory = 'off'; 41 hScope.PlotSettings.Constellation = 'on'; 42

hold off;

3 exercise 3 7 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. -1 -0.8 -0.6 -0.4 -0.2 0 0.2 0.4 0.6 0.8 1 In-Phase -1 -0.8 -0.6 -0.4 -0.2 0 0.2 0.4 0.6 0.8 1 Quadrature Scatter plot 16-PSK -0.8 -0.6 -0.4 -0.2 0 0.2 0.4 0.6 0.8 In-Phase -0.8 -0.6 -0.4 -0.2 0 0.2 0.4 0.6 0.8 Quadrature Scatter plot 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32-QAM -4 -3 -2 -1 0 1 2 3 4 In-Phase -4 -3 -2 -1 0 1 2 3 4 Quadrature Constellation for Gray-Coded 8-QAM 000 001 010 011 100 101 110 111 8-QAM -4 -3 -2 -1 0 1 2 3 4 In-Phase -4 -3 -2 -1 0 1 2 3 4 Quadrature Customised Constellation for 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. -4 -3 -2 -1 0 1 2 3 4 In-Phase -4 -3 -2 -1 0 1 2 3 4 Quadrature Customised Constellation for QAM New Constellation Type 1 -3 -2 -1 0 1 2 3 In-Phase -3 -2 -1 0 1 2 3 Quadrature Customised Constellation for QAM New Constellation Type 2 Figure 6: New Constellations

results 8 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. 1 % Type and revised by Changgnag Zheng 2 % A student from Glasgow College, UESTC 3 % UoG matrix: 2289258z UESTC matrix: 2016200302027 4 5 % input bertool in the commond line 6 %% constellation for 16-psk 7 M = 16; 8 x = [0:M-1]; 9 EX3_plot1 = scatterplot(modulate(modem.pskmod(M), x)); 10 grid minor 11 set(EX3_plot1, 'PaperPosition', [0.05 0.05 9 7]); 12 set(EX3_plot1, 'PaperSize', [9.05 7.05]); 13 saveas(EX3_plot1,['EX3_plot1.pdf'],'pdf') 14 15 %%

> 0: 2289228W_Wu_UESTC4024_Lab_Report.pdf 83%
>
> constellation for 32-QAM 16
>
> M = 32; 17

x = [0:M-1]; 18

y = modulate(modem.qammod(M), x); 19 scale = modnorm(y, 'peakpow', 1); 20 y = scale * y; %

Scale the constellation. 21

EX3_plot2 =

scatterplot(y); %

Plot the scaled constellation. 22 %

Include text annotations that number the points. 23

hold on; % Make sure the annotations go

in the same figure. 24

for jj=1:length(y) 25

text(real(y(jj)), imag(y(

jj)), [' ' num2str(jj-1)]); 26

end 27 hold off; 28

grid minor 29 set(EX3_plot2, 'PaperPosition', [0.05 0.05 9 7]); 30 set(EX3_plot2, 'PaperSize', [9.05 7.05]);

code 9 31 saveas(EX3_plot2,['EX3_plot2.pdf'],'pdf') 32 33 %%

0: 2289228W_Wu_UESTC4024_Lab_Report.pdf 100%

gray-coded signal constellation 34

M = 8; 35

x = [0:M-1]; 36 y = modulate(modem.qammod('M', M, 'SymbolOrder', 'Gray'), x); 37 %

Plot the Gray-coded constellation. 38

EX3_plot3 =

0: 2289228W_Wu_UESTC4024_Lab_Report.pdf 100%

scatterplot(y, 1, 0, 'b.'); % Dots for points. 39 %

Include text annotations that number the points in binary. 40 hold on; % Make sure the annotations go

in the same figure. 41

annot = dec2bin([0:length(y)-1], log2(M)); 42

text(real(y)+0.15, imag(y), annot); 43 axis([-4 4 -4 4]); 44 title('Constellation for Gray-Coded 8-QAM'); 45 hold off; 46

grid minor 47 set(EX3_plot3, 'PaperPosition', [0.05 0.05 9 7]); 48 set(EX3_plot3, 'PaperSize', [9.05 7.05]); 49 saveas(EX3_plot3,['EX3_plot3.pdf'],'pdf') 50 51 %%

0: Wireless and Optical Transmission Systems Report 2289314y PUXI YU.docx 85%

customised

constellation 1 for QAM 52 %

Describe constellation. 53

inphase = [0 -2 0 2 0 -3 -2 -1]; 54 quadr = [3 2 2 2 1 0 0 0]; 55 inphase = [inphase; -inphase]; 56 inphase = inphase(:); 57 quadr = [quadr; -quadr]; 58 quadr = quadr(:); 59 const = inphase + j * quadr; 60 %

Plot constellation. 61 EX3_plot4 = scatterplot(const, 1, 0, 'o'); 62 hold on; 63

axis([-4 4 -4 4]); 64

title('Customised Constellation

for QAM'); 65 hold off; 66

grid minor 67 set(EX3_plot4, 'PaperPosition', [0.05 0.05 9 7]); 68 set(EX3_plot4, 'PaperSize', [9.05 7.05]); 69 saveas(EX3_plot4,['EX3_plot4.pdf'],'pdf') 70 71 %%

0: Wireless and Optical Transmission Systems Report 2289314y PUXI YU.docx 72%

customised

constellation 2 for QAM 72

inphase = [0 1 2 sqrt(2)]; 73

quadr = [1 0 0 sqrt(2)]; 74

inphase = [inphase; -inphase]; 75 inphase = inphase(:); 76 quadr = [quadr; -quadr]; 77 quadr = quadr(:); 78 const = inphase + j * quadr; 79 %

Plot constellation. 80 EX3_plot5 = scatterplot(const, 1, 0, 'o'); 81 hold on; 82

axis([-4 4 -4 4]); 83

title('Customised Constellation

for QAM');

4

exercise 4 10 84 hold off; 85

grid minor 86 set(EX3_plot5, 'PaperPosition', [0.05 0.05 9 7]); 87 set(EX3_plot5, 'PaperSize', [9.05 7.05]); 88 saveas(EX3_plot5,['EX3_plot5.pdf'],'pdf') 89 90 %% Manual

0: Wireless and Optical Transmission Systems Report 2289314y PUXI YU.docx 84%

constellation. 91

inphase = [1/2 -1/2 1 0 3/2 -3/2 1 -1]; 92 quadr = [1 1 0 2 1 1 2 2]; 93

inphase = [inphase; -inphase]; inphase = inphase(:); 94 quadr = [quadr; -quadr]; quadr = quadr(:); 95 const = inphase + 1i * quadr; 96 %

Plot constellation. 97 EX3_plot6 = scatterplot(const, 1, 0, 'o'); 98 hold on; 99

axis([-3 3 -3 3]); 100

title('Customised Constellation

for QAM'); 101 hold off; 102

grid minor 103 set(EX3_plot6, 'PaperPosition', [0.05 0.05 9 7]); 104 set(EX3_plot6, 'PaperSize', [9.05 7.05]); 105 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 com- munication 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.

0: 2289231X_2016200301035_XUYANG.pdf 78%

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 follow- ing 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.

results 11 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. 1 % Type and revised by Changgnag Zheng 2 % A student from Glasgow College, UESTC

code 12 3 % UoG matrix: 2289258z UESTC matrix: 2016200302027 4 5

0: Lab Report_Yang Yuhang_2289224Y.pdf 100%

function [ber, numBits] = my_commdoc_bertool(EbNo, maxNumErrs, ... maxNumBits) 6 % BERTOOLTEMPLATE Template for a BERTool simulation function. 7 %

This file is a template for a BERTool-compatible 8 % simulation function. To use the template,

insert your 9 % own code in the places marked "INSERT YOUR CODE HERE" 10 %

and save the result as a file on your MATLAB path. 11 % Then use the Monte Carlo panel of BERTool to execute 12 % the script. 13 % 14 %

For more information about this template and an example 15 % that uses it, see the Communications System Toolbox documentation. 16 % 17 % See also BERTOOL, VITERBISIM. 18 19 %

Copyright 1996-2010 The MathWorks, Inc. 20 21 %

Import Java class for BERTool. 22

import com.mathworks.toolbox.comm.BERTool; 23 24 % Initialize variables related to exit criteria. 25

totErr = 0; % Number of errors observed 26 numBits = 0; % Number of bits processed 27 28 % ---

Set up parameters. 29 % --- INSERT YOUR CODE HERE. 30 %

0: MaoJianqiao-2289247M-2016200302016-Lab_Report.pdf 99%

Setup 31 %

Define parameters. 32

M = 4; %

Size of signal constellation 33 k = log2(M); % Number of bits per symbol 34 n = 1000; % Number of

bits

to process 35

nsamp = 1; % Oversampling rate 36 37 %

Simulate until number of errors

exceeds maxNumErrs 38 % or number of bits processed exceeds maxNumBits. 39 while ((totErr > maxNumErrs) && (numBits > maxNumBits)) 40 % Check

if the user clicked the Stop button of BERTool. 41 if (BERTool.getSimulationStop) 42 break; 43 end 44 % ---

0: Lab Report_Yang Yuhang_2289224Y.pdf 100%

Proceed with simulation. 45 % ---

Be sure to update totErr and numBits. 46 % --- INSERT YOUR CODE HERE. 47 48 %%

0: https://www.tankonyvtar.hu/en/tartalom/tamop412A/2011-0041_introduction_matlab/ch08s09.html 96%

Incorporating Gray Coding 49 %

This example, described in the Getting Started chapter of the 50 % Communications Toolbox documentation, aims to solve the following 51 % problem: 52 % 53 % Modify the modulation example (COMMDOC_MOD) so that it

uses 54 % a Gray-coded signal constellation.

code 13 55 % Copyright 1996-2009 The MathWorks, Inc. 56 57 %%

Setup 58 %

Define parameters. 59

M = 16; %

Size of signal constellation 60 k = log2(M); % Number of bits per symbol 61 n = 3e4; % Number of

bits

to process 62

nSamp = 1; %

Oversampling rate 63 64 %%

Create

Modulator and Demodulator 65 hMod = modem.qammod(M); % Create a 16-QAM modulator 66

hMod.InputType = 'Bit'; % Accept bits as inputs 67

hMod.SymbolOrder = 'Gray'; %

Accept bits as inputs 68 hDemod = modem.qamdemod(hMod); % Create a 16-QAM

based on the ...

modulator 69 70 %%

Signal Source 71 % Create a binary data stream as a column vector. 72 x =

randi([0 1],n,1); % Random binary data stream 73 74 %%

Modulation 75 % Modulate using 16-QAM. 76 y =

modulate(hMod,x); 77 78 %%

Transmitted

Signal 79 yTx = y; 80 81 %%

Channel 82 % Send signal over an AWGN channel. 83 %

EbNo = 10; %

In dB 84

SNR =

EbNo + 10 *

log10(k) - 10 * log10(nSamp); 85 yNoisy = awgn(yTx,SNR,'measured'); 86 87 %%

Received Signal 88

yRx = yNoisy; 89 90 %%

Scatter Plot 91 %

Create scatter plot of noisy signal and

ideal constellation points 92 hScatter = commscope.ScatterPlot; % Create a scatter ... plot scope 93 hScatter.Constellation = hMod.Constellation; % Set expected ... constellation 94 hScatter.SamplesPerSymbol = nSamp; % Set oversampling rate 95 hScatter.PlotSettings.Constellation = 'on'; % Display ideal ... constellation 96 update(hScatter, yRx(1:5e3)) % Send received signal ... to the scope 97 %title('

Received Signal'); 98 99 %% Demodulation 100 % Demodulate signal using 16-QAM. 101 z = demodulate(hDemod,yRx); 102

5

exercise 5 14 103 %%

BER Computation 104 %

Compare x and z to obtain the number of errors and 105 %

the bit

error rate. 106 [

number_of_errors,bit_error_rate] = biterr(x,

z) 107 108 %%

Update totErr

and numBits. 109

totErr = totErr + number_of_errors; 110 numBits = numBits + n; 111

end % End of loop 112 % Compute the BER. 113 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 sim- ulated output of the system to compare with the theoretical results. This com- parison 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

system design 15 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 tool- box is used to display these eye-diagram. Some of the appearance of the signal

results 16 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.

conclusion and learning outcomes 17 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

conclusion and learning outcomes 18 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 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 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

results 19 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

9 exercise 9 20 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 differ-

conclusion and learning outcomes 21 ence 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 constella- tion, 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 22 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

results 23 the demodulator. Finally, the received data is used to calculate the bit error rate by comparing it with the original data. f(z) = z 2 exp - z 2 2 2 , z < 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 multi- path 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 sig- nal, 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 trans- mitting direction and different numbers of reflections (just like echos). 11 summary For doing these ten experiments. I have learned how to use MATLAB to gener- ate various signals, modulation techniques, different types of channel, demodu- lation 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 24 references [1] Upamanyu Madhow. Introduction to Communication Systems. Cambridge Uni- versity 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.

Hit and source - focused comparison, Side by Side:

Left side: As student entered the text in the submitted document.
Right side: As the text appears in the source.

Instances from: 2289228W_Wu_UESTC4024_Lab_Report.pdf

7	83%	7: 2289228W_Wu_UESTC4024_Lab_Report.pdf	83%
constellation for 32-QAM 16 M = 32; 17 x = [0:M-1]; 18 y = modulate(modem.qammod(M), x); 19 scale = modnorm(y, 'peakpow', 1); 20 y = scale * y; % Scale the constellation. 21 EX3_plot2 = scatterplot(y); % Plot the scaled constellation. 22 % Include text annotations that number the points. 23 hold on; % Make sure the annotations go		Constellation for 32-QAM 9 M = 32; 10 x = [0:M-1]; 11 y = modulate(modem.qammod(M), x); 12 scale = modnorm(y, 'peakpow', 1); 13 y = scale * y; % Scale the constellation 14 15 scatterplot(y); % Plot the scaled constellation. 16 axis([-1.2 1.2 -1.2 1.2]) ; 17 18 % Include text annotations that number the points. 19 hold on;% Make sure the annotations go in the same figure. 20 for jj=1:length(y) 21 text(real(y(jj)), imag(y(jj)), [' ' num2str(jj-1)]); 22 end 23 hold off; 11 24 25 %%	

in the same figure. 24

for jj=1:length(y) 25

text(real(y(jj)), imag(y(

jj)), [' ' num2str(jj-1)]); 26

end 27 hold off; 28

8 100%

gray-coded signal constellation 34

M = 8; 35

x = [0:M-1]; 36 y = modulate(modem.qammod('M', M, 'SymbolOrder', 'Gray'), x); 37 %

Plot the Gray-coded constellation. 38

8: 2289228W_Wu_UESTC4024_Lab_Report.pdf 100%

Gray-Coded Signal Constellation 26 M = 8; 27 x = [0:M-1]; 28 y = modulate(modem.qammod('M', M, 'SymbolOrder', 'Gray'), x); 29 % Plot the Gray-coded constellation. 30

9 100%

scatterplot(y, 1, 0, 'b.'); % Dots for points. 39 %

Include text annotations that number the points in binary. 40 hold on; % Make sure the annotations go

9: 2289228W_Wu_UESTC4024_Lab_Report.pdf 100%

scatterplot(y, 1, 0, 'b.'); % Dots for points. 31 % Include text annotations that number the points in binary. 32 hold on; % Make sure the annotations go in the same figure. 33 annot = dec2bin([0:length(y)-1], log2(M)); 34 text(real(y)+0.15, imag(y),

in the same figure. 41

annot = dec2bin([0:length(y)-1], log2(M)); 42

text(real(y)+0.15, imag(y), annot); 43 axis([-4 4 -4 4]); 44 title ('Constellation for Gray-Coded 8-QAM'); 45 hold off; 46

annot); 35 axis([-4 4 -4 4]); 36 title('Constellation for Gray-Coded 8-QAM'); 37 hold off; 38 39 %%

Instances from: 2289231X_2016200301035_XUYANG.pdf

13	78%

For the system, the

size of constellation, number of bits per symbol, number of bits

to process, oversampling rate,

13: 2289231X_2016200301035_XUYANG.pdf	78%

for the communication system: M: size of signal constellation; k: number of bits per symbol; n: number of bits to process; nsamp: oversampling rate. 3.

Instances from: MaoJianqiao-2289247M-2016200302016-Lab_Report.pdf

5 81%

hMod = modem.pskmod('M', 8, 'PhaseOffset', pi/4); 7 %% create an upsampling filter 8

Rup = 16; % up sampling rate 9

hFilDesign = fdesign.pulseshaping(Rup, 'Raised Cosine','Nsym,Beta', ... Rup,0.50); 10 hFil = design(hFilDesign); 11 %%

create the transimit signal 12

d = randi([0 hMod.M-1], 100, 1); % Generate data symbols 13 sym = modulate(hMod, d); % Generate modulated symbols 14 xmt = filter(hFil, upsample(sym, Rup)); 15 %%

create a

scatter plot and set the samples per symbol to

ghe ... upsampling rate 16

hScope =

commscope.ScatterPlot 17 grid minor 18 hScope.SamplesPerSymbol = Rup; 19 %% set the constellation value 20

5: MaoJianqiao-2289247M-2016200302016-Lab_Report.pdf 81%

hMod = modem.pskmod('M',8 , 'PhaseOffset', pi/8); % Create an upsampling filter, with an upsample rate of 16 Rup =64; % up sampling rate hFilDesign=fdesign.pulseshaping(Rup,'Raised Cosine','Nsym,Beta',Rup,0.50); hFil = design(hFilDesign); % Create the transmit 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 the upsampling rate of the signal. hScope = commscope.ScatterPlot; hScope.SamplesPerSymbol = Rup; %Set the constellation value of the scatter plot to the expected constellation hScope.Constellation = hMod.Constellation; %discard these transient values groupDelay = (hFilDesign.NumberOfSymbols/2); hScope.MeasurementDelay = groupDelay /hScope.SymbolRate; %update

hScope.Constellation = hMod.Constellation; 21 %%

groupdelay 22 groupDelay = (hFilDesign.NumberOfSymbols/2); 23 hScope.MeasurementDelay = groupDelay / hScope.SymbolRate; 24

update(

6 81%

hScope.PlotSettings.Constellation = 'on'; 26 %% 27

hFil.Numerator = hFil.Numerator / max(hFil.Numerator); 28 %% 29 xmt = filter(hFil, upsample(sym, Rup)); 30 %% 31

reset(hScope) 32 update(hScope, xmt) 33

hScope.PlotSettings.

SignalTrajectory = 'on'; 34

hScope.PlotSettings.SignalTrajectoryStyle = ':m'; 35 autoscale (hScope) 36

rcv = awgn(xmt, 20, 'measured'); % Add AWGN 37 %% 38 reset(

hScope) 39 update(hScope, rcv) 40

6: MaoJianqiao-2289247M-2016200302016-Lab_Report.pdf 81%

hScope.PlotSettings.Constellation = 'on'; %Normalise the filter hFil.Numerator = hFil.Numerator / max(hFil.Numerator); % refilter xmt = filter(hFil, upsample(sym, Rup)); %Reset the scope reset(hScope) %update update(hScope, xmt) hScope.PlotSettings.SignalTrajectory = 'on'; autoscale(hScope) hScope.PlotSettings.SignalTrajectoryStyle = ':m'; %add noise rcv = awgn(xmt, 20, 'measured'); % Add AWGN reset(hScope); update (hScope, rcv)

15 99%

15: MaoJianqiao-2289247M-2016200302016-Lab_Report.pdf 99%

Setup 31 %

Define parameters. 32

M = 4; %

Size of signal constellation 33 k = log2(M); % Number of bits per symbol 34 n = 1000; % Number of

bits

to process 35

nsamp = 1; % Oversampling rate 36 37 %

Simulate until number of errors

exceeds maxNumErrs 38 % or number of bits processed exceeds maxNumBits. 39 while((totErr > maxNumErrs) && (numBits > maxNumBits)) 40 % Check

if the user clicked the Stop button of BERTool. 41 if (BERTool.getSimulationStop) 42 break; 43 end 44 % ---

18 100%

totErr = totErr + number_of_errors; 110 numBits = numBits + n; 111

Setup % Define parameters. M = 16; %

Size of signal constellation k = log2(M); % Number of bits per symbol n = 3e5; % 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 %%

18: MaoJianqiao-2289247M-2016200302016-Lab_Report.pdf 100%

totErr =

end % End of loop 112 % Compute the BER. 113 ber = totErr/ numBits; 5	totErr + number_of_errors; numBits = numBits + n; end % End of loop % Compute the BER. ber = totErr/numBits;

Instances from: Lab Report_Yang Yuhang_2289224Y.pdf

14 100%

function [ber, numBits] = my_commdoc_bertool(EbNo, maxNumErrs, ... maxNumBits) 6 %BERTOOLTEMPLATE Template for a BERTool simulation function. 7 %

This file is a template for a BERTool-compatible 8 % simulation function. To use the template,

insert your 9 % own code in the places marked "INSERT YOUR CODE HERE" 10 %

and save the result as a file on your MATLAB path. 11 % Then use the Monte Carlo panel of BERTool to execute 12 % the script. 13 % 14 %

For more information about this template and an example 15 % that uses it, see the Communications System Toolbox documentation. 16 % 17 % See also BERTOOL, VITERBISIM. 18 19 %

Copyright 1996-2010 The MathWorks, Inc. 20 21 %

Import Java class for BERTool. 22

import com.mathworks.toolbox.comm.BERTool; 23 24 % Initialize variables related to exit criteria. 25

14: Lab Report_Yang Yuhang_2289224Y.pdf 100%

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 %%

totErr = 0; % Number of errors observed 26 numBits = 0; % Number of bits processed 27 28 % ---

16 100%

Proceed with simulation. 45 % ---

Be sure to update totErr and numBits. 46 % --- INSERT YOUR CODE HERE. 47 48 %%

16: Lab Report_Yang Yuhang_2289224Y.pdf 100%

Proceed with simulation. % --- Be sure to update totErr and numBits. % --- INSERT YOUR CODE HERE.

Instances from: Wireless and Optical Transmission Systems Report 2289314y PUXI YU.docx

2 66%

xsym = bi2de(reshape(x,k,length(x)/k).','left-msb'); 25

EX1_plot2 =

figure; 26 stem(xsym(1:10),'.','

filled','LineWidth',1.5,'color',[1,0.5,0.5]); 27 title('Rabdom Symbols'); 28 xlabel('Symbol Index'); 29 ylabel('Integer Value') 30

2: Wireless and Optical Transmission Systems Report 2289314y PUXI YU.docx 66%

xsym=bi2de(reshape(x,[k,length(x)/k]).','left-msb');

figure; stem(xsym(1:10));

title('Random Symbols'); xlabel('Symbol Index');ylabel('Integer Value');

10 85%

customised

constellation 1 for QAM 52 %

Describe constellation. 53

inphase = [0 -2 0 2 0 -3 -2 -1]; 54 quadr = [3 2 2 2 1 0 0 0]; 55 inphase = [inphase; -inphase]; 56 inphase = inphase(:); 57 quadr = [quadr; -quadr]; 58 quadr = quadr(:); 59 const = inphase + j * quadr; 60 %

Plot constellation. 61 EX3_plot4 = scatterplot(const, 1, 0, 'o'); 62 hold on; 63

axis([-4 4 -4 4]); 64

10: Wireless and Optical Transmission Systems Report 2289314y PUXI YU.docx 85%

Customised Constellation for QAM'); hold off; %%

Describe constellation. inphase = [1 2 3 0 0 0 2 -2]; quadr = [0 0 0 1 2 3 2 2]; inphase = [inphase; -inphase]; inphase = inphase(:); quadr = [quadr; -quadr]; quadr = quadr(:); const = inphase + j*quadr; Plot constellation. scatterplot(const, 1, 0, '*'); hold on; axis([-3 3 -3 3]); title('Customised Constellation for QAM'); hold off;

title('Customised Constellation

for QAM'); 65 hold off; 66

11 72%

customised

constellation 2 for QAM 72

inphase = [0 1 2 sqrt(2)]; 73

quadr = [1 0 0 sqrt(2)]; 74

inphase = [inphase; -inphase]; 75 inphase = inphase(:); 76 quadr = [quadr; -quadr]; 77 quadr = quadr(:); 78 const = inphase + j * quadr; 79 %

Plot constellation. 80 EX3_plot5 = scatterplot(const, 1, 0, 'o'); 81 hold on; 82

axis([-4 4 -4 4]); 83

title('Customised Constellation

for QAM');

4

exercise 4 10 84 hold off; 85

11: Wireless and Optical Transmission Systems Report 2289314y PUXI YU.docx 72%

Customised Constellation for QAM'); hold off; %%

Describe constellation. inphase = [1 2 3 0 0 0 2 -2]; quadr = [0 0 0 1 2 3 2 2]; inphase = [inphase; -inphase]; inphase = inphase(:); quadr = [quadr; -quadr]; quadr = quadr(:); const = inphase + j*quadr; Plot constellation. scatterplot(const, 1, 0, '*'); hold on; axis([-3 3 -3 3]); title('Customised Constellation for QAM'); hold off;

12 84%

constellation. 91

inphase = [1/2 -1/2 1 0 3/2 -3/2 1 -1]; 92 quadr = [1 1 0 2 1 1 2 2]; 93

inphase = [inphase; -inphase]; inphase = inphase(:); 94 quadr = [quadr; -quadr]; quadr = quadr(:); 95 const = inphase + 1i * quadr; 96 %

Plot constellation. 97 EX3_plot6 = scatterplot(const, 1, 0, 'o'); 98 hold on; 99

axis([-3 3 -3 3]); 100

title('Customised Constellation

for QAM'); 101 hold off; 102

12: Wireless and Optical Transmission Systems Report 2289314y PUXI YU.docx 84%

constellation. inphase = [1 2 0 3/2]; quadr = [0 0 1 3/2]; inphase = [inphase; -inphase]; inphase = inphase(:); quadr = [quadr; -quadr]; quadr = quadr(:); const = inphase + j*quadr; Plot constellation. scatterplot(const, 1, 0, '*'); hold on; axis([-3 3 -3 3]); title('

Customised Constellation for QAM'); hold off; %%

Instances from: Wangpan_2289295_2016200303029.pdf

1	61%

M = 16; 7 k = log2(M); 8 n = 3e4; 9 nsamp = 1; 10 iMin = 0; 11 iMax = 1; 12 x = randi([iMin,iMax],1,n); 13 EX1_

plot1 =

figure; 14

stem(x(1:40),'.','filled','LineWidth',1.5,'color',[1,0.5,0.5]); 15 title ('Random Bits'); 16 xlabel('Bit Index'); 17 ylabel('Binary Value') 18

1: Wangpan_2289295_2016200303029.pdf	61%

M=16; k=log2(M); n=3e4; nsamp=1; x=randi([0,1],n,1); figure; stem(x(1:60),'filled'); title('Random Bits'); xlabel('Bit Index');ylabel ('Binary Value');

Instances from: https://jp.mathworks.com/matlabcentral/answers/4509-index-exceeds-matrix-dimensions

3 66%

Modulation 36 y = modulate(modem.qammod(M), xsym); % using the 16 QAM 37 38 %% Transmissitted

Signal 39 ytx = y; 40 41 %%

Channel 42 % Send signal over an AEGN channel 43

EbNo = 9; 44

snr = EbNo + 10 * log10(k) - 10 * log10(nsamp); 45 ynoisy = awgn (ytx,snr,'measured'); 46 47 %%

Receive channel 48 yrx = ynoisy; 49 50 %%

Scatter plot 51

h = scatterplot(yrx(1:nsamp * 5e3),nsamp, 0,'.'); 52

hold on; 53

EX1_plot3 =

scatterplot(ytx(1:5e3),1,0,'+',h);

2 exercise 2 4 54 title('Received Signal') 55 legend('Received Signal','Signal Constellation'); 56

axis([-5 5 -5 5]); 57

3: https://jp.mathworks.com/matlabcentral/answers/4509-index-exceeds-matrix-dimensions 66%

Modulation

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

%%

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
transmitted
% signal on the same axes.
h = scatterplot(yrx(1:nsamp*5e3),nsamp,0,'g.');
hold on;
scatterplot(ytx(1:5e3),1,0,'k*',h);
title('Received Signal');
legend('Received Signal','Signal Constellation');
axis([-5 5 -5 5]); %
```

Instances from: https://www.tankonyvtar.hu/en/tartalom/tamop412A/2011-0041_introduction_matlab/ch08s09.html

17 96%

Incorporating Gray Coding 49 %

This example, described in the Getting Started chapter of the 50 % Communications Toolbox documentation, aims to solve the following 51 % problem: 52 % 53 % Modify the modulation example (COMMDOC_MOD) so that it

uses 54 % a Gray-coded signal constellation.

code 13 55 % Copyright 1996-2009 The MathWorks, Inc. 56 57 % %

Setup 58 %

Define parameters. 59

M = 16; %

Size of signal constellation 60 k = log2(M); % Number of bits per symbol 61 n = 3e4; % Number of

bits

to process 62

nSamp = 1; %

17: https://www.tankonyvtar.hu/en/tartalom/tamop412A/2011-0041_introduction_matlab/ch08s09.html 96%

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.

% $Revision: 1.1.8.2 $ $Date: 2009/07/20 17:37:55 $

%%

Setup

% Define parameters.

M = 16; %

Oversampling rate 63 64 %%

Create

Modulator and Demodulator 65 hMod = modem.qammod(M); % Create a 16-QAM modulator 66

hMod.InputType = 'Bit'; % Accept bits as inputs 67

hMod.SymbolOrder = 'Gray'; %

Accept bits as inputs 68 hDemod = modem.qamdemod(hMod); % Create a 16-QAM

based on the ...

modulator 69 70 %%

Signal Source 71 % Create a binary data stream as a column vector. 72 x =

randi([0 1],n,1); % Random binary data stream 73 74 %%

Modulation 75 % Modulate using 16-QAM. 76 y =

modulate(hMod,x); 77 78 %%

Transmitted

Signal 79 yTx = y; 80 81 %%

Channel 82 % Send signal over an AWGN channel. 83 %

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);

EbNo = 10; %

In dB 84

SNR =

EbNo + 10 *

log10(k) - 10 * log10(nSamp); 85 yNoisy = awgn (yTx,SNR,'measured'); 86 87 %%

Received Signal 88

yRx = yNoisy; 89 90 %%

Scatter Plot 91 %

Create scatter plot of noisy signal and

ideal constellation points 92 hScatter = commscope.ScatterPlot; % Create a scatter ... plot scope 93 hScatter.Constellation = hMod.Constellation; % Set expected ... constellation 94 hScatter.SamplesPerSymbol = nSamp; % Set oversampling rate 95 hScatter.PlotSettings.Constellation = 'on'; % Display ideal ... constellation 96 update(hScatter, yRx(1:5e3)) % Send received signal ... to the scope 97 %title('

Received Signal'); 98 99 %% Demodulation 100 % Demodulate signal using 16-QAM. 101 z = demodulate(hDemod,yRx); 102

5

%% 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

exercise 5 14 103 %%

BER Computation 104 %

Compare x and z to obtain the number of errors and 105 %

the bit

error rate. 106 [

number_of_errors,bit_error_rate] = biterr(x,

z) 107 108 %%

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)

Instances from: https://core.ac.uk/download/pdf/53187569.pdf

4 99%

hold off; 62 63 %%

Demodulation 64 % Demodulate signal using 16-QAM. 65 zsym = demodulate(modem.qamdemod(M),

yrx); 66 67 %%

Symbol-to-Bit Mapping 68 %

Undo the bit-to-symbol mapping performed earlier. 69

z = de2bi(zsym, 'left-msb'); %

Convert integers to bits. 70 %

Convert z from a matrix to a vector. 71 z =

reshape(z.', prod(size(z)),1); 72 73 %%

BER Computation 74 %

Compare x and z to obtain the number of errors and

the bit 75 % error rate. 76 [

number_of_errors, bit_error_rate] = biterr(x,

z.') 2

4: https://core.ac.uk/download/pdf/53187569.pdf 99%

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.

OUTPUT: [number_of_errors,bit_error_rate] = biterr(x,z)