Lab 2: Digital Modulation and Demodulation

Report Due: 21:00, 4/7, 2017

1 Goal

In this lab, we would like to learn how to implement digital modulation and demodulation. In the first part of the lab, we use LabVIEW to simulate the transmission of 8-PSK and 16-QAM signal constellations over an additive white Gaussian noise (AWGN) channel, along with Gray code mapping bits to symbols. In the second part, we use USRP to transmit data using uncoded modulation over the air. For over-the-air transmission, there are some additional practical issues must be considered, including channel estimation and frame synchronization. To overcome these issues, some methods are provided in this lab. We also evaluate the performance of the implemented system and compare it to the theoretical results.

2 Archietecture and Experiment Setup

This lab includes the modulator and the demodulator (blocks with red texts in Fig. 1).

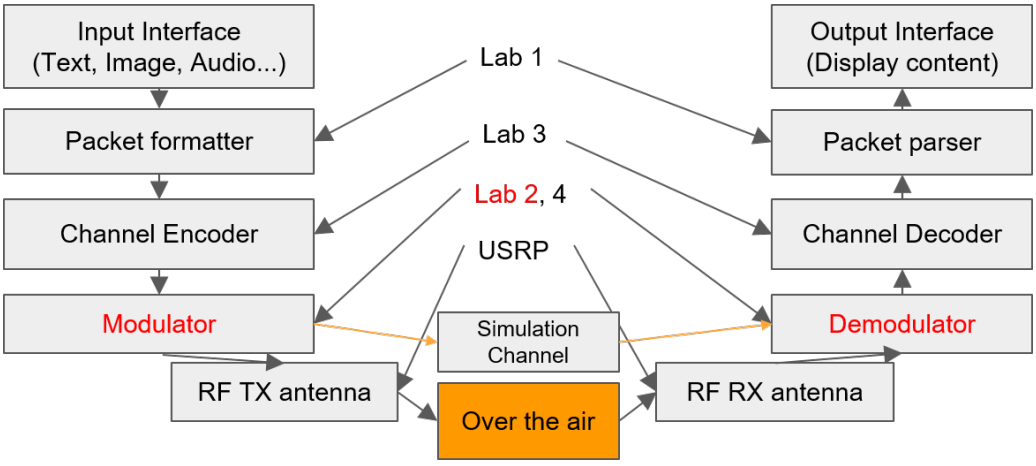

Figure 1: Block Diagram

For the first part of the lab, we can simplify this system to the one depicted in Fig. 2, where bits are mapped into constellation points, depending on the underlying modulation scheme. Then, the transmitted signal is added the AWGN noise $w[n]$. The received signal is then mapped to a constellation point via the specified detection rule. Finally, bit error rate (BER) can be calculated by comparing the message bits with the detected bits. For the second part of the lab depicted in Fig. 3, the main difference is that the signal is transmitted over the air by using USRP.

3 Implementation

In this lab, you will implement two basic digital modulation and demodulation schemes: 1) Phase Shift Keying (PSK) and 2) Quadrature Amplitude Modulation (QAM). These schemes are commonly used in existing technologies such as Wi-Fi and LTE. Similar to previous labs, the templates have already created for you. Namely, `sub_psk_symbol_map_gen.vi`, `sub_qam_symbol_map_gen.vi`, `sub_psk_demod.vi` and `sub_qam_demod.vi`. The rest of this section is organized as follow: Sec. 3.1 introduces the LabVIEW simulator. In Sec. 3.3, we will use the same text transmission system as in Lab 1 to apply your implementation to real-time over-the-air transmission.

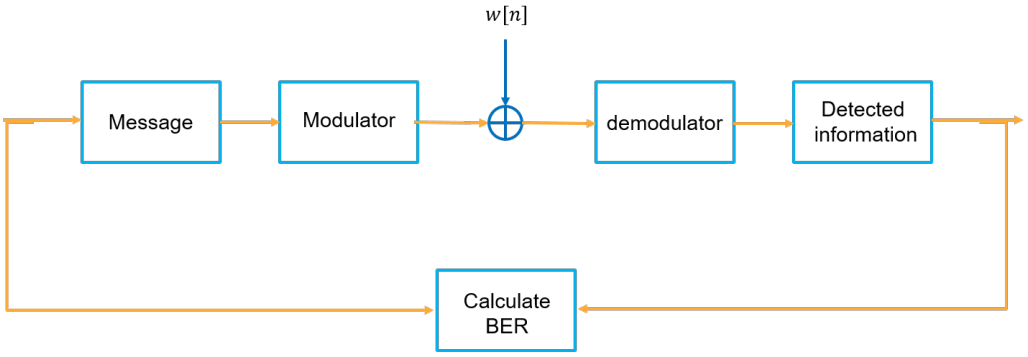

Figure 2: Block Diagram of Simulated Scenario

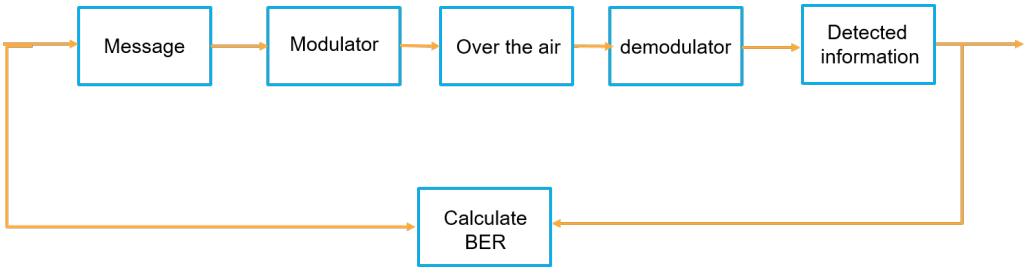

Figure 3: Block Diagram of Over-the-Air Transmission

3.1 LabVIEW Simulation

The goal of this part is to introduce some technical issues when implementing digital modulator and demodulator. First, we will introduce Gray code with some examples. Second, we will show you how to use digital modem simulator to measure the performance of your implementation.

3.1.1 Gray Code

Gray code is a special set of binary numbers. Each element of a Gray code is different from its adjacent codes by one bit. This is an useful property in digital communication and is commonly used in modern communication technologies. Fig. 4 is an example of 8-PSK with Gray code. Also, Gray code can be extended to two-dimensional array. Fig. 5 is an example of 16QAM with Gray code. Note that Gray code is not unique, so there exist permutation of a Gray code which is also a Gray code. In this lab, we will implement the PSK and QAM modulator and demodulator with Gray code.

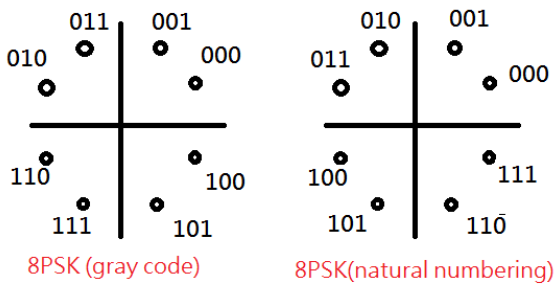

Figure 4: 8PSK with gray code and natural numbering

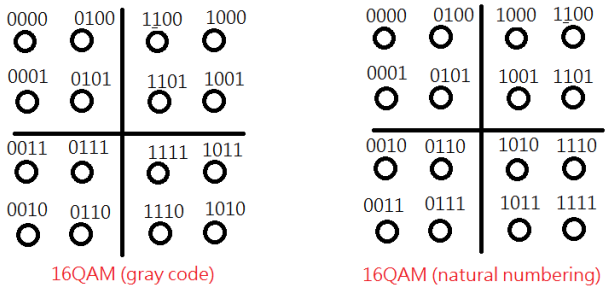

Figure 5: 16QAM with gray code and natural numbering

3.1.2 Digital Modem Simulator

As you learn in the lecture, BER curve is a commonly used performance measure of modulation and demodulation (Fig. 6). Therefore, the purpose of this simulator is to make it easy for you to generate such performance measure. The simulator generate bits randomly and use AWGN noise model to simulate radio noise. You can change the parameter "$E_b/N_0(dB)$" to configure the SNR and "M-ary", "Modulation type" to select the modulation of interest. We will give some results of this simulator as references. Before you start, we have to learn how modulator and demodulator work in this simulator.

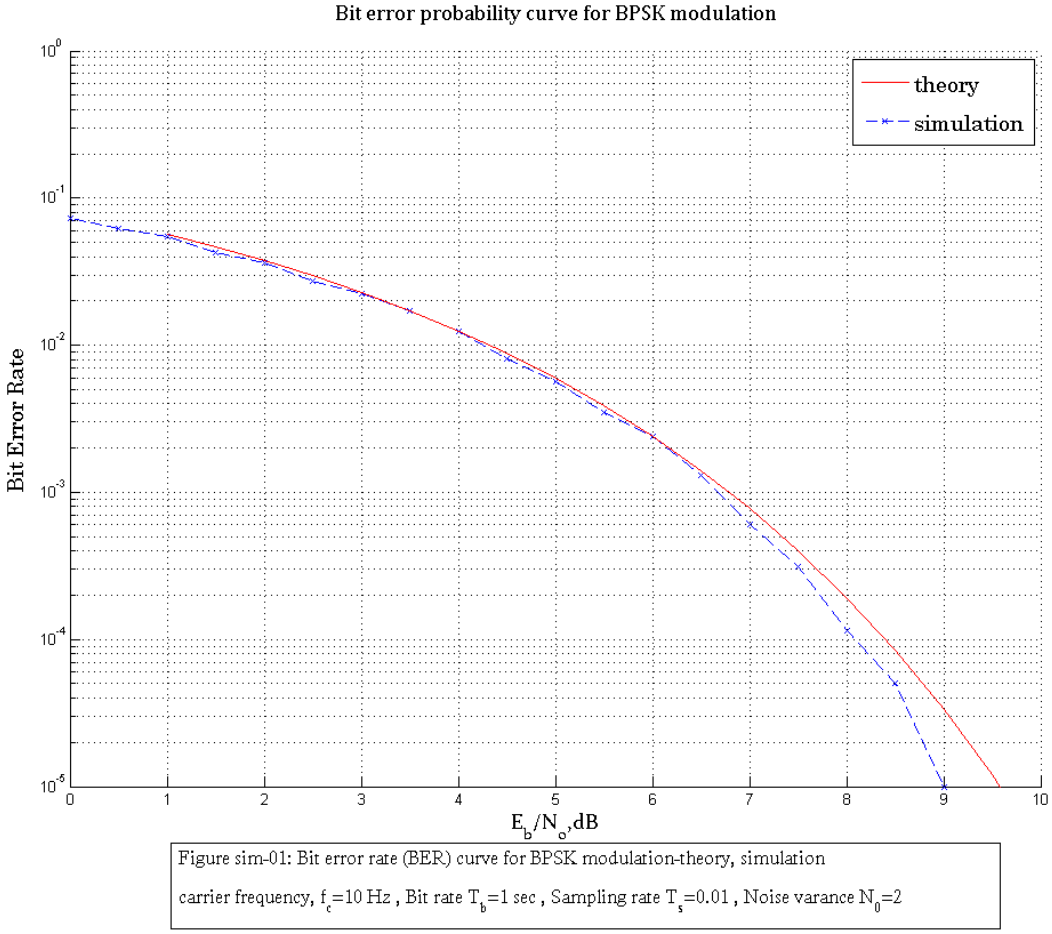

Figure 6: BER curve example, BPSK

Modulator To reduce computation time, modulation is done by table-lookup method. Therefore, to implement modulator, our job is to generate PSK and QAM constellation map. As for the demodulator, it also needs the constellation map to turn symbols back to bits. The procedure of a modulator is as follow:

1. **Bits grouping**: Suppose the message bitstream is (LSB)"101101000"(MSB) and the modulation is 8PSK. Since each 8PSK symbol is represented with 3-bits (Fig. 4), we can separate the message bitstream to 3-bits tuples: "101,101,000". You can append zeros to make the number of bits be divisible by the number of bits per symbol.
2. **Table lookup:** The next step is to turn bits tuples to constellation symbols. To do this, each tuple is represented in decimal format. Therefore the tuples: "101,101,000" is equivalent to "5,5,0". Then we use these decimal numbers to find the represented constellation point.
3. **Output symbols:** The last step is to concatenate the symbols generated in step 2 to an array.

Demodulator Demodulator is more complex than the operation of modulator. Since the received symbol is corrupted by noise, the received symbol will deviate from the original constellation point. In this case, the demodulation problem is equivalent to a detection problem. You can show that in maximum likelihood detection, the demodulation of QAM is to find the closest symbol on the constellation map to the received symbol. For PSK, the decision criterion is specified by phase regions, and the final decision is made according to the phase region in which the received symbol falls (Fig. 7).

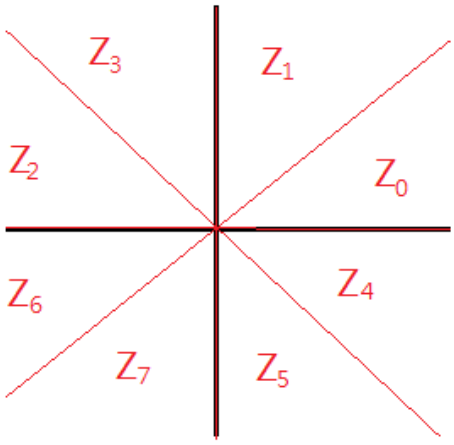

Figure 7: Decision region for 8PSK with Gray code

1. **Decision making:** As explained previously, the decision criterion of PSK and QAM are different. For M-PSK the constellation map is divided into M regions according to the phase of each symbol. An example of 8-PSK is shown in Fig. 7. For QAM, the decision making criterion is minimum Euclidean distance: calculate the distance between M-QAM symbols and received symbol and choose the closest one to be the final decision.
2. **Mapping symbols to bits:** After a decision is made, the demodulated symbol can be related to an index on the constellation map, and by using constellation map, the index can be turned back to bits.

Assignment Implement the following modules: PSK symbol map generator, PSK demodulator, QAM symbol map generator and QAM demodulator. The modules you need to implement are placed in folder named "subVIs". Please complete them before using the simulator. The following are some notes for implementing these modules:

- **PSK:** The inputs of the symbol map generator are: 1)M-ary, and 2)Gray code. The output sare:1)symbol map and 2)decimal symbol map. For the details please refer to `sub_psk_symbol_map_gen.vi`.
- **QAM:** The input of the symbol map generator is similar to that of PSK, while in M must be square number. For example: 4, 16, 64...etc.

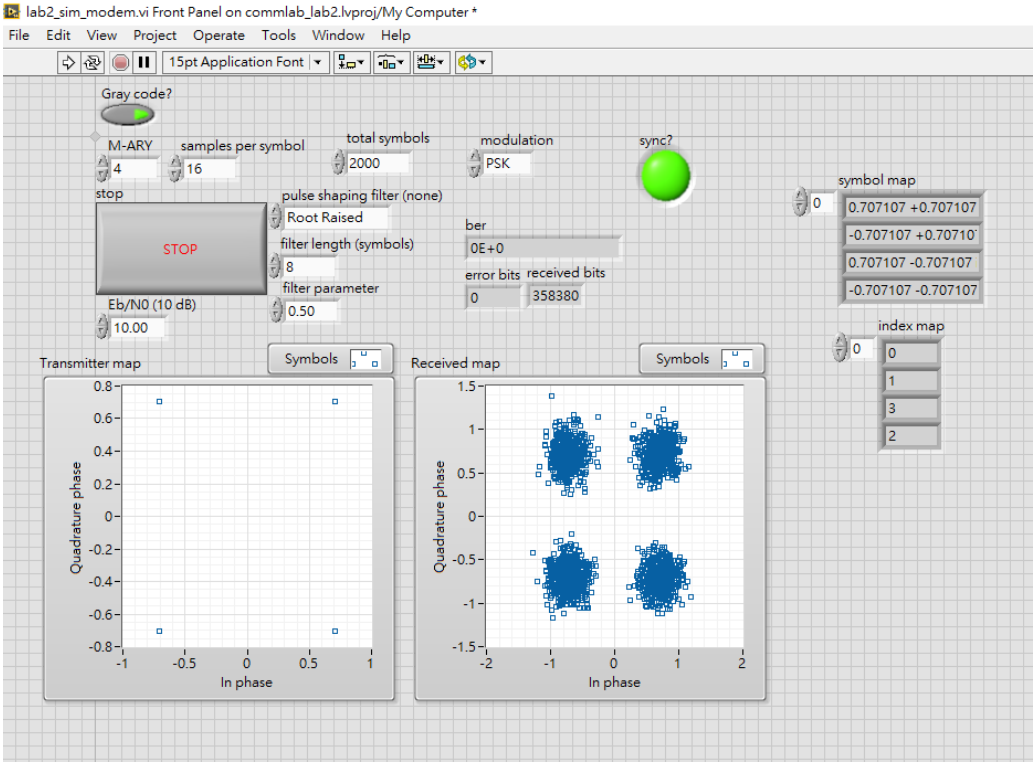

Figure 8: Interface of Modem Simulator

3.2 Modem Simulator

After we complete the implementation of modulator and demodulator, we can use simulation to see the performance of PSK or QAM. First, we will generate message bits randomly. Then we map bits to symbols according to constellation map. After that, the symbols are added with AWGN noise. Finally, the noisy symbols are passed to the demodulator and turned back to bits. We can compare the transmitted and received bits and see if there is error. The following are some parameters of the simulator and Fig. 8 is the interface of it.

- *Modulation:* The type of modulation to run the simulation. It can be either PSK or QAM.
- E_b/N_0 *(dB):* The SNR of the system.
- *Gray code:* Turn on this button to generate constellation map with Gray code. Otherwise, turn off it.
- *M-ary:* Specify the number of points in modulation. Note that for QAM it must be even power of 2.
- *total symbol:* The number of symbols per iteration of the simulator.

3.3 USRP Implementation

In this part, we will apply the modulator and demodulator to the system we used in Lab 1. Furthermore, we will be faced with some practical issues in digital communication, namely, channel estimation and sampling time offset correction, and learn how to solve them.

3.3.1 Channel Estimation

In wireless communications, the preamble is used to combat channel impairments such as channel distortion, carrier frequency offset, and phase rotation. These are some practical issues for communication system to function successfully in real world. In this lab, we will not cover all of these issues. Instead, we will focus on channel estimation issue. Channel estimation is required since the real world channel is time varying and hard to predict. In Lab 1, we introduce a packet header, which composed of preamble, packet length, and header counter. The preamble is a known sequence of bits to both transmitter and receiver, which makes it a suitable

helper to estimate channel. We begin with a simple case, where the channel effect is modeled as a complex scalar.

$$r[n] = \alpha x[n] + w[n]$$

where $r[.]$ is received symbol, α is the unknown channel coefficient, $x[.]$ is preamble symbol, and $w[.]$ is AWGN noise. If the SNR is high enough (i.e. $\mathbb{E}\left[|x\left[.\right]|^2\right] \gg \mathbb{E}\left[|w\left[.\right]|^2\right]$) the scalar channel coefficient can be approximated as:

$$\hat{\alpha} = \mathbb{E}\left[\frac{rx^*}{|x|^2}\right]$$

In practice, the expectation is calculated with sample mean:

$$\hat{\alpha} = \frac{1}{N}\sum_{n=0}^{N-1}\frac{r\left[n\right]x^*\left[n\right]}{|x\left[n\right]|^2}$$

This estimate can also be written as follow:

$$|\hat{\alpha}| = \left|\frac{1}{N}\sum_{n=0}^{N-1}\frac{r\left[n\right]x^*\left[n\right]}{|x\left[n\right]|^2}\right|$$

$$\arg\hat{\alpha} = \arg\frac{1}{N}\sum_{n=0}^{N-1}\frac{r\left[n\right]x^*\left[n\right]}{|x\left[n\right]|^2}$$

The attenuation and phase shift can be corrected accordingly.

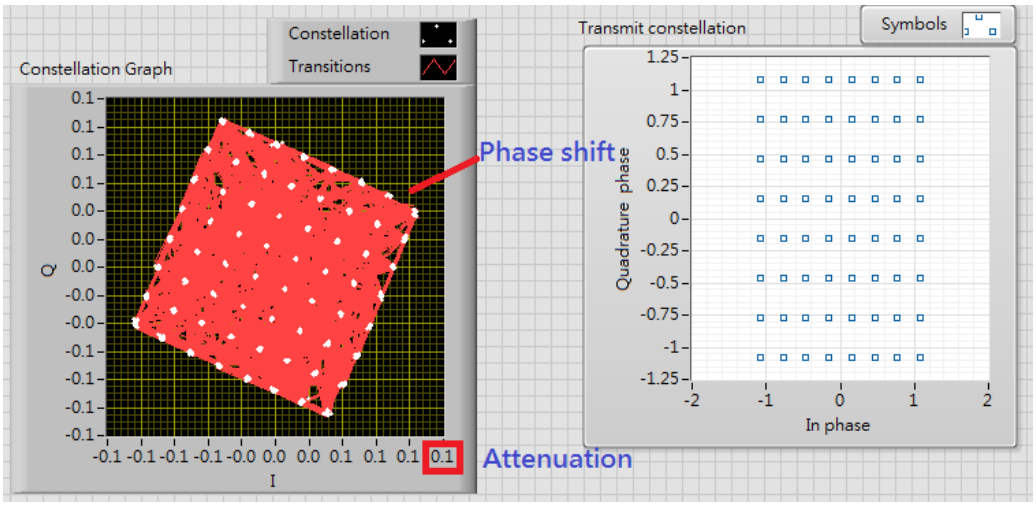

Figure 9: Example of channel distortion

3.3.2 Frame synchronization

After channel estimation, the question follows is how to find the preamble of a packet with received symbols. We can also utilize the preamble to complete this task. Since transmitter and receiver must negotiate which modulation scheme to use, the symbol-level preamble can be generated accordingly. We can correlate symbol-level preamble to received symbols. This is similar to cross correlation. In theory, the absolute value of cross correlation will be the highest when reaching the position of preamble. We can set a threshold and declare a preamble has been found if the correlation is greater than this threshold. Mathematically, the cross-correlation is computed as follow:

$$C[n] = \frac{\sum_{i=0}^{N-1} r[n+i]s^*[i]}{\sqrt{\sum_{i=0}^{N-1}|r[n+i]|^2}\sqrt{\sum_{i=0}^{N-1}|s[i]|^2}}$$

$C[.]$ is the normalized cross-correlation, $r[.]$ is the received symbol, and $s[.]$ is preamble symbol. The correlation threshold $\lambda \in [0, 1]$. As a result, we declare the preamble is found when $C[n] \geq \lambda$ at some index n.

3.3.3 Text Transceiver

To implement the text transceiver, we reuse the system in Lab 1. We combine the components introduced in Sec. 3.3.1–3.3.2 and replace the modulator and demodulator with the one we implemented. The control panel of the system is similar to that in Lab 1. We have additional controls, including: "Gray code" and "modulation type". Also, there is a monitor for you to observe the power spectrum. Please refer to `lab2_usrp_packet_transceiver.vi`.

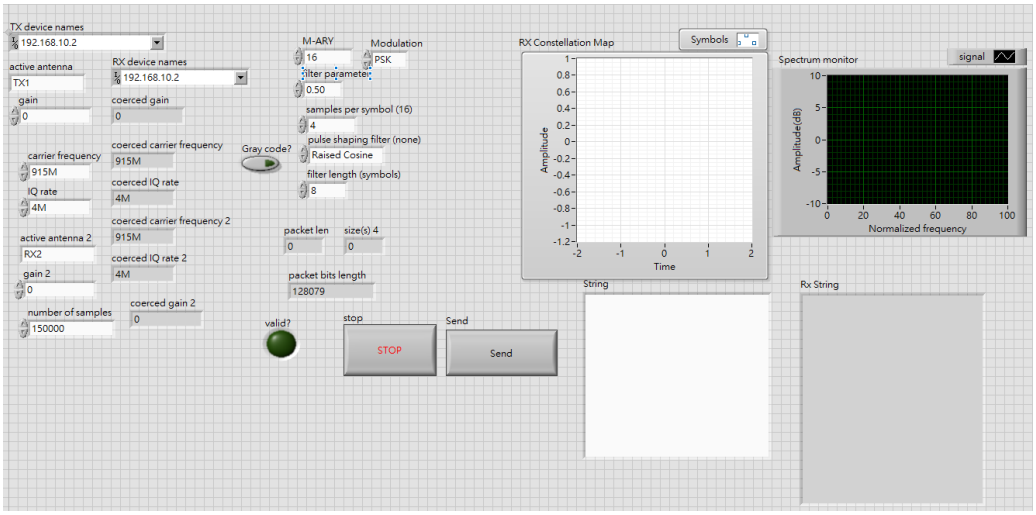

Figure 10: Interface of text transceiver

Assignment Implement the channel estimation and frame synchronization modules: `rx_ceEq.vi`.

4 Exercises

4.1 LabVIEW simulation

1. Build your 8PSK and 16QAM simulation program using LabVIEWblocks. Plot SER and BER versus $\frac{E_b}{N_0}$ curves of 8PSK and 16QAM in AWGN channel. The curves should start from $\frac{E_b}{N_0} = 0$ all the way to the $\frac{E_b}{N_0}$ that generates the BER of 10^{-5}. You should have no less than 10 simulation points spreading evenly on the curve.

2. Search for analytical SER/BER versus $\frac{E_b}{N_0} = 0$ curves in books, literature, or internet. Plot those on the same figure along with your curves to verify the correctness of the simulation results. You should specify where you find those curves in the figure caption. The sources of the curves should be creditable. (You can also derive the SER/BER curves on your own, and please include your derivations in the report.)

3. (Optional) Try to derive an approximation for the BER bound from the SER union bound. Plot it along with your BER curves and comment on how good is your bound.

4.2 USRP implementation

1. Build your 8PSK and 16QAM USRP implementation. Plot SER, BER versus transmit power curves of 8PSK and 16QAM by using USRP. The curves should start from "gain = 0(dB)" all the way to the "gain" that generates the BER of 10^{-5}. You should have no less than 5 simulation points spreading evenly on the curve.

2. Observe RX constellation map and record RX string under different transmit power.

3. Block the line-of-sight between the two antennas. You will find that receiver power instead of transmit power is the right factor that affects the performance. Try to record the received power and replace transmit power by received power when re-doing 1 and 2. Discuss what material is the best blockage.

5 Lab Report

There is no format requirements for your lab report. In the report, you should address the results of the exercises mentioned above. You should also include your simulation program in the appendix of the report. Include whatever discussions about the new findings during the lab exercise, or the problems encountered and how are those solved. Do not limit yourself to the exercises specified here. You are highly encouraged to play around with your simulation program on self-initiated extra lab exercises/discussions.