Lucas Ghanem
Jason Chiarulli
Steven Arose

April 18, 2019

QAM Simulation Report

Introduction:

Quadrature Amplitude Modulation (QAM) allows for multiple amplitude modulated signals to be combined into a single channel. Although QAM sends analog amplitude/phase modulated signals, digital QAM can be performed by assigning symbols to specific ranges of amplitudes and phases read by a receiver. These symbols refer to specific packets of digital information represented in bits. Additive White Gaussian Noise (AWGN) can be added to the transmitted QAM signals in the channel. Depending on the amplitude and phase modulation due to the significance of the AWGN, an incorrect symbol may be read by the receiver. In this project, students were to simulate the transmission of a jpg picture with AWGN through the use of QAM signals.

Method/Results:

The first goal was to generate 3 functions in matlab to simulate a transmitter to send both binary and gray mapped M-QAM signals, a channel with AWGN, and a receiver which converts the read M-QAM signals with AWGN to packets of bits.

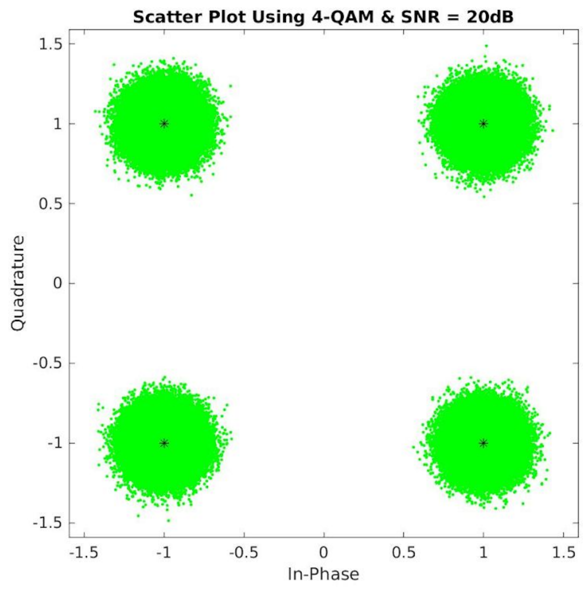
First, in order to simulate a transmitter, we required only 2 inputs: the jpg filename and the M value. Using the Matlab function *imread*, we were able to read the image file. Then using the MatLab function *de2bi*, we were able to convert the image file which was represented as a matrix of decimal values to a matrix of binary values. The image data was then converted to a column vector using the MatLab function *reshape*. The function *reshape* was also used to represent the vector of bits as a vector of packets of bits. This was done by reshaping the vector so that it reads k bits per position of the vector where k is the number of bits per symbol

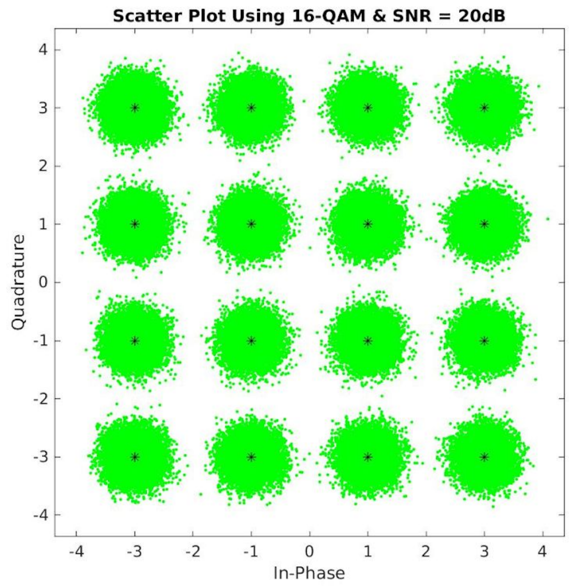
($k = \log_2(M)$). The MatLab function *qammod* was then used to simulate the transmission of the QAM signal for both binary and gray coding. The function returned the modulated data for both binary and gray coding as well as the input bit stream which was represented as a column vector.

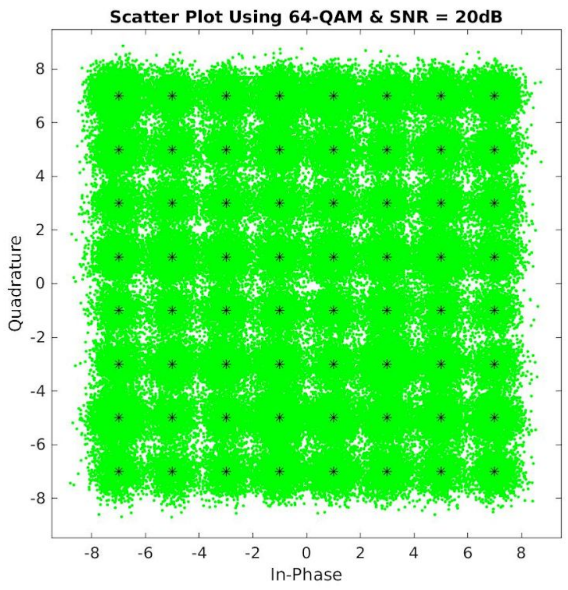
Next, simulating the channel which adds AWGN was simple. The function would read both the gray and binary coded QAM signals as well as the specified value of signal-to-noise (SNR) in decibels. The MatLab function *awgn* was used to add the noise to both signals with the given Signal to Noise Ratio (SNR). The function then returned both the binary and gray coded received signals.

Finally, the receiver function required to both convert the QAM signals to a vector of bits and reconstruct the image. The function would input the gray and binary signals, the M value, and a variable which would determine whether or not the reconstructed image would be shown when the function was called. The MatLab function *qamdemod* was used to convert the signals back to vector form. The vector of bits was converted to a matrix of decimal values in order to convert each element to a number between 0 and 255. The bit stream for the gray coded received signal was then converted to matrix with a row size of eight and each row was then converted to a decimal number between 0 and 255. This matrix was then converted back to the original image size by reshaping it to a 180x180x3 matrix. The values in the matrix were then converted back to the original image data type of uint8. The function *imwrite* was used to write the reconstructed image to a file. The function would then return the received image using the Matlab function *imshow* and it would also return the output bit streams for both the gray coded and binary signals.

The next step was to provide a scatter plot of the QAM mapped signals with AWGM. This was done in between the channel and received steps on the transmission process so that we could view the mapped QAM signals (from the transmission stage) with AWGM (from the channel stage). This was simply performed by using the MatLab function *scatterplot* for $M = \{4,16,64\}$.


Scatter Plot Using 4-QAM & SNR = 20dB
Quadrature
In-Phase
1.5
1
0.5
0
-0.5
-1
-1.5


Scatter Plot Using 16-QAM & SNR = 20dB
Quadrature
In-Phase
4
3
2
1
0
-1
-2
-3
-4



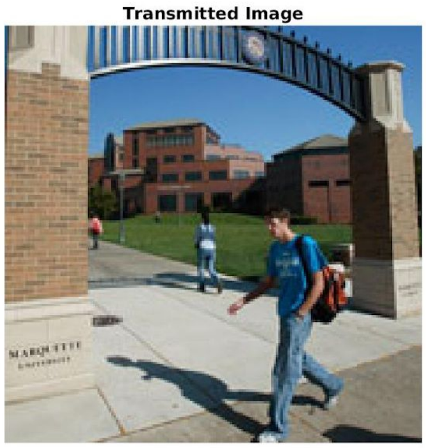
It is evident from the scatter plots that 4-QAM is prone to the least amount of errors since the received signal bits are clustered around their respective symbols with no overlap. Using 16-QAM also results in very few errors since the received signal bits are once again clustered around their respective symbols with very little overlap. Using 64-QAM results in thousands of errors because the received signal bits are clustered around their respective symbols; however, the overlap between the bits is very large because there are many more symbols. This means when the received signal bits are being mapped the distance between the symbol it should get mapped to is actually greater than the distance between a neighboring symbol, so the received signal bits get mapped to the neighboring symbol because the mapping is determined by the minimum distance between the received signal bits and the each symbol..

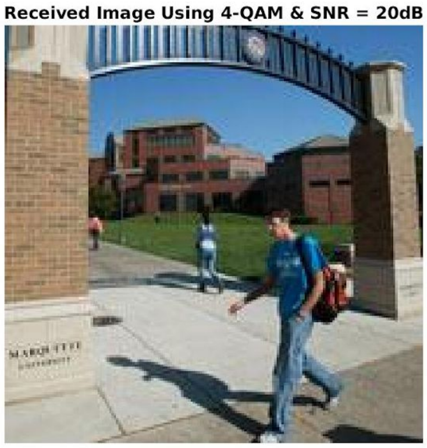
In order to compute the probability of error after decoding, the received data (from *qamdemod*) is compared with the transmitted data (from the vector of packets in the transmission phase). Since the received data was reconstructed with AWGN, there is expected to be some error. To check if a bit does not match that of the input bit, each bit of the input vector would be subtracted by the corresponding bit in the output vector. In order for no error to occur, all of these values would be equal to 0; however, if error occurs, a value of -1 or 1 would show up. The number of times a 0 didn't show up was counted in a for-loop to compute the number of errors. The number of errors was divided by the length of the data vector to calculate the overall simulated probability of error.

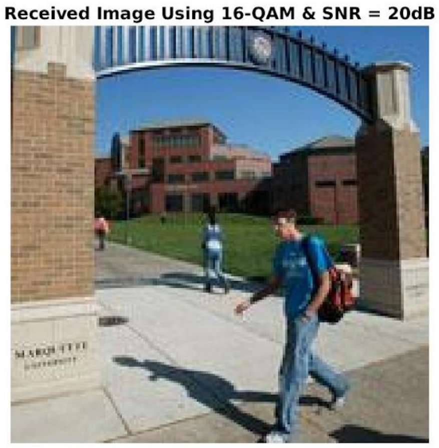
```
The Gray coding bit error rate = 0.00e+00, based on 0 errors for M = 4 & SNR = 20dB

The binary coding bit error rate = 0.00e+00, based on 0 errors for M = 4 & SNR = 20dB

The theoretical binary coding bit error rate = 0.00e+00, for M = 4 & SNR = 20dB


The Gray coding bit error rate = 3.86e-06, based on 3 errors for M = 16 & SNR = 20dB

The binary coding bit error rate = 7.72e-06, based on 6 errors for M = 16 & SNR = 20dB

The theoretical binary coding bit error rate = 2.90e-06, for M = 16 & SNR = 20dB


The Gray coding bit error rate = 9.13e-03, based on 7102 errors for M = 64 & SNR = 20dB

The binary coding bit error rate = 1.56e-02, based on 12159 errors for M = 64 & SNR = 20dB

The theoretical binary coding bit error rate = 8.38e-03, for M = 64 & SNR = 20dB
```

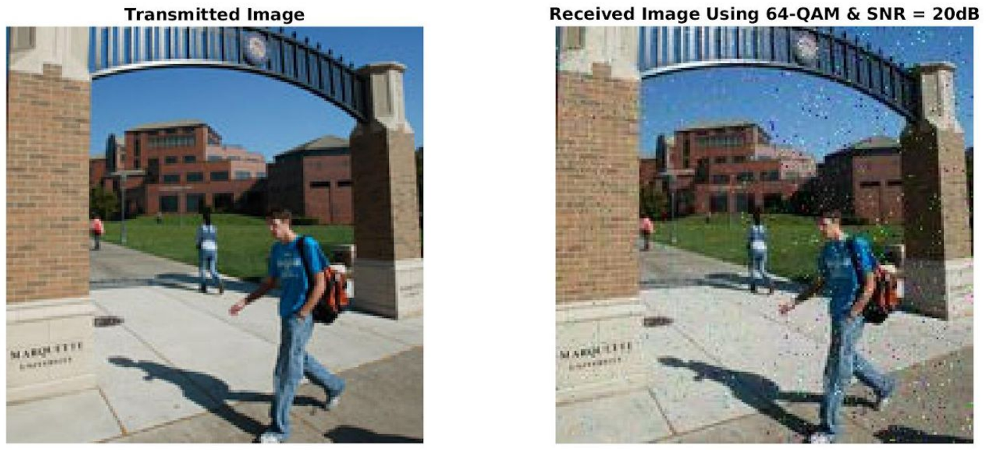
From the bit error rate output it is obvious that when M is increased the bit error rate and number of errors increase for Gray coding, binary coding, and the theoretical binary coding. The bit error rate for Gray coding is also lower than the binary coding for 16-QAM and 64-QAM which is caused by the fact that when using Gray coding there is only a one bit difference between the symbols. The bit error rate for 4-QAM is zero for all three methods though which is caused by the way the received signal bits are clustered around their corresponding symbols for 4-QAM. This can be seen in the scatter plot for 4-QAM above.

As stated before, the image was reconstructed in the receiver function. The 3 reconstructed images are shown below:









It is clear from the image comparison of the transmitted image with each reconstructed image that 4-QAM results in the most accurate reconstruction. The reconstructed image using 16-QAM is also very similar to the transmitted image, and the reconstructed signal using 64-QAM results in a very inaccurate reconstruction. This was the expected result which was based off of the the scatter plots and the computed bit error rate.

For different values of SNR, we chose values ranging from -6 to 12 dBs. These values were simply sent into our transmission, channel, and receiver functions. The figures of the probability of error were constructed using the calculated probability of error for each SNR value, using the method stated previously, and plotted using *semilogy*. This was done for both gray and binary mapping for M = {4, 16, 64}. We computed the probability of bit error as well as the probability of a symbol error. In order to compute the theoretical probability of error we used the formulas provided in class. The mathematical formulas used to generate the theoretical expected probability of bit error:

$$k = log_2(M)$$

$$\varepsilon_b/N_0(dB) = SNR - 10log(k)$$

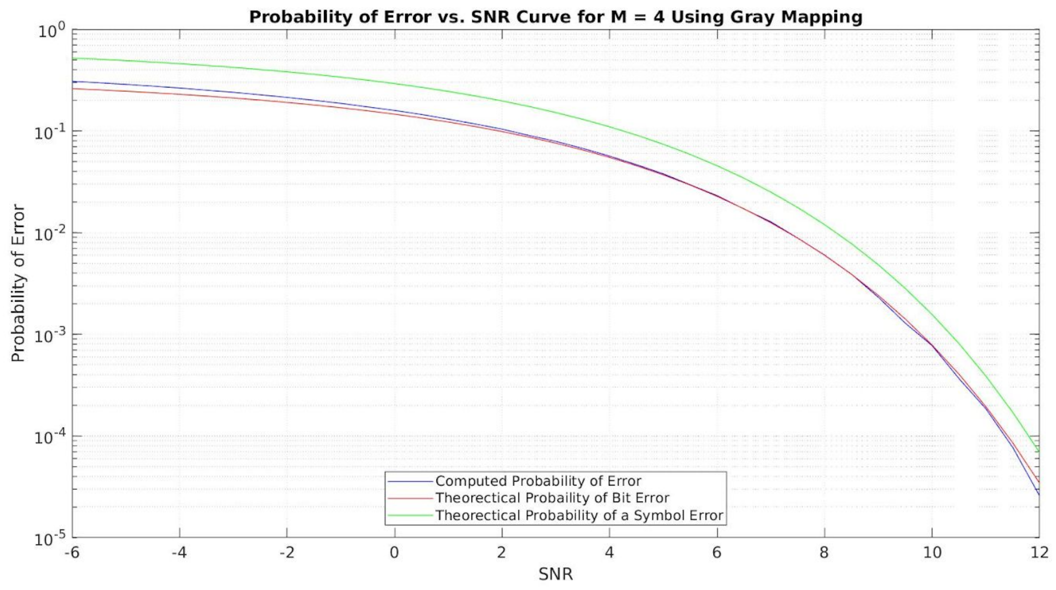
$$\varepsilon_b/N_0 = 10^{(SNR - 10log(k))/10}$$

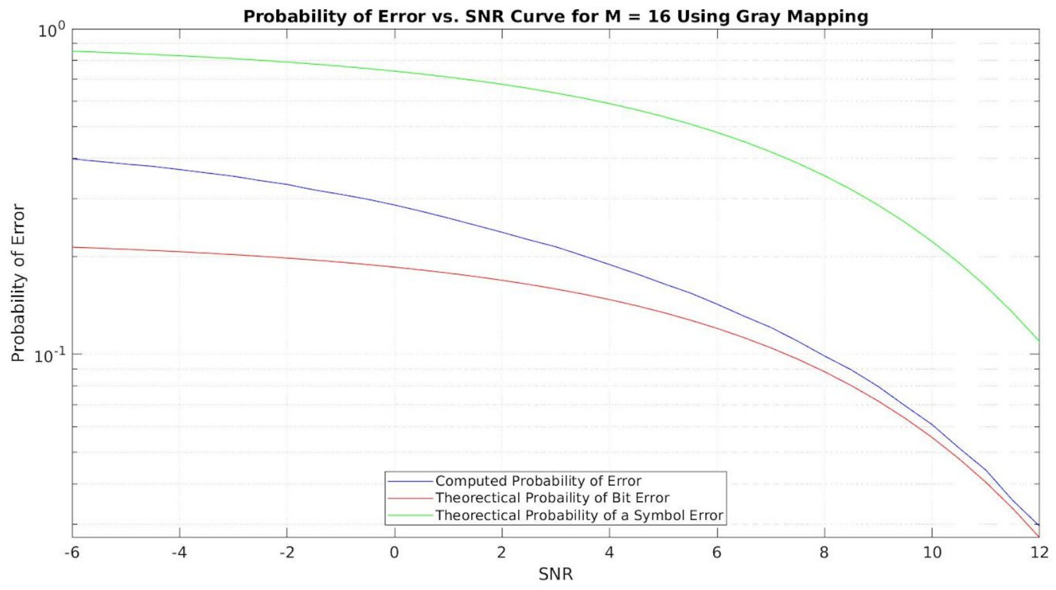
$\varepsilon_{av} = \varepsilon_b k$

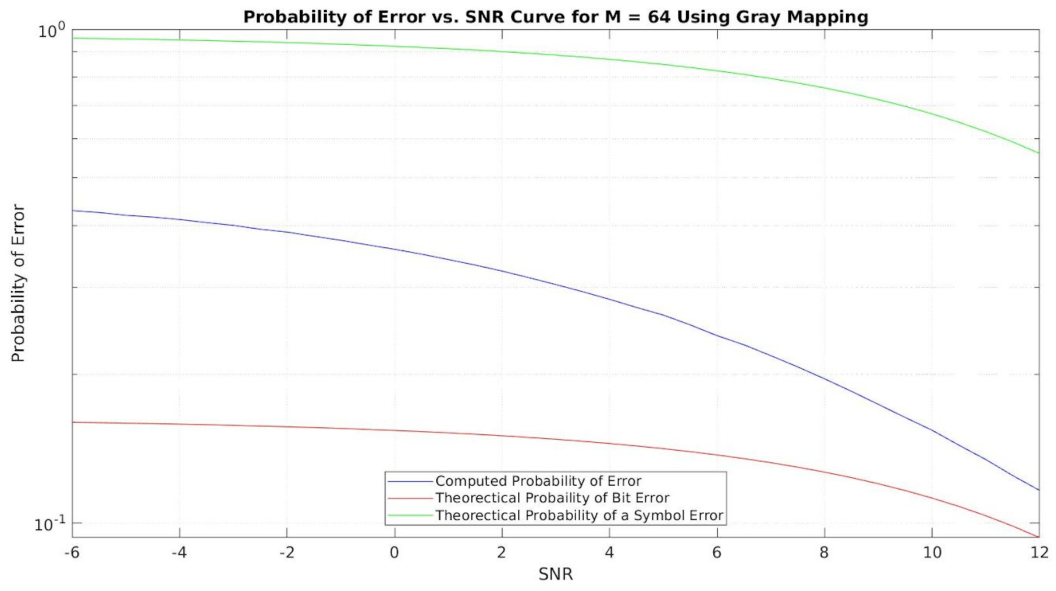
$P_{\sqrt{M}} = 2(1 - \frac{1}{\sqrt{M}})Q(\sqrt{\frac{3\varepsilon_{av}}{(M-1)N_0}})$

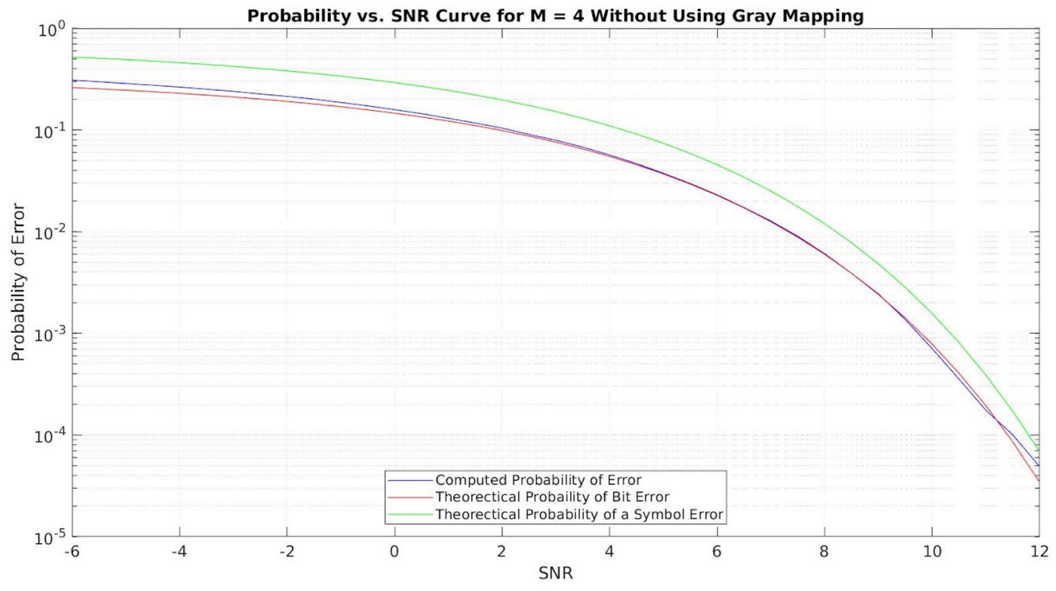
$P_M = (1 - (1 - P_{\sqrt{M}})^2)/k$

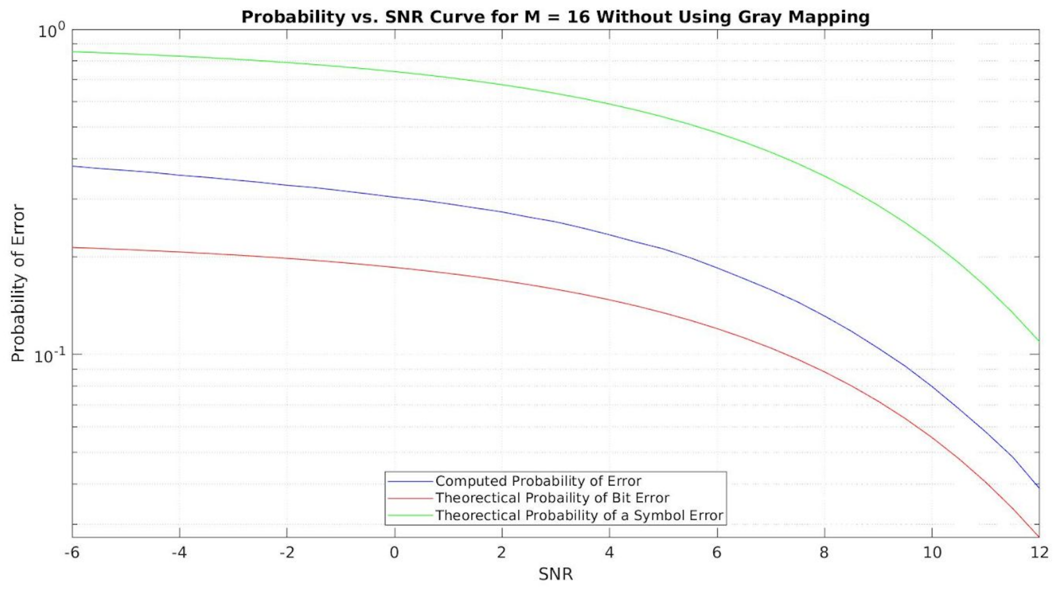
Now for the probability of a symbol error, we used the same method as the theoretical probability of bit error, but with P_M not being divided k. The graphs of our results are below:

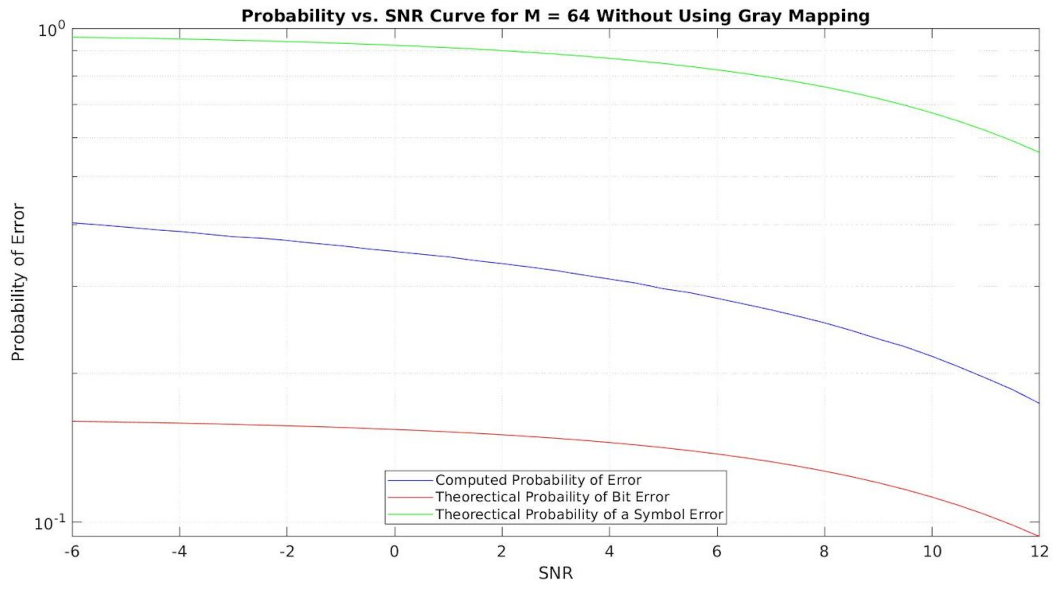





Probability of Error vs. SNR Curve for M = 64 Using Gray Mapping
Probability of Error
SNR
10^0
10^{-1}
-6
-4
-2
0
2
4
6
8
10
12
Computed Probability of Error
Theorectical Probaility of Bit Error
Theorectical Probability of a Symbol Error


Probability vs. SNR Curve for M = 4 Without Using Gray Mapping
Probability of Error
SNR
10^0
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
-6
-4
-2
0
2
4
6
8
10
12
Computed Probability of Error
Theorectical Probaility of Bit Error
Theorectical Probability of a Symbol Error





It is evident from the graphs that the theoretical probability of bit error and the computed probability of error for 4-QAM with and without the use of Gray mapping are very similar. The probability of error is also very small and decreases as the SNR increases. For 16-QAM the theoretical probability of bit error and the computed probability of error are also very similar with and without the use of Gray mapping; however, the computed probability of error is a little closer to the theoretical probability of bit error when the Gray mapping is used. The probability of bit error both theoretical and computed and the theoretical probability of a symbol error are higher than the values obtained for 4-QAM which is to be expected, and once again the probability of error decreases as the SNR increases. For 64-QAM the theoretical probability of bit error and the computed probability of error are closer when Gray mapping is used. The probability of bit error both theoretical and computed and the theoretical probability of a symbol error are higher than the values obtained for 4-QAM and 16-QAM which is once again to be expected. Also, the probability of error once again decreases as the value of the SNR increases.

Gray mapping allows for a smaller probability of error simply due to the way the bits are mapped to the M-QAM signals. Without gray, the bits are sorted in a binary fashion, e.g., (00, 01, 10, 11). The neighboring packets, 01 and 10, differ by 2 bits. This means that potentially 2 errors in bits could be detected in the case that a single wrong packet is received. If sorted with gray mapping, the neighboring packets would only differ by 1 bit, e.g., (00, 01, 11, 10). Now, if a neighboring packet is received, there will only be a single error in bits rather than potentially more than one. This ultimately leads to a lower bit error rate (BER).

Conclusion:

We found this project to be a great way for us to visualize what we have been learning in class. Comparing gray and binary mapping let us understand why the arrangement of the mapping of packets is important for minimizing the expected error in QAM transmissions. It was also very interesting being able to see the actual distortion of the image when it was reconstructed. Although high M-QAM transmissions allow for more data to be sent per symbol, the project also made it very clear why this can cause a drastic increase in error when we analyzed the scatterplots. Calculating the theoretically expected error was difficult, but it really helped us make sense of the dependency on both the M values and the SNR values. This helped us understand that while SNR is extremely high, 64-QAM may cause no error while when SNR is low, 64-QAM can lead to very significant error. All in all, we found this team project to really make sense of what was learned in class and we found it really exciting to apply to a simulation.

References

1. http://www.dsplog.com/2008/06/05/16qam-bit-error-gray-mapping/