EE233 Wireless Communication Final Report

Ching-Yen Chung, Alex McAuley and Yubo Wang

1. Introduction

In this final report, we will summarize all previous projects and connect sub-blocks to form a complete 802.11 communication system. The system includes transmitter, channel model and receiver for 1x1, 1x2, and 2x2 cases.

Fig. 1.1 shows a block diagram of the simulated system. On the transmitter side, 802.11 standard header is added to raw data. Then the data stream is modulated with 16-QAM. As the emphasis of this course is not on channel coding, we ignore the interleaver and any error correcting codes. At a last step in the transmitter, the data stream is up-converted to 2.4GHz and transmitted through the antenna(s). The channel is another important component in the simulation wherein AWGN, Rayleigh fading and multipath channels are all simulated and studied in this report. On the receiver side, the data stream is first down-converted. Then it is fed into the AGC block, in which block boundary and package detection are also performed. The output of the AGC block is forwarded to the carrier frequency and channel estimation blocks. The carrier frequency block calculates the offset of the carrier frequency, while the channel estimation block provides an estimate of the channel response. With accurate channel estimation, demodulation can be implemented. The performance of the system is evaluated using BER as a metric. One point worth mentioning is that this simulation ignores some 802.11 blocks such as the stream parser and interleaver, but most of the major OFDM blocks are covered in the simulation. SIMO and MIMO cases are also examined to see how much improvement is made relative to the SISO case.

The report is organized as follows: Section 2 reviews the basic 802.11 data structure. It is followed by channel modeling in Section 3. Section 4 details modulation and demodulation scheme. Sections 5, 6 and 7 introduce AGC, frequency detection and channel estimation, respectively. 1×2 and 2×2 cases are studied in section 8. System level simulation results are given and analyzed in Section 9. Finally, discussion of the results and conclusions are drawn in Section 10.

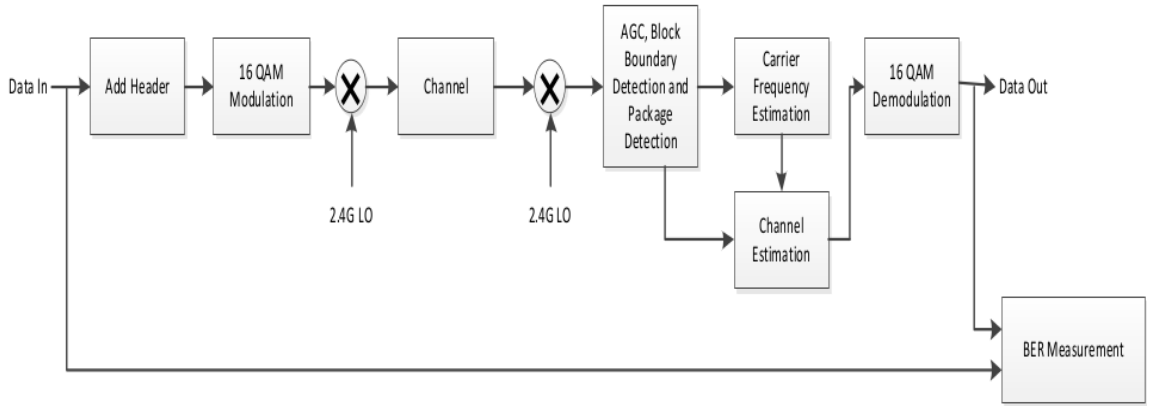


Figure 1.1: Simulation Diagram

2. 802.11 Data Structure

The standard mixed mode 802.11a/b/g/j/n data structure is shown in Fig. 2.1 [1].

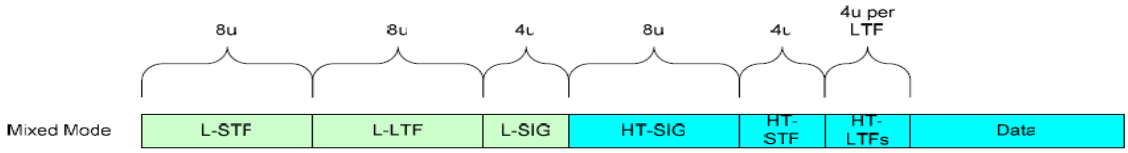


Figure 2.1: 802.11 data structure in mixed mode

To generate L-SIG and HT-SIG segments, which contain information about the data, the encoder, the interleaver, and the mapper need to be implemented. Since these elements are not used in the simulation, the data structure is simplified by ignoring L-SIG and HT-SIG as shown in Fig. 2.2.

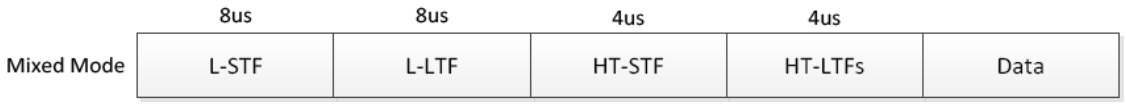


Figure 2.2: Simplified mixed mode 802.11 data structure

The Legacy Short Training Field (L-STF) is used for Automatic Gain Control (AGC), packet detection and coarse frequency detection, which will be described more detail in the following sections. The 0.8us symbol is repeated 10 times to form an 8us length of L-STF signal. The role of Legacy Long Training Field (L-TF) is for fine frequency detection and channel estimation, which will be discussed in the following sections. The 3.2us symbol is repeated 2 times with a 1.6us GI ahead of it. The “HT” in HT-STF and HT-LTF's stands for High Throughput; therefore, these two signals are used in Multiple Input Multiple Output (MIMO) system. Similar to the L-LTF's, HT-LTF's are used for MIMO channel estimation.

In the EWC HT PHY Specification, the header and data are generated using an Orthogonal Frequency Division Multiplexing (OFDM) system with 64 sub-channels. Only 52 of the 64 sub-channels carry data. 8 of the remaining 12 sub-channels, which are 1~4, 33, and 62~64, does not carry any data. The remaining 4 channels are for pilot signal which are labeled as channel 12, 26, 40, and 54. Fig. 2.3 shows illustration of the sub-channels distribution.

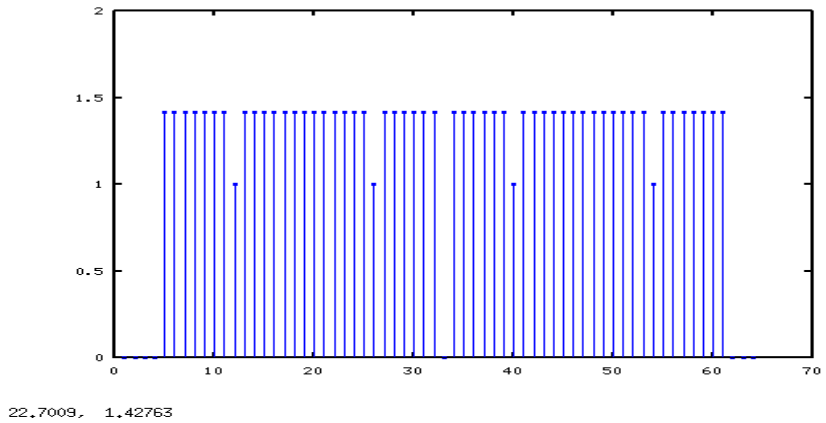


Figure 2.3: Sub-channels distribution

Following the EWC HT PHY Specification, the header of the simplified mixed mode 802.11 data structure (Fig. 2.2) is generated, a time-domain representation of which is shown in Fig.2.4. It can be observed that the signal in the L-STF field repeats 10 times while the signal in L-LTF field only repeats twice.

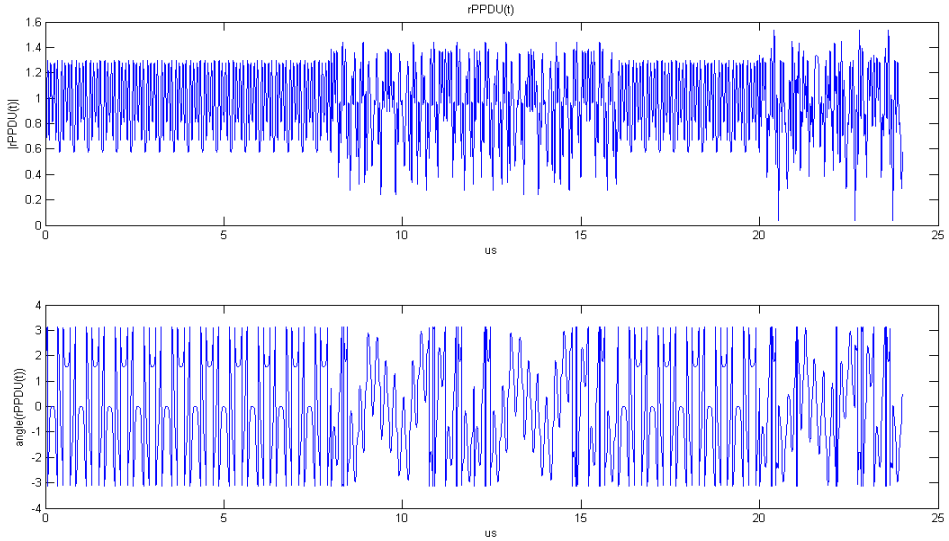


Figure 2.4: Generation of 802.11 packet header

3. Channel Modeling

In this section, we describe our channel model for AWGN, Rayleigh flat fading and multipath channels. In our simulation, we assume the channel and all subsequent blocks are sampled at 1GHz. This is a very large value and is somewhat unrealistic. It was initially chosen to ensure good signal resolution, and then later blocks used it because they were designed with the previous blocks in mind. We realize an actual system would probably operate closer to a 20MHz sampling rate, but it would have taken a massive amount of work to redesign all of the blocks to work correctly at a lower sampling rate. This choice led to some problems with the AGC later on, but other than being an impractically high sampling rate for real-world ADC's, our 1GHz sampling did not prevent us from successfully simulating the system. Still, if we could start over, we would have chosen a more reasonable sampling rate in order to better adhere to real life and also to speed up our simulations.

3.1 AWGN Channel

Fig 3.1 shows the simulation diagram for the AWGN channel. The AWGN channel is the simplest one. Desired SNR is generated based on TX power. Yet in the following sections readers will see AWGN channel has the best performance.

Figure 3.1: Simulation diagram of AWGN channel

3.2 Rayleigh Fading Channel

Fig. 3.1 shows the block diagram for Rayleigh flat fading channel modeling. The figure clearly shows how 3 components: Rayleigh fading, path loss and shadowing interact with each other. Desired SNR is generated using Tx power and path loss.

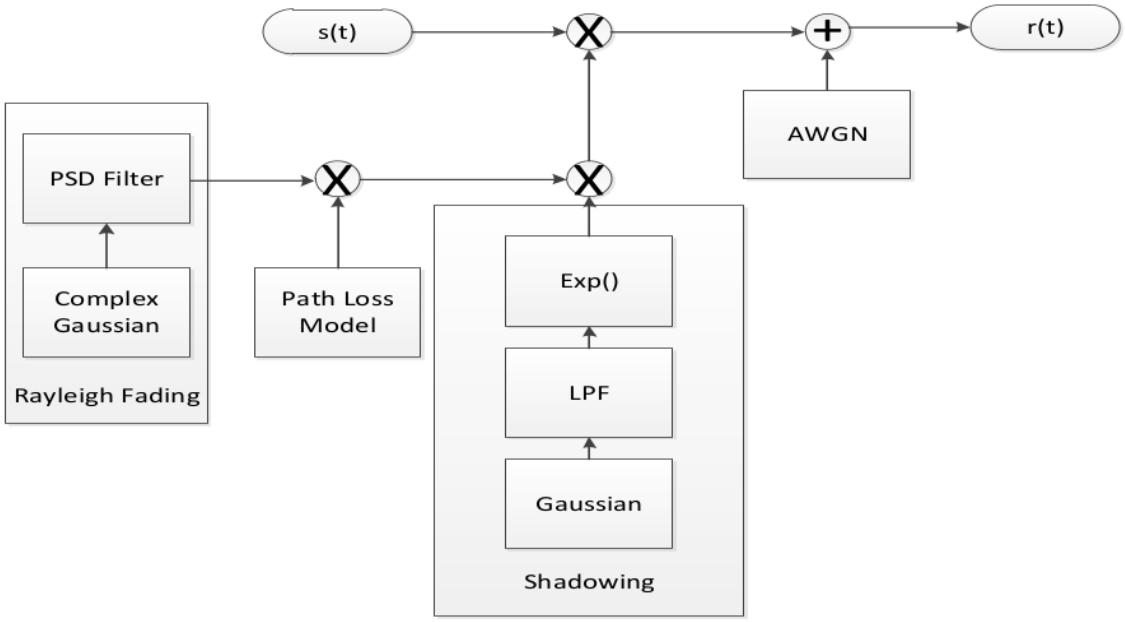


Figure 3.2: Simulation diagram of Rayleigh flat fading channel

Figures 3.3 and 3.4 clearly show the simulation results of amplitude and phase of Rayleigh fading. From these figures below, readers can see that the amplitude is Rayleigh distributed and phase is uniformly distributed.

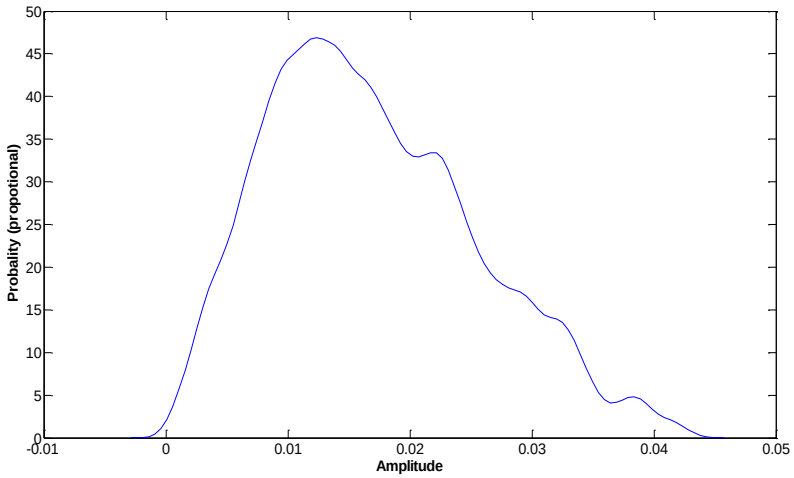


Figure 3.3: PSD of the amplitude of a complex Gaussian

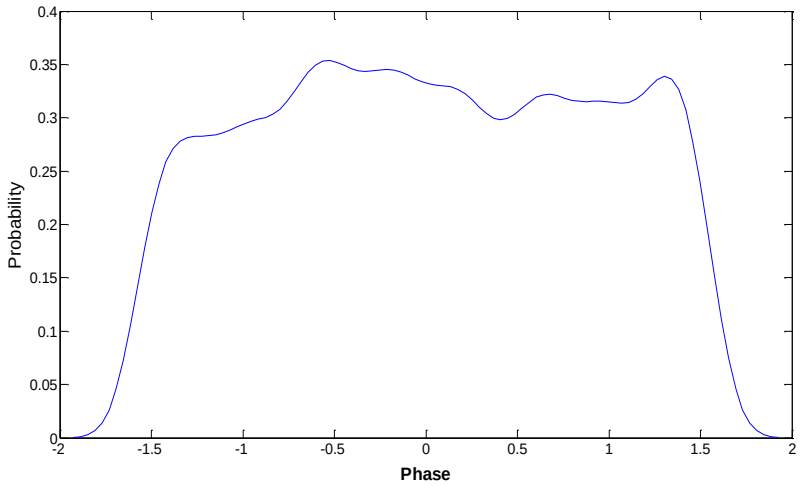


Figure 3.4: PSD of the phase of a complex Gaussian

Figures 3.5 and 3.6 show the results of the Rayleigh filter output at different Doppler frequencies. From the figure readers will notice that the filter cuts off exactly at Doppler frequency. One point worth mentioning is that most of the energy falls at the Doppler frequency. Therefore, most of the signals coming out of the channel will have frequency close to the Doppler frequency.

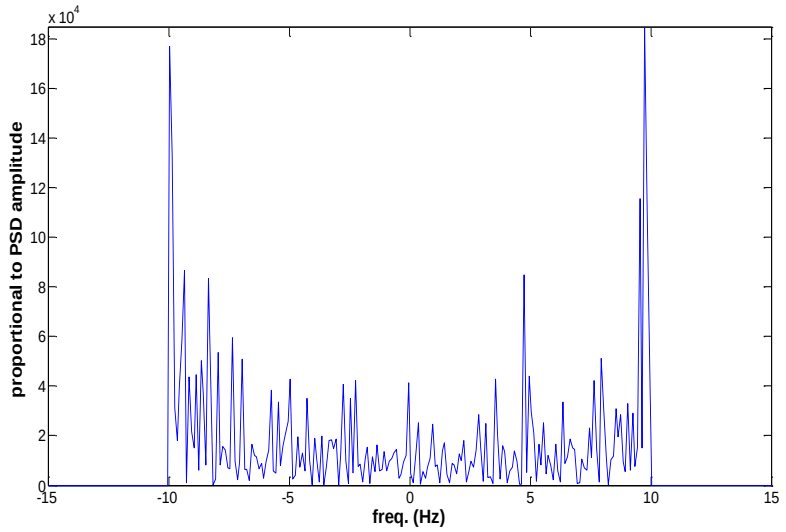


Figure 3.5: PSD for Rayleigh fading at FD = 10 Hz

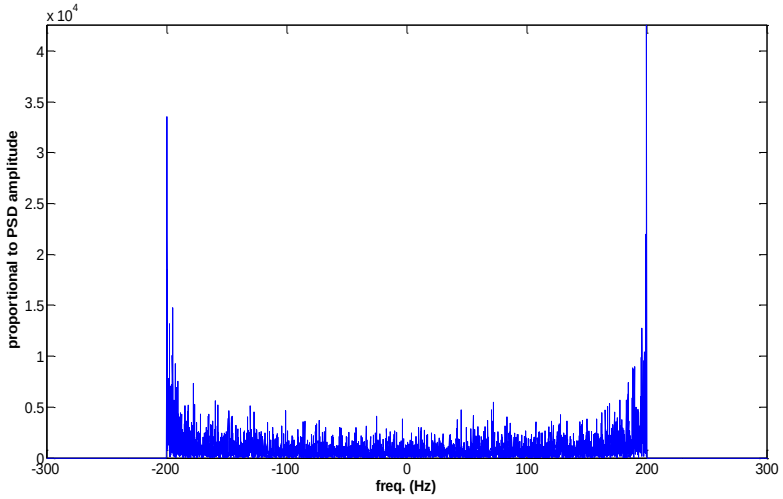


Figure 3.6: PSD for Rayleigh fading at FD = 200 Hz

In order to see fading effects in our packet, we had to use a slightly modified channel model. The entire 802.11 packet's duration is on the order of 10us to 100us, so fading with a Doppler frequency of 200 Hz or below would never be noticed and our BER curves would look just like the regular AWGN case. When curves in fading were called for, we simulated Rayleigh fading by using a fixed-amplitude scalar with complex Gaussian noise added to it so that each sample would vary in amplitude and phase around a fixed magnitude. The fixed scalar was included because without it there every sample would have a completely random phase, which would throw off the channel estimation and everything following it. An alternative approach would have been to generate a very long payload so that each packet would actually notice 200Hz fading, but that approach was discarded due to obscenely long runtime that would have resulted in. With faster code (especially a faster AGC block and lower sampling rate), using a long data payload would have been an appealing choice. Nevertheless, as seen in section 9, our slight modification seemed to perform a decent job of creating fading effects.

We had difficulty writing code that could successfully estimate how much noise to add to get a given SNR in the presence of multipath. To ensure accurate noise levels were used during BER simulations, signal power was measured for the flat fading case, and all channel parameters were then frozen in all subsequent simulations and the measured power was then hard-coded into the AWGN function. This way multipath interference wouldn't look like extra signal power when calculating how much noise to add for a given SNR.

3.3 Multipath Channel

Fig 3.7 shows the simulation diagram of a channel model with *n* multipath components. Each path is first fed to an amplitude scaling block which controls the relative amplitude in each path. The scaled output is fed into a delay. Both the scaling and delay amounts are user-input parameters that describe the channels of interest. Next, each path undergoes flat fading as described in the previous section (except in the AWGN-only case). Finally the multipath components are added together to give a complete channel response. In this report, we deal with both 2-ray multipath and 4-ray multipath models. For the 2-ray model, two equal multipath components are present at 0ns and 50ns. For the 4-ray case, multipath with relative power of 0, -3, -6, -10 dB at 0, 70ns, 150ns and 200ns respectively are

considered.

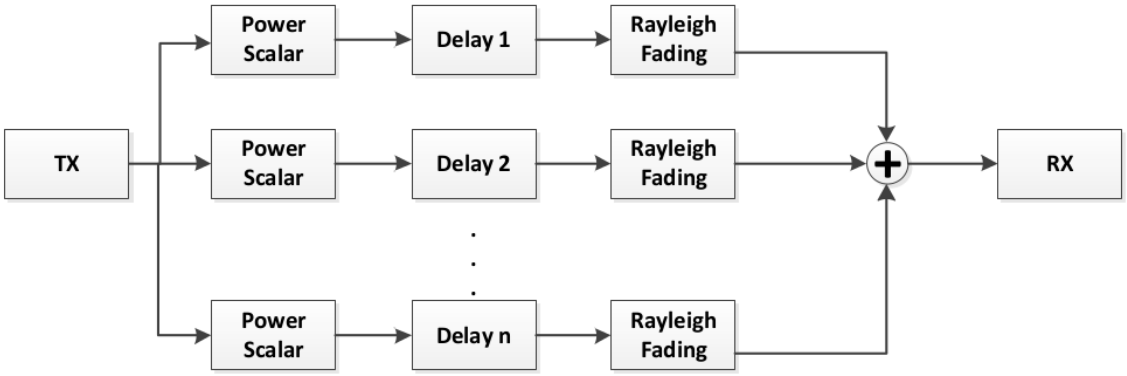


Figure 3.7: Simulation diagram of n multipath

4. Modulation and Demodulation

The simulation uses a grey-coded 16-QAM constellation to modulate data onto each of the packet's data subcarriers. The EWC specification did not list the specific constellation to use with modulation and demodulation, so the grey-coding scheme was chosen to minimize BER for a given symbol error rate. The point labeling and values were assigned according to the diagram in Fig. 4.1, created by Krishna Sankar and obtained from [2].

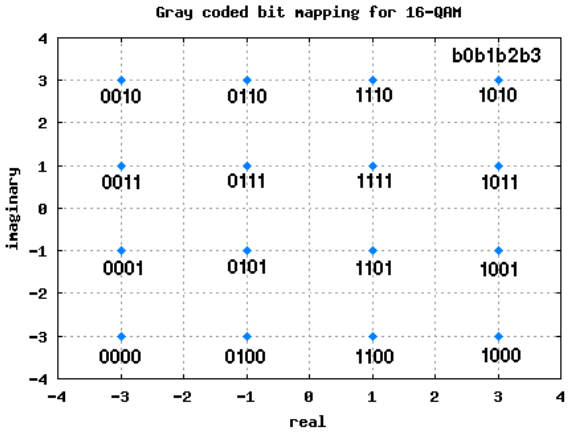


Figure 4.1: 16-QAM constellation

4.1 Encoder

The encoder performs the mapping of bits to symbols by working on blocks of 208 bits (4 bits/sym x 52 sym/OFDM data block) at a time. Each input block is divided into groups of four successive bits and then those groups of four are converted into symbols. This straightforward modulation approach does not utilize any type of interleaving between subcarrier frequencies. In an actual system, interleaving between frequencies cold allow for lower bit error rates if an error correcting code were used. After taking the IFFT of the OFDM data block, a 0.8us guard interval is inserted after the symbol in order to protect against inter-OFDM-symbol interference. Including the guard interval, a total of 80 symbols sampled at 20 MHz leaves the encoder for each input block of 208 bits.

4.2 Decoder

At the decoder, the data payload is divided into groups of 80 symbols. The symbols in the guard interval are added to the beginning of the received OFDM symbol according to Fig. 12.8 of Goldsmith. The FFT of the resulting 64 symbols is taken, and the symbols are decoded according to the maximum likelihood criterion.

5. AGC, Packet Detection, and Block Boundary Detection

The AGC, packet detector, and block boundary detector are treated as a single block because of how closely interconnected they are. As the AGC operates, it feeds its output to the package detector. The output of the package detector is monitored by the block boundary detector in order to determine where the end of the STF segments occurs.

The biggest problem with the AGC was its excruciatingly slow run time. The slow runtime was due to the use of multiple for loops operating on thousands of points of data. Using a slower sampling frequency would have helped speed up the program, but only a complete rebuild of the function could hope to make it as quick as the other functions. To give an idea of how slow it was, generating each of the plots in the Results section took anywhere from 30min to an hour. When the actual AGC was bypassed as part of testing, each curve could be generated in a matter of seconds.

5.1 AGC

The purpose of the AGC is to scale the received signal, which could range over many orders of magnitude, to a fixed range of amplitudes. Signals larger than the ADC input range should be avoided in order to prevent clipping and the associated distortion. Signals that are too small, on the other hand, will lead to distortion due to ADC quantization noise and suffer from poor BER due to low signal energy. Due to the nature of noise, it is impossible to eliminate the possibility of clipping, so the goal of our AGC design was to scale the received signal close to the maximum range of the ADC while experiencing clipping less than 1% of the time. Additionally, the AGC gain should have a sufficiently fast rise time so as to stabilize within 3us of a packet's arrival. Our AGC settings accomplish the first two goals by scaling the received signal to greater than half the ADC input range of +/-1V (units are assumed to be Volts), but not scaling it further. This keeps the clipping rate very low, and the input still utilizes most of the ADC's range (within 1 bit of it's maximum resolution). The actual AGC is composed of two sub-circuits: the saturation limiter and the AGC proper (“Main AGC”).

5.1.1 AGC: Saturation Limiter

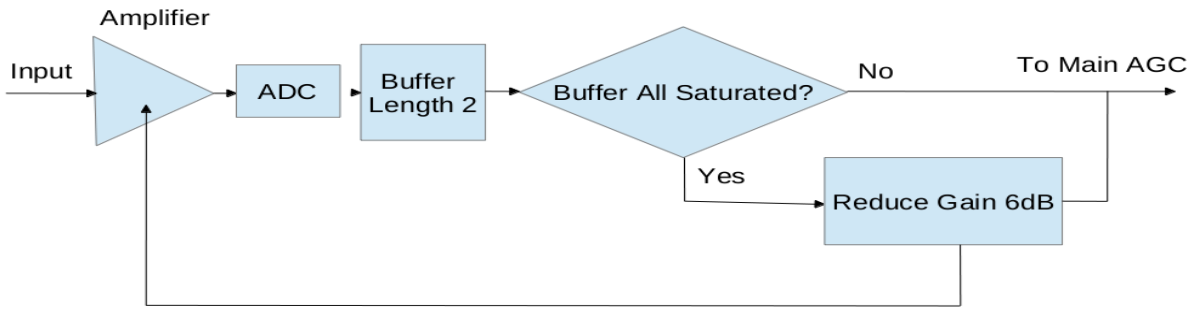


Figure 5.1 Saturation limiter

The saturation limiter block diagram is shown in Fig. 5.1. Its purpose is to aggressively reduce receiver gain when the input consistently saturates. For our simulation, consistent saturation means two or more saturated samples in a row, where saturation occurs whenever the In-phase or Quadrature branch is outside the +/-1V range. The buffer is needed so that the input gain does not suffer a large jump due to noise pushing a single received point beyond the ADC limits. Such a sudden jump would force the main AGC (see next section) to re-stabilize its gain when there was no real problem to begin with. Buffer length longer than 2 is unnecessary. During normal operation, two successive symbols are very unlikely to saturate due to noise. In our simulations, the saturation limiter would perform its duty within the first few samples of the input, within a few tens of nanoseconds.

5.1.2 AGC: Main AGC

The main portion of the AGC takes its input from the saturation limiter and finely adjusts it, usually by increasing the gain, until the signal reaches a specified average power level. A block diagram of the main AGC is shown in Fig. 5.2.

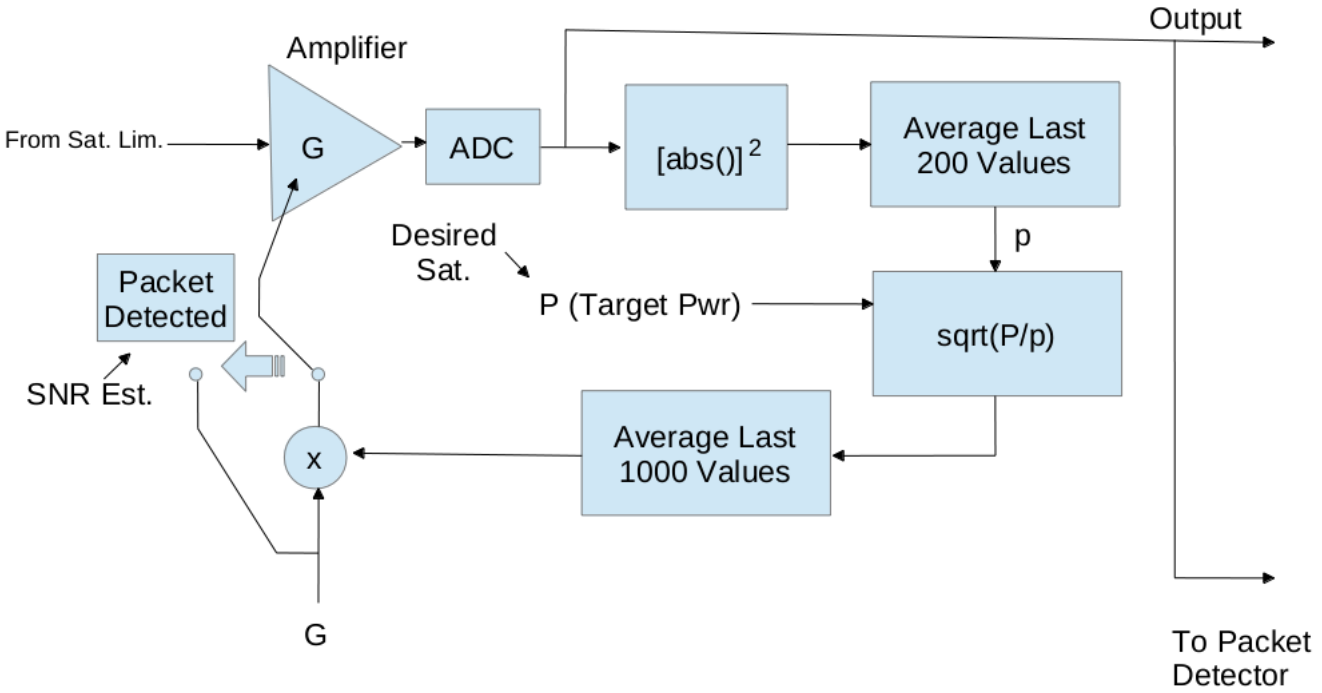


Figure 5.2: Main AGC

It would have been possible to design an AGC feedback loop based on signal amplitude instead of power, but the power-based implementation was simpler because using an amplitude-based AGC would require a separate AGC for both the I and Q channels since each have their own ADC to worry about. This would create extra hardware or software overhead in the design. Communication between the two AGC's would be particularly important because if the gains weren't synchronized, the recovered constellation would be stretched in one direction. One could argue that the amplitudes

could be combined in quadrature before being fed into a single ADC, but that is essentially the same as the power-based solution presented above.

Our AGC implementation averages the power of the 200 most recently received values, *p*, and adjusts the amplifier gain so as to drive the received power towards the setpoint *P*. The setpoint during simulation was 0.35, which corresponded to saturation less than 0.9% of the time down to 5dB SNR, which was the lowest SNR of interest for the project. 1000 *P* values are averaged to smooth out the AGC's performance, and the amplifier's gain is adjusted until the packet detector declares that a packet has been detected. After packet detection, the gain is frozen for 30us to ensure its stable until the end of the data payload.

Figures 5.3 and 5.4 show the performance of the saturation limiter and AGC combined with package detection and block boundary detection. These figures will be discussed further in the next sub-sections, but at this point it can be noted that the magnitude of the AGC output (red) increases until it freezes after 2us (cyan line indicates gain freeze, blue horizontal line is 10% of the AGC gain). The output amplitude average is approximately 0.6V, and saturation occurs less than 1% of the time. The input amplitude was lower than the ADC saturation threshold, so the saturation limiter did not have to kick in (black).

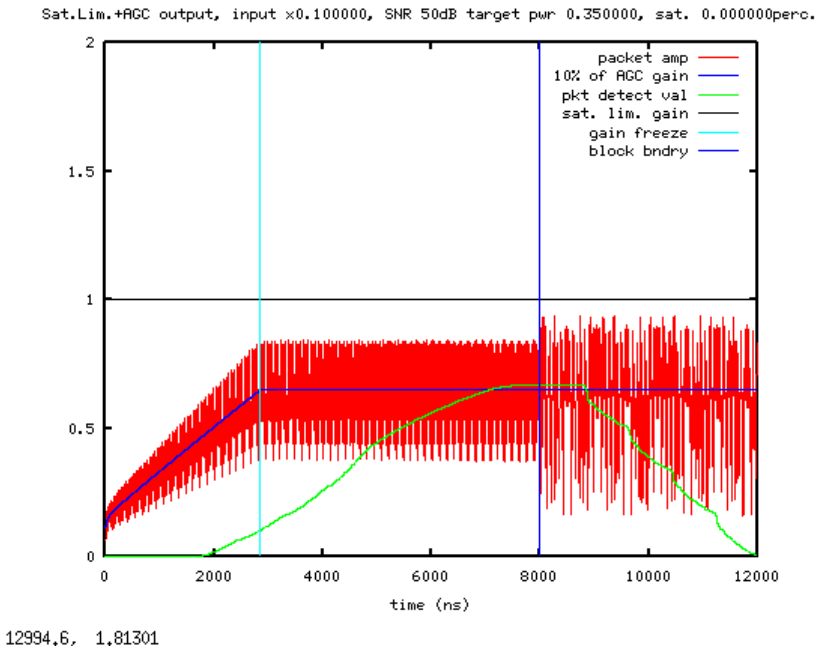


Figure 5.3: AGC performance, 50dB SNR

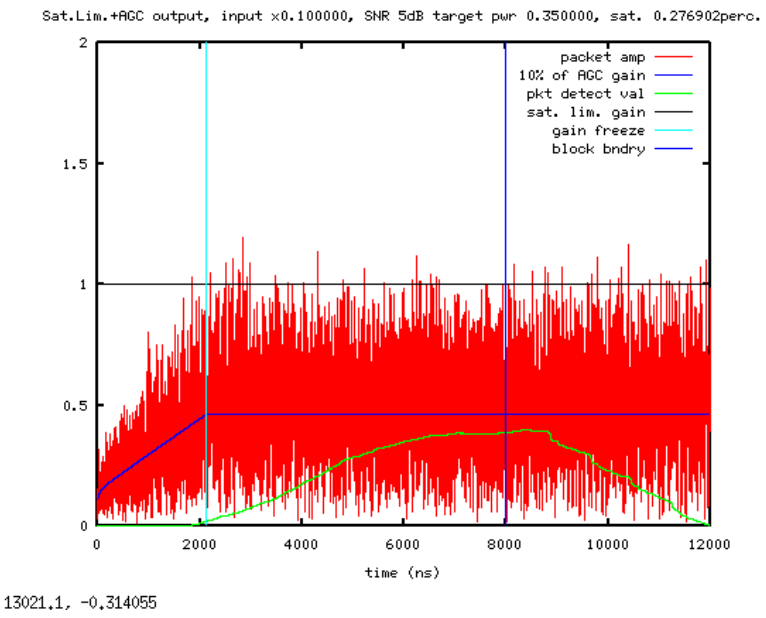


Figure 5.4: AGC performance, 5dB SNR

5.2 Packet Detector

The packet detector's job is to determine when a packet is present at the receiver. Three different packet detection methods were considered: sliding window, matched filter, and autocorrelation. The autocorrelation method was chosen. The matched filter method was not chosen due to the high computational complexity of performing convolution. A sliding window implementation was not pursued because, compared to the autocorrelation approach, it wastes all information about the structure of the incoming signal. The autocorrelation method is an elegant solution that utilizes the periodic nature of the STF fields to detect the presence of an 802.11 packet. Figure 5.5 shows the packet detection block diagram.

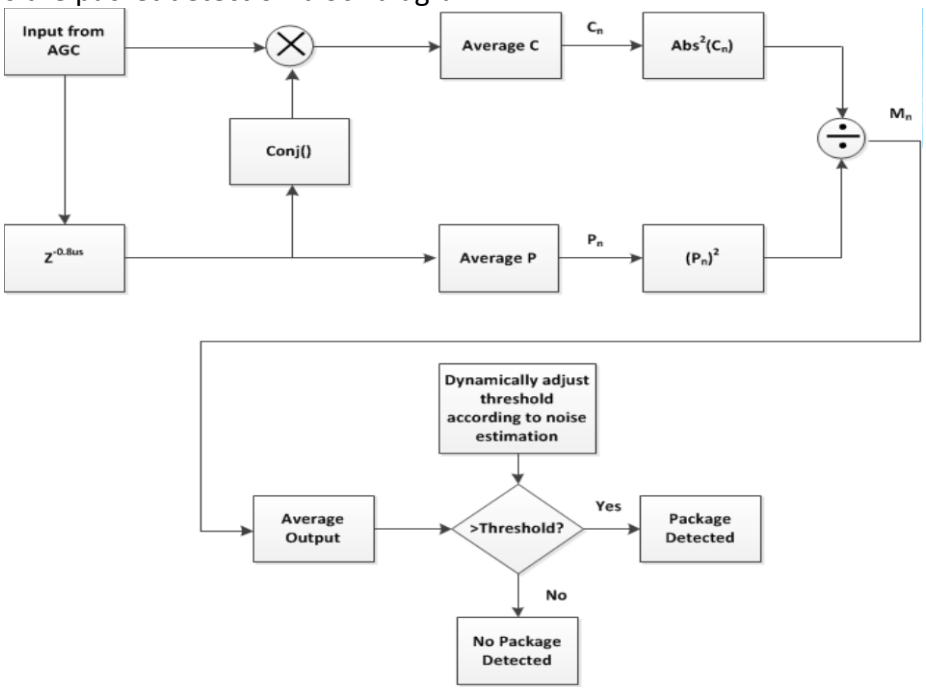


Figure 5.5 Packet Detection

The packet detector works by correlating the received signal with its value from 0.8us in the past. When a packet is present, the STF field should cause the correlation output to be high. The correlation output is normalized to input signal energy, with the highest possible output value being 1. The threshold for deciding a packet is present depends on the receiver SNR because lower SNR's correspond to lower packet detection outputs, as shown in the simulated curve in Fig. 5.6. It is not possible to simply set a low threshold to handle all cases because noise and non-packet signals with periods around 0.8us could then cause false positives.

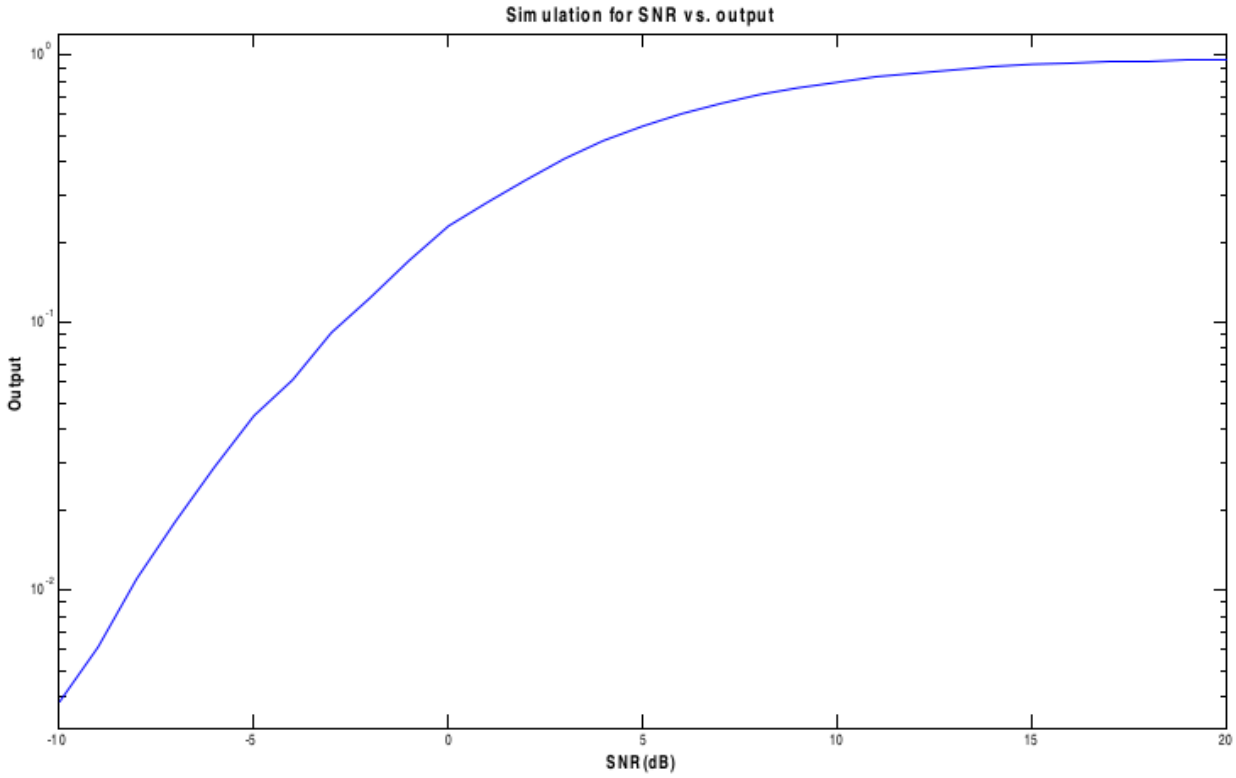


Figure 5.6: Packet detection output as a function of input SNR

The threshold value in the packet detector is a function of the estimated input noise, where increased noise corresponds to a lower threshold. The noise estimation block diagram is shown in Fig. 5.7. It makes use of the fact that the STF segments of the packet should repeat every 0.8us, so subtracting the received signal from itself 0.8us ago should only leave the noise. In practice the changing AGC gain and edge effects at the beginning and end of the STF segments make the noise estimate non-precise, but it works well enough to set thresholds that work for SNR's greater than 0dB.

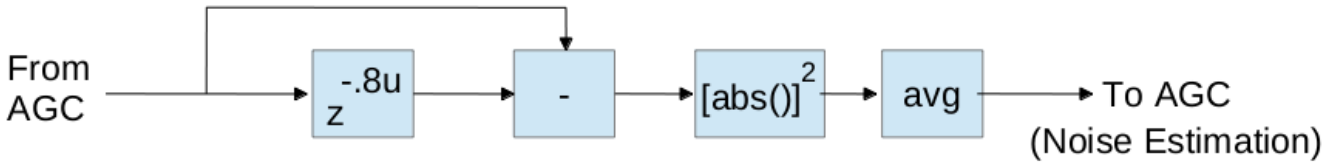


Figure 5.7: Noise estimation block diagram

The performance of the packet detector can be seen in the previously-introduced Figures 5.3 and 5.4. The green line in both plots represents the output value form the packet detector. Its maximum value is higher in the 50dB SNR case than in the 5dB SNR case. Accordingly, the threshold in the high SNR case is higher than in the low SNR case. The actual threshold values were chosen through trial and error such that the packet would be detected and AGC gain frozen just before 3us. The criteria for choosing values was that the timing specification should be met while still letting the threshold be as large as possible in order to prevent false positives.

5.3 Block Boundary Detector

After a packet is detected, the receiver still needs to decide where one packet segment ends and another starts. Our packet detector monitors the packet detection output and uses it to determine when the STF segment of the packet header transitions to the LTF segment. The entire packet can be broken into its constituent segments (i.e. STF's, LTF's, HT_LTF's, Data Payload) using this one transition as a reference.

The block boundary detector could be implemented as an independent module, but for simplicity this simulation bases the detector on the output of the existing packet detector. In particular, it was observed that the packet detector output remains fairly stable during the last of the STF segments and then abruptly drops 0.8us after the STF/LTF transition (see the green line indicating packet detector output on Figures 5.3 and 5.4). This drop is detected by monitoring the packet detection output for five criteria:

1. The mean of the last 70ns of packet detector output is greater than the gain-freezing threshold
2. The newest input value is greater than 0.005 away from the mean of the last 70ns of packet detector output
3. At least 4us have passed since gain-freezing
4. The previous input met criteria 1-3 but the newest input does not
5. The newest input is smaller than the mean of the last 70ns of packet detector output

Criteria 1 through 3 should all be met during the stable portion of packet detector output at the end of the STF segment. Criterion 4 detects that the stable output has ended, and criterion 5 ensures that the output is experiencing a sudden drop instead of a sudden rise (as might occur when a packet first arrives when previously there was only noise). Without all of these criteria working together, the boundary detection performed very poorly. With all the criteria present, and having tuned the algorithm parameters, the STF/LTF transition can be located to within 20ns of its actual position for SNR's of 5dB and above. The dark blue vertical lines on Figures 5.3 and 5.4 indicate the estimated block boundary. In both the 50dB and 5dB SNR cases the block boundary detector picked the point of transition from STF to LTF exactly 8us into the packet. Although the detection is not

always perfect, it is surprisingly robust and has worked very well during our simulations.

6. Carrier Frequency Estimation

In this section, a two-stage carrier frequency offset estimation algorithm is introduced. Coarse estimation is performed in time domain using the STF header. Fine estimation is performed in the frequency domain using the LTF.

6.1 Coarse Carrier Frequency Estimation

Fig. 6.1 shows the simulation diagram of coarse carrier frequency estimation. The estimation is in time domain based on the periodic characteristic of STF signal. The STF training signal repeats every 0.8us in the beginning of an 802.11 packet. In the presence of carrier offset, samples delayed by 0.8us would have a phase shift equal to

$$\theta = 2\pi \times 0.8\text{us} \times f_{offset} \qquad \text{(Eq. 6.1)}$$

For baseband signals, this phase offset can be computed by multiplying the input sequence by the conjugate of an appropriately delayed version of itself. The phase difference is calculated from this product, and the result is averaged to reduce the effects of noise. Finally, the $2\pi \times 0.8$us term is averaged over 4us of samples, filtering out the noise.

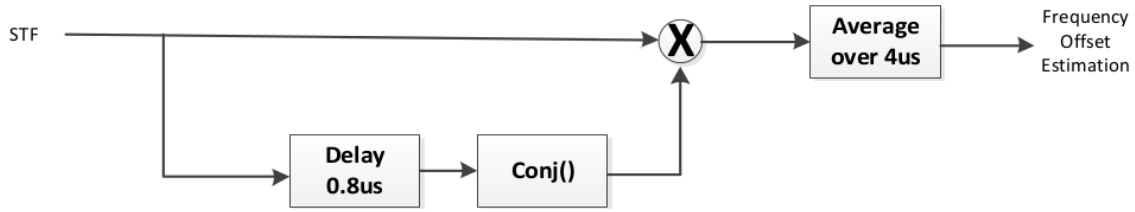


Figure 6.1 Simulation diagram of coarse carrier frequency estimation

Fig 6.2 shows the simulation results of coarse frequency estimation results using the proposed algorithm. For all three different SNR cases, the accuracy is greater than 99.97%. In the worst case scenario the error still goes to 24000Hz. Therefore, fine frequency estimation is still needed.

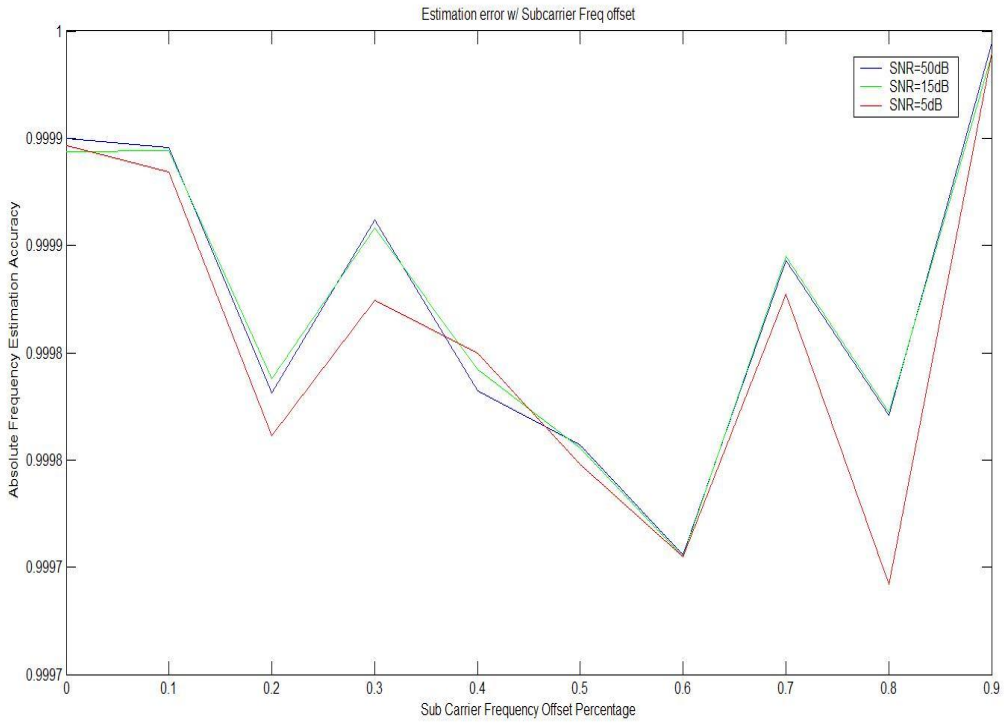


Figure 6.2: Coarse estimation accuracy with different offset

6.2 Fine Carrier Frequency Estimation

Fig. 6.3 shows the simulation diagram of fine frequency estimation. Fine frequency estimation is performed int the frequency domain using the periodic 2 LTFs. The signal is first passed through a low pass filter. High frequency noise is first filtered out with an ideal 10MHz lowpass filter. Then a buffer stores LTF_1 which is used to compare with its counterpart in LTF_2. The two repeated 50 samples of LTFs coming from LTF_1 and LTF_2 respectively are multiplied. Their estimation of carrier frequency offset is weighted based on their amplitude. This effectively eliminates the effect of noise since noise that is not on data carriers because such noise usually has relatively small amplitude. This partly explains why fine frequency is usually more accurate.

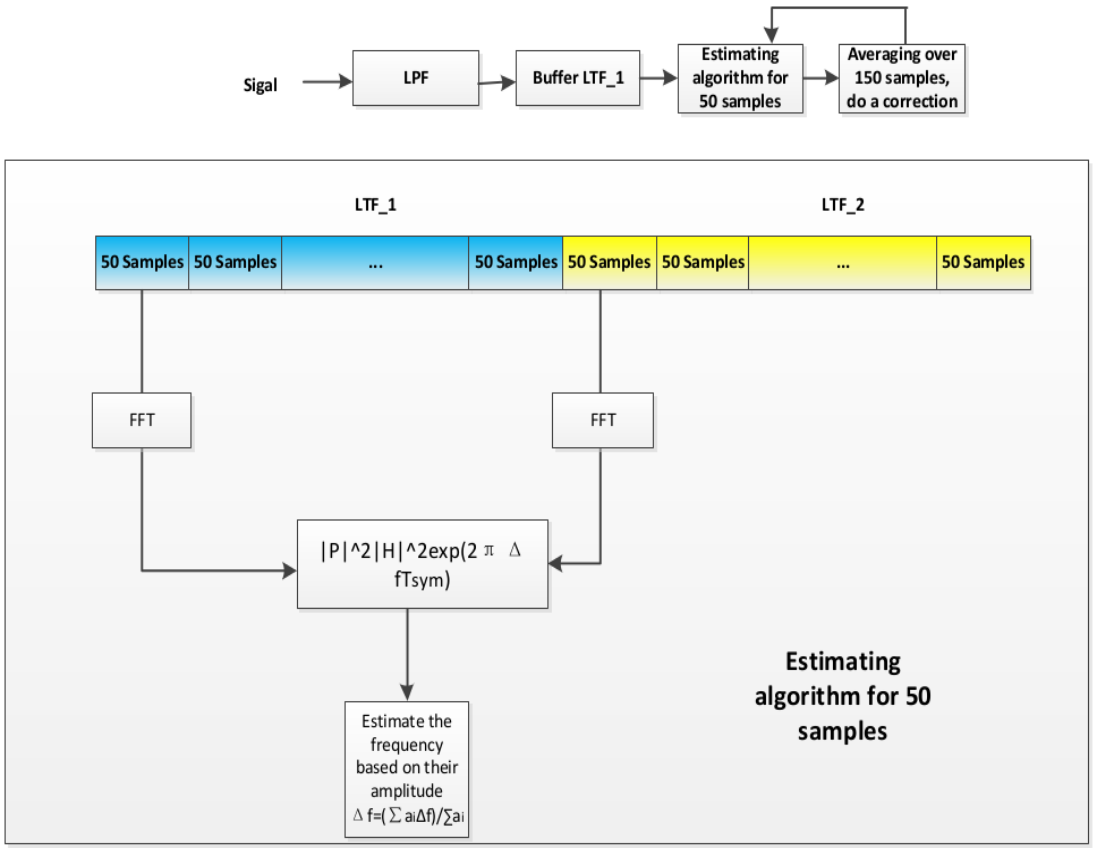


Figure 6.3: Simulation diagram of fine carrier frequency estimation

Simulation results shows that with the combination of coarse and fine carrier frequency offset estimation combined, the error is usually less than 50Hz.

7. Channel Estimation

Channel estimation is discussed first for the SISO case (Section 7.1) and then for the SIMO and MIMO cases (Section 6.2).

7.1 SISO Channel Estimation

The concept of channel estimation is to estimate the channel response given the input and output of the channel. The input signal in our algorithm, the L-LTF, contains signals in the 52 data sub-channels and is rich enough to excite the responses of the sub-channels of interests. The classical frequency domain method in the EE233 class notes, which is easily implemented in an OFDM system which makes heavy use of the FFT and IFFT anyway, is used in this project. Since the original sampling rate is 1GHz, the signals fed into the channel estimation block needs to be down-sampled to 40MHz because the bandwidth of the baseband is 20MHz. Taking 128 points for FFT, each output point represent the response of each sub-channel.

However, after we down-sample the L-LTF to 40MHz, the original 8000 sample points under 1GHz sampling rate shrinks to 320 points which is not enough for channel estimation to average out the noise. In order to utilize all the 8000 points of L-LTF for channel estimation, the channel input and

output sequences can be reconstructed by the method shown in Fig. 7.1, which gives acceptable results for channel estimation as shown in Fig. 7.2.

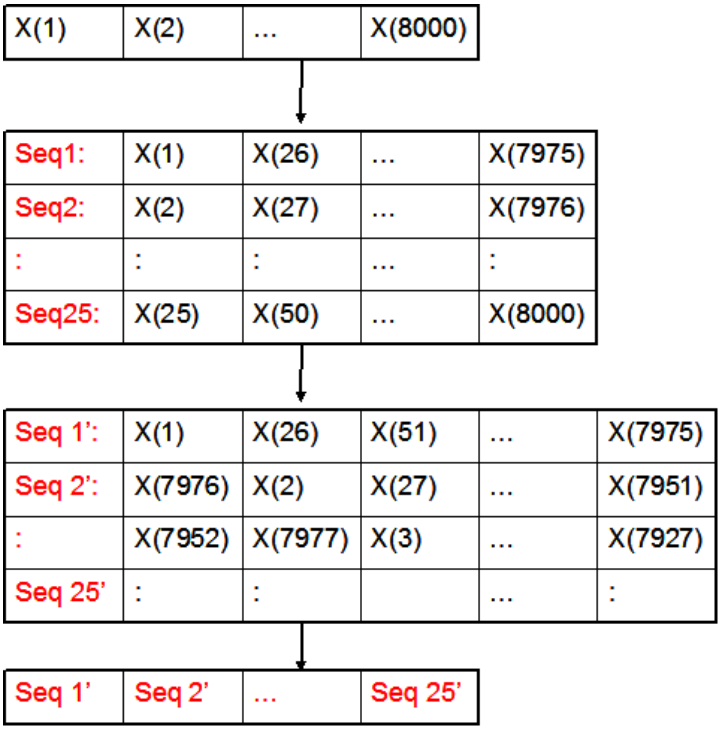


Figure 7.1: Reconstruction of input and output sequence

Figure 7.2 shows the result of channel estimation in the case of an AWGN channel with SNR=10dB. The estimated channel is mostly flat, as it should be.

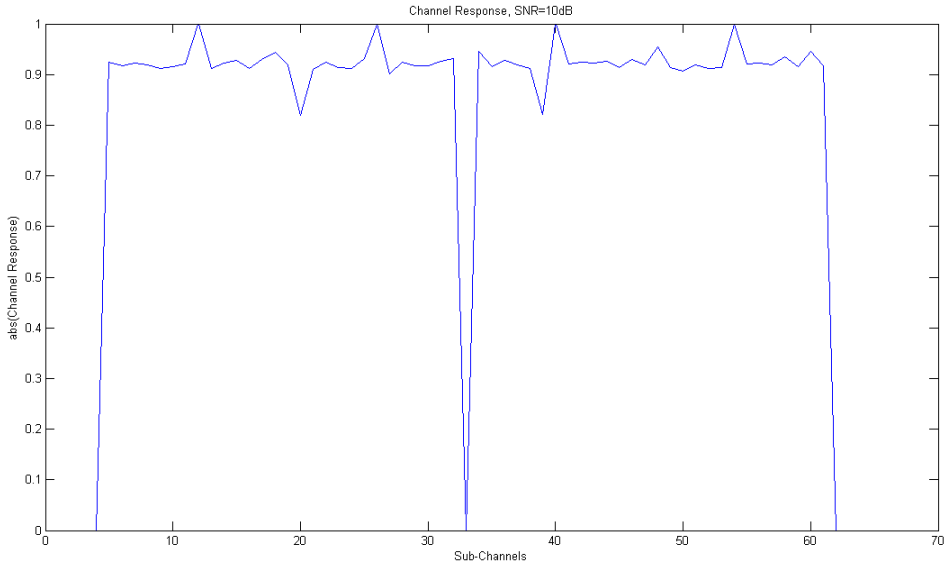


Figure 7.2: Channel estimation in AWGN, SNR=10dB

The classical frequency domain method with our reconstruction of channel input and output

provides credible and reliable channel estimation which will be seen in the Results section.

7.2 SIMO and MIMO Channel Estimation

When we extend the channel estimation for MIMO case, L-LTF can no more be used since each channel has the same L-LTF signal. Following the EWC HT PHY Specification, instead of L-LTF, HT-LTF is used for the 2x2 case which is illustrated in Fig. 7.3.

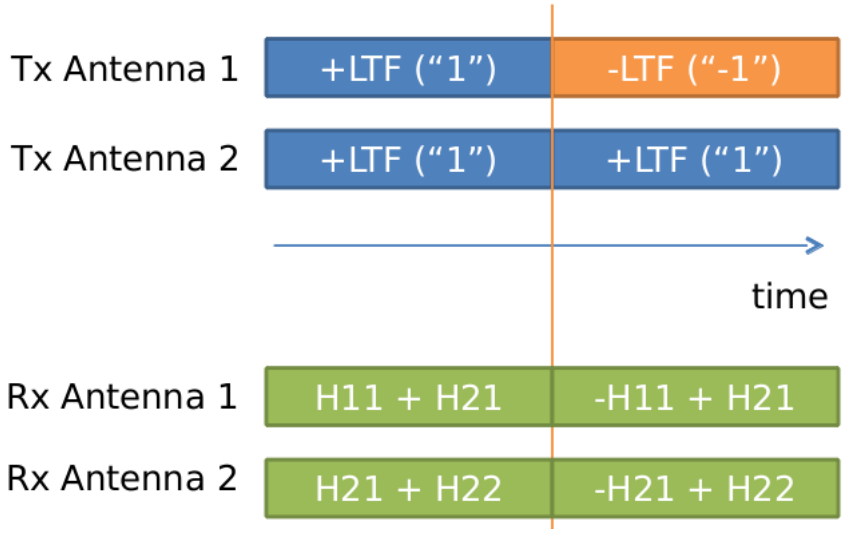


Figure 7.3: Illustration of 2x2 channel estimation

Each channel responses can be calculated by the following Eq. 7.1:

$$\begin{cases} H_{11} = 0.5*(est.11 - est.12) \\ H_{21} = 0.5*(est.11 + est.12) \\ H_{21} = 0.5*(est.21 - est.22) \\ H_{22} = 0.5*(est.21 - est.22) \end{cases} \quad \text{(Eq. 7.1)}$$

However, if we feed the HT-LTF signals in Fig. 7.3 to both the TX antennas, the latter part of the received signal will be close to zero which cannot be used for channel estimation. Therefore, before feeding the signal into the 2x2 channel estimation, the HT_LTF signal from either TX antenna needs to be scaled down to another level to avoid this kind of situation. The gains of the channel responses need to be compensated after the channel estimation. The channel estimation for the 2x2 case under AWGN channel with SNR=10dB is shown in Fig. 7.4, which provides good estimations of the channel.

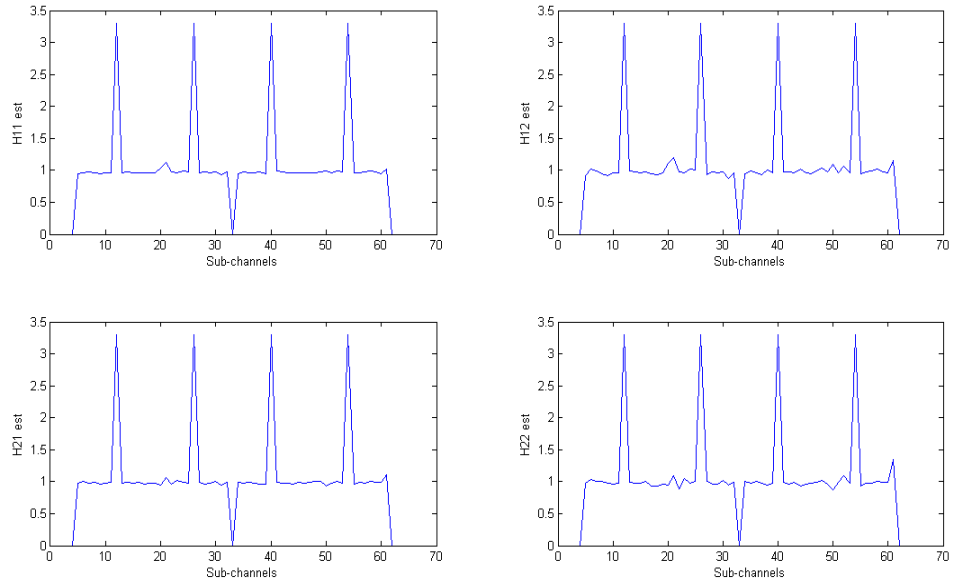


Figure 7.4: Channel estimation for 2x2 case AWGN channel, SNR=10dB

8. SIMO AND MIMO Implementation

The system simulation was also extended to investigate 1x2 SIMO (Single Input Multiple Output) and 2x2 MIMO (Multiple Input Multiple Output) cases. These cases required modifications to the system that will be discussed in section 8.1 for SIMO and 8.2 for MIMO.

8.1 SIMO 1x2

In the 1x2 SIMO system, a single antenna is used for transmission, but two antennas are used for reception. The transmitter is not modified in this situation, but the channel model is applied twice to the transmitted waveform. The first receiver is fed the first channel output and the second receiver is fed the second output. This model assumes that the two receiver antennas are close enough to each other that they experience identical path loss and shadowing, but far enough apart (greater than ~0.4λ) that they experience independent fading and noise.

At the receiver, the two antennas combine their received signals according to equal gain combining (EGC). EGC was chosen instead of MRC because of EGC does not require the difficult measurement of instantaneous SNR on each receiver. The drawback is a small BER degradation as compared to MRC, but as is seen in the Results section, the loss is quite small.

Prior to combining, each receiver's signal passes through its own AGC/packet detector/block boundary detector, frequency estimator, and channel estimator. Using individual channel estimation and compensation results in the same effect as co-phasing, so the channel-compensated output from each branch is averaged to produce the equal-gain-combined signal which is used as decoder input.

8.2 MIMO 2x2

In our implementation of 2x2 MIMO, we decided to send the same signal on both channels for

the sake of simple implementation. An actual 802.11 system would cyclically shift the data transmitted on each antenna in order to prevent beamforming, but our simulation does not deal with beamforming. Also, we did not achieve a working 2x2 system until very late in the project and so there wasn't available time to further tweak the 2x2 system and fix any resulting bugs in the code. The simplified 2x2 implementation. Each transmit signal's amplitude is scaled by 0.707 to ensure that the same amount of energy is transmitted as in the 1x1 or 1x2 case. Without this scaling it is not fair to compare results to the other cases since twice the energy is used – BER improvements would be partially due to increased Tx power instead of just diversity or array gains.

As in the 1x2 case, it is assumed that each of the channels experiences the same path loss and shadowing but independent fading and AWGN. The relationship between transmitted and received signals can be expressed by the following frequency-domain matrix equation:

$$\begin{bmatrix} Y_{1,fn} \\ Y_{2,fn} \end{bmatrix} = \begin{bmatrix} H_{n11} & H_{n12} \\ H_{n21} & H_{n22} \end{bmatrix} \begin{bmatrix} X_{1,fn} \\ X_{2,fn} \end{bmatrix} \quad \text{(Eq. 8.1)}$$

In Eq. 8.1, Y_{1fn} and Y_{2fn} represent the received frequency components on subcarrier *n* for receivers 1 and 2. H_{nij} is the flat fading value on subcarrier *n* for the channel between Tx antenna *i* and Rx antenna *j*. To recover the packet, the Eq. 8.1 is left-multiplied by H^{-1} or an estimate of H^{-1} to obtain the values transmitted on each antenna. This approach is the zero-forcing (ZF) MIMO equalizer. Compared to an MMSE equalizer, the ZF equalizer performs poorly when nulls are present in the channel. However, its implementation is simpler, and we learned in EE230B that for high SNR's the difference between ZF and MMSE is small. The channels of interest do not contain strong nulls, and so an MMSE equalizer was not deemed necessary.

The relative magnitudes of the channel matrix ignoring fading and noise are $H_{11} = H_{12} = H_{13} = 1$ and $H_{22} = 1.01$ for all subcarriers. This channel model corresponds to equal channels (neglecting shadowing and noise) between each Rx antenna and each Tx antenna. H_{22} is not set exactly to 1 because doing so would result in a singular matrix during perfect channel estimation. When perfect channel estimation is not used, noise in the estimate ensures that, in practice, the matrix is always non-singular.

The most difficult part of the MIMO simulation for us was determining how to estimate the channel matrix. We originally tried to devise an estimation method using only the STF's or LTF's because we had forgotten about the existence of the HT_LTF's. The HT_LTF's are crucial because they have an orthogonal polarity matrix ([1, -1; 1, 1]), which allows the received LTF signals to be broken down into the channel responses between each transmitter and receiver pair. Without this orthogonal nature, the channel would not be invertible and the H_{n11}, H_{n12}, etc. components of the channel matrix would not be identifiable.

9. Simulation Results

The following sections present simulation results for the cases of the 1x1 system in AWGN, 1x1 system in Rayleigh fading, 1x2 system and the 2x2 system in Rayleigh fading. Cases corresponding to multipath situations and are also briefly presented in the Appendix.

9.1 1x1 in AWGN

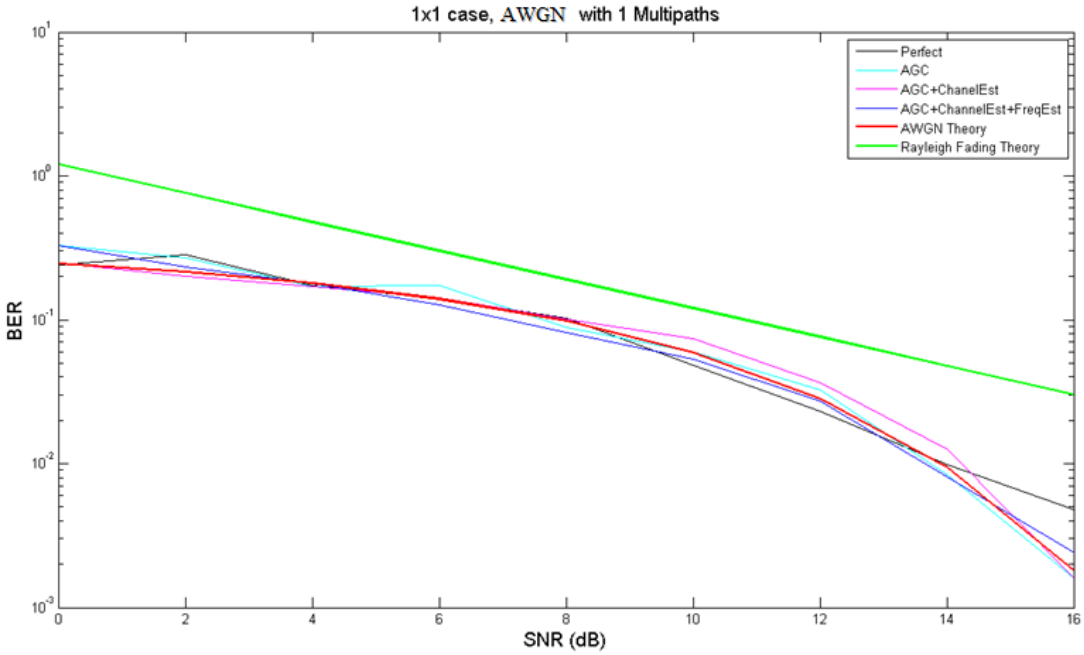


Figure 9.1: 1x1 system in AWGN

Fig. 9.1 shows theoretical and simulation curves for the 1x1 system in the presence of AWGN. The red line shows the theoretical value for AWGN, calculated by Eq. 9.1 and green line shows the theoretical value for Rayleigh calculated by Eq. 9.2 (modified form of [3]) for α_M =3 and β_M = 5.

$$BER = \frac{3}{4} Q\left(\sqrt{0.2 \times 10^{0.1 SNR(dB)}}\right) \quad \text{(Eq. 9.1)}$$

$$BER \approx \frac{1}{4} \frac{\alpha_M}{2\beta_M \bar{\gamma}_S} \quad \text{(Eq. 9.2)}$$

For points with SNR < 15dB, packets with 208 bits of payload were repeatedly generated and sent through the system until at least 50 bit errors occurred. For points above 15dB SNR, the simulation was only run until two errors occurred due to the long code runtime. The low error count at 16dB explains why deviation from the ideal curve is largest there. The simulation results is given for all perfect case, AGC imperfect case, AGC and channel estimation imperfect case and AGC, channel estimation and carrier frequency imperfect case. From the figure, readers can see Enabling successive blocks of the system did not cause any real performance degradation. Based on these results, the 1x1 case is successful. One point that deserves mentioning is that we only simulated SNR from 0 to 16dB because once SNR is beyond 16dB, it takes hours to get 1 point of BER in the plot. Furthermore, in our algorithm, the block boundary detection is turned on once the AGC is turned on. **So there is no curve for AGC + block boundary detection case.** The block boundary detection block works well in our simulation which always gives perfect block boundary detections.

9.2 1x1 with Multipath and Fading

Figures 9.2 and 9.3 show the simulation results with fading and 2-ray multipath and 4-ray multipaths, respectively. For the 2-ray model, two equal multipath components are present at 0ns and 50ns. For the 4-ray case, multipath with relative power of 0, -3, -6, -10 dB at 0, 70ns, 150ns and 200ns respectively are considered. Similarly, we plot with same cases in each scenario as mentioned in the previous paragraph. It is evident that 2-ray multipath fading is linear, as expected, and appear to be approaching the theoretical Rayleigh fading. The reason for the initial discrepancy at lower SNR is that the Rayleigh fading theoretical curve is a high-SNR approximation. With the growth of numbers of multipath, the BER become higher as expected. The error at high SNR mainly results from the addition of non-ideal channel estimation, but on the whole they are still in a reasonable range.

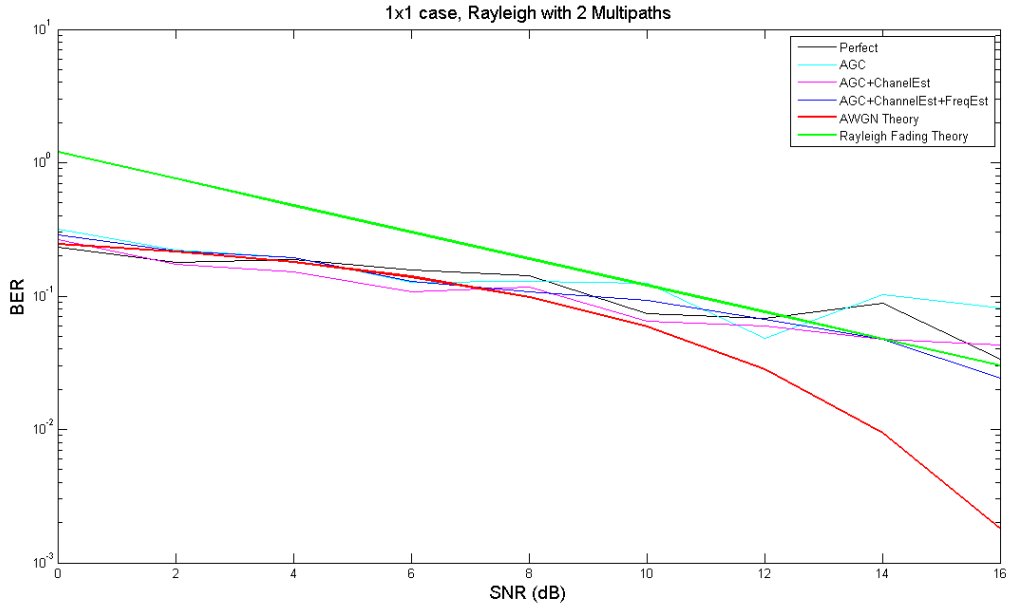


Figure 9.2: 1x1 system in 2-ray scenario

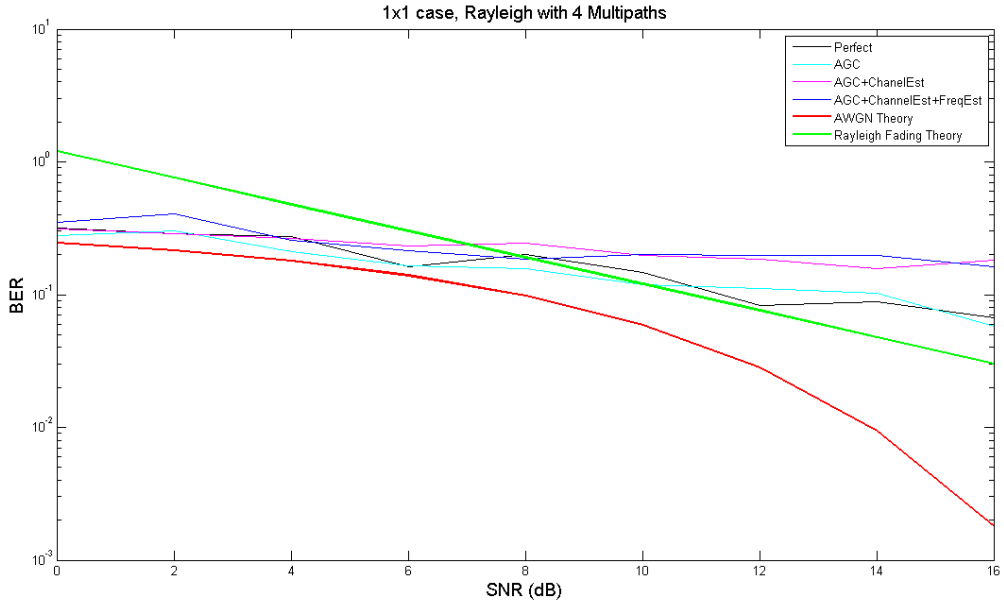
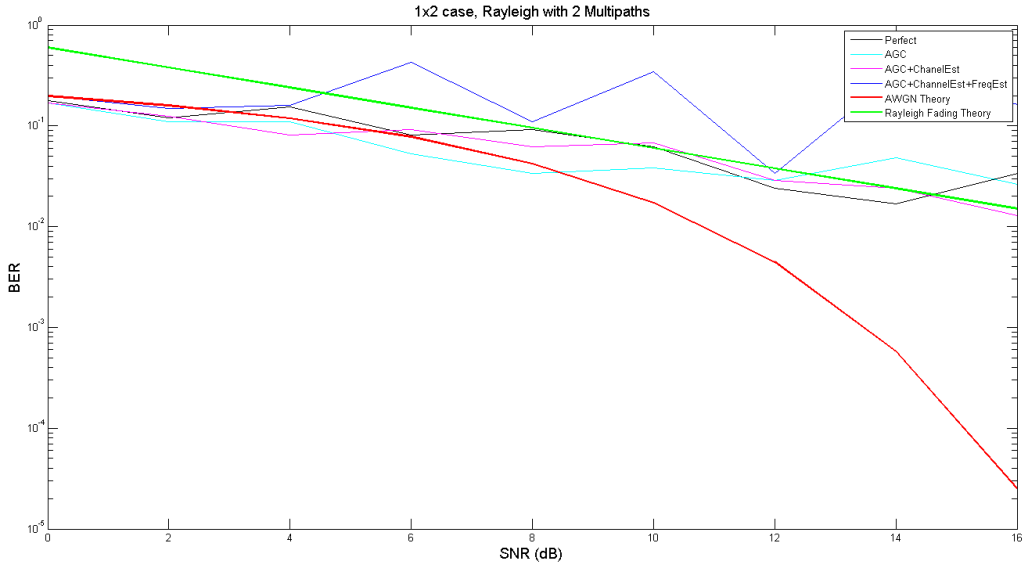


Figure 9.3: 1x1 system in 4-ray scenario

Figures 9.4 and 9.5 show the simulation results for a 1×2 system with 2-ray and 4-ray Rayleigh fading. The 1×2 system gets array gain so, on average, the performance is better than 1×1 system. Assuming everything is perfect scenario, readers can see a roughly 3dB gain in each of the corresponding plots compared with their 1×1 counterpart, which agrees with the theory (theoretical curves are shifted 3dB from their previous values) for MRC and equal gain combining. The curve with everything turned in is so jagged due to only simulating 6 error points at each SNR due to time constraints, and also just because each enabled component adds its own imperfections to the



simulation. As whole, though, the curve generally follows the trend of the theoretical line.

Figure 9.4: 1x2 system in 2-ray scenario

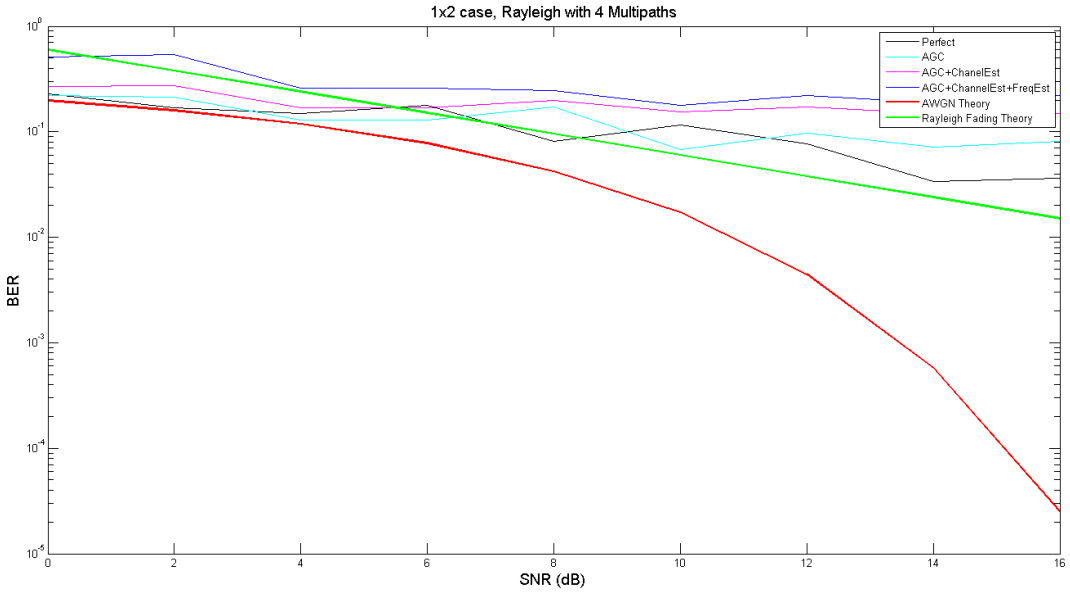


Figure 9.5: 1x2 system in 4-ray scenario

We further extend our simulation to 2x2 case in AWGN as shown in Figure 9.6 and Figure 9.7. In AWGN case, we only extend the SNR to 12dB, because beyond that, the simulation is far too slow. In both 1x2 and 2x2 cases in AWGN channel, the simulation results are still close to the theoretical curves. However, in Rayleigh fading channel, the BER doesn't have apparent improvement when SNR goes higher by either 1x2 case or 2x2 case.

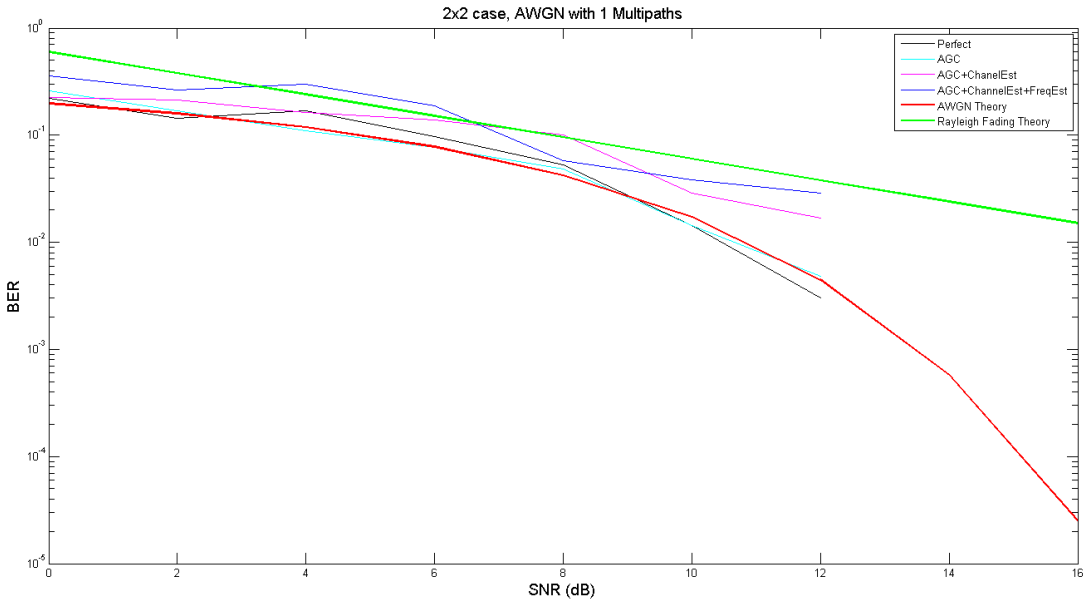


Figure 9.6: 2x2 system in AWGN superimposed on MRC theoretical curve

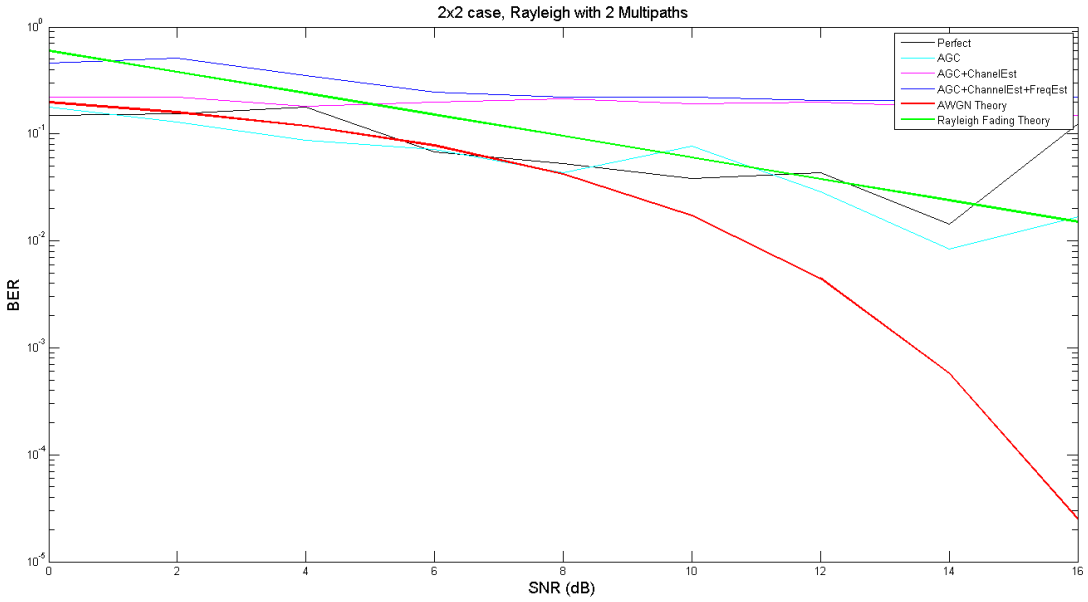


Figure 9.7: 2x2 system in 2-Ray scenario superimposed on MRC theoretical curve

10. Discussion

As a whole, the simulated 802.11n system worked well. The BER simulation in AWGN for the 1x1 case closely matches the theoretical curve, and the simulations in fading exhibit a linear dependence on SNR as expected.

Since this was a multi-week project, there were many difficulties that had a chance to arise. The greatest difficulties came at the end when we began interfacing the various components that we had created over the course of the project. We knew from the beginning that connecting everything would be difficult and so we started each week's project by defining input and output of our various functions, but some details inevitably slipped through the cracks, which lead to searches for the cause of various little errors like a misplaced factor of two (connecting out AGC to the channel estimator) or a dropped negative (2x2 channel estimation). Our planning did allow us to avoid disasters such as discovering that two functions were so incompatible that they would have to be rewritten, but it seems like no amount of planning can prevent every possible little mistake.

Another difficulty for this final project was due to our previous failures to implement a 2x2 system, either ideal or actual. Because we forgot about the existence of the HT_LTF fields until the final week, we struggled to understand how the implement MIMO channel estimation given on the STF's, LTF's, and data payload. Once we remembered the HT_LTF's the problem wasn't as difficult to approach, but incorporating new, largely untested functionality into the newly-combined system caused the entire system to fail whenever we would mix up some variable names or experience an indexing error. Finishing the MIMO simulations when they were originally due would have left us more time to combine all the systems and would have introduced fewer unknowns into the system.

Using a lower sampling frequency would have made the simulation more realistic and

improved runtime. The original, high sampling rate was chosen during the second week of the project to provide high time resolution and channel models, but all subsequent code built on that project, and so the sampling frequency was essentially locked from that point onwards. The most visible effect is the painfully slow AGC simulation due to its having to iterate over a large number of samples.

Although this final piece of the project was frustrating, it was exciting to enable each section of the system one by one and, after some troubleshooting, see the entire system behave in a way that theory said it should. First seeing Figure 9.1 in particular was great as it was confirmation that our hard work from the past two months had produced something that worked. By the end of the project we knew the overall system well enough that when we saw some anomalous output, we could quickly pinpoint its cause from three or four stages back in the simulation. That kind of familiarity with a system is hard-won but gives us confidence to approach other communication systems in the future.

Acknowledgement

We would like to thank Prof. Babak Daneshrad for discussion of project ideas.

References

[1] Figure from EWC HT PHY Standard v.1.01
[2] http://www.dsplog.com/2008/06/05/16qam-bit-error-gray-mapping/.
[3] Andrea Goldsmith, *Wireless Communications*, Eq. 6.61 on pg. 184

Appendix

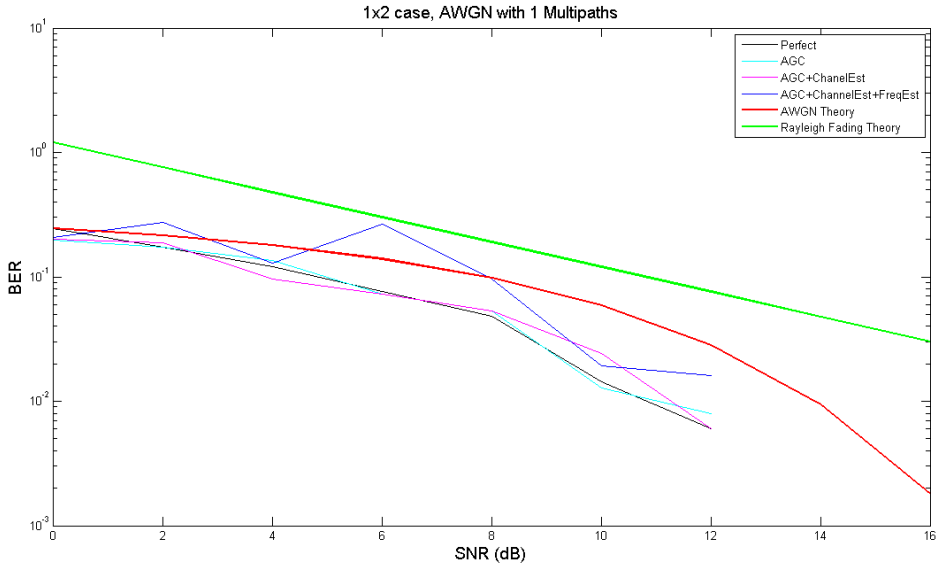


Figure A.1: 1x2 system in AWGN

Fig. A.1 shows the simulation output for the 1x2 case in only AWGN and for a truncated range of SNRs. It is clear that the curves tend to follow a copy of the AWGN theoretical curve shifted 3dB to the left, as would be expected from this equal gain combining system (which closely

mirror MRC in BER performance). Fig. A.2 shows the same simulation results superimposed on this MRC 3dB-shifted theoretical curve. The simulation is only to 12dB, since beyond that it is too slow to simulate

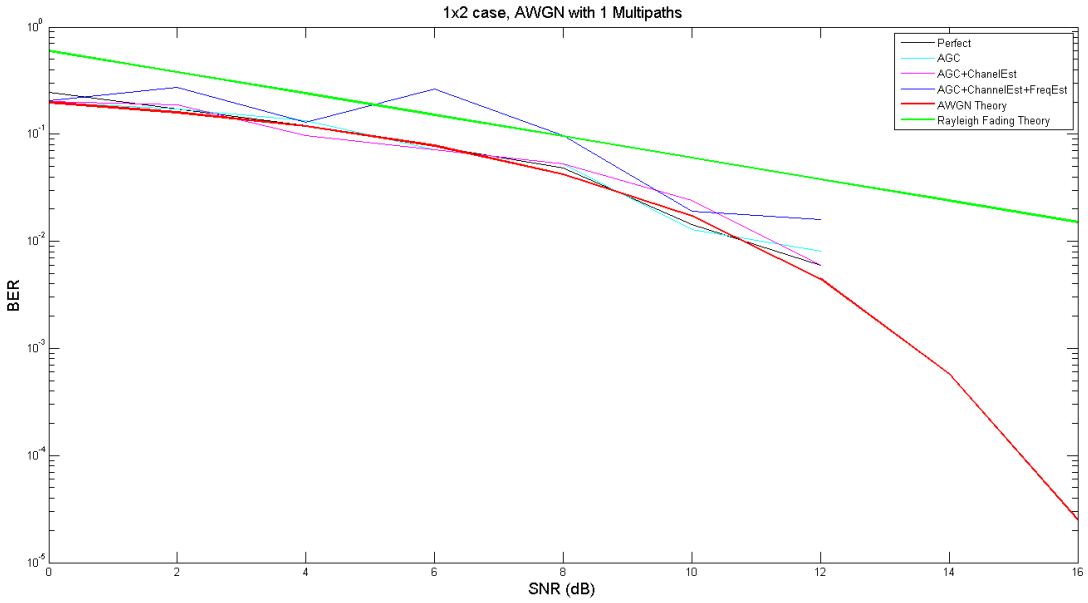


Figure A.2: 1x2 system in AWGN superimposed on MRC theoretical curve

Figure A.3 and A.4 are 1x2 cases in 2-Ray and 4-Ray model under Rayleigh fading. Compared to 1x1 case, besides the diversity gain, 1x2 case have no extra effect.

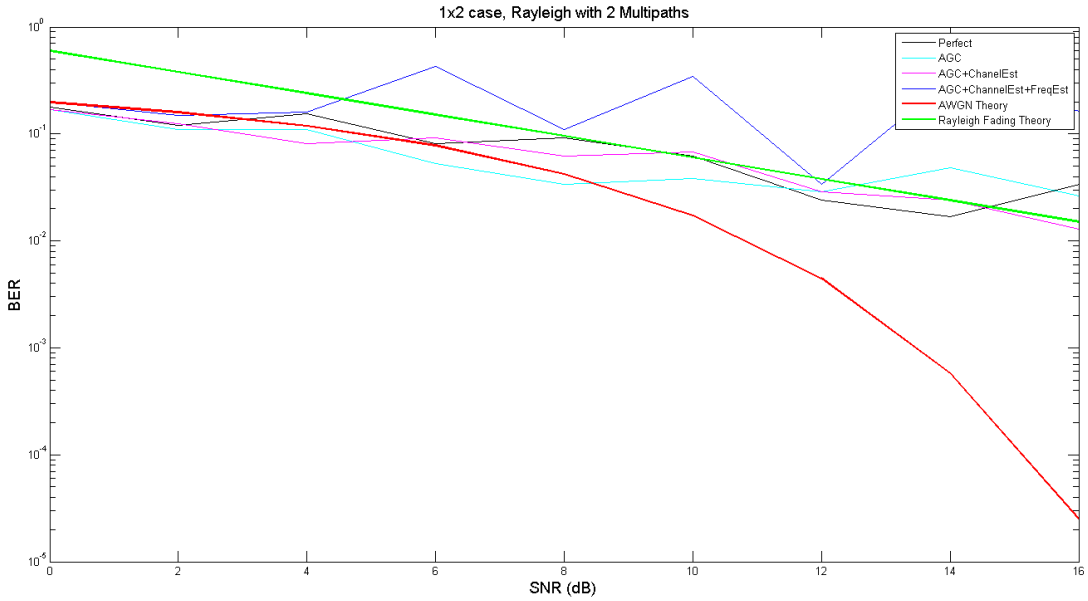


Figure A.3: 1x2 system in 2-Ray scenario superimposed on MRC theoretical curve

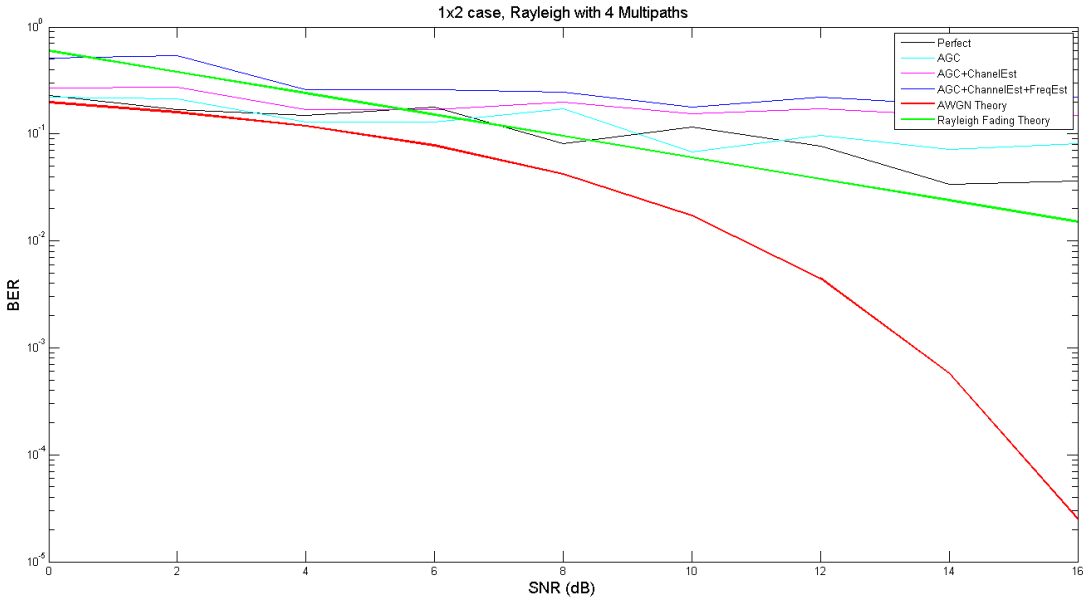


Figure A.4: 1x2 system in 4-Ray superimposed on MRC theoretical curve