Error Correction Coding: Weeks 1-2

AWGN Channel & Coding Theory Review

Eric DeFelice
Department of Electrical Engineering
Rochester Institute of Technology
ebd9423@rit.edu

***Abstract*—Error correction coding is used extensively in mobile communication systems to help combat the effects of mobile radio channels. These coding schemes need to be able to detect and correct bit errors caused by the channel while maintaining efficiency. The most appealing coding schemes are that which give the best bit error rate while adding the fewest number of redundant bits. This paper will review one wireless communication channel (the additive white Gaussian noise channel), give an introduction to the field of channel coding and illustrate the benefits of channel coding by presenting a simple channel coding scheme.**

I. INTRODUCTION

The field of error correction coding is very important in communications systems, because the data is almost always sent through a noisy channel from the transmitter to the receiver. The noise from this channel introduces bit errors at the receiver, which is where the channel coding scheme comes in. The purpose of the error correction code is to minimize the bit errors seen by the receiver, while using the fewest number of redundant bits. Having some information about the communications channel helps to design a more efficient code. For example, if the channel was more likely to flip a '1' to a '0', causing an error, but a '0' was never in error at the receiver, then the coding scheme should not add redundant code bits to the '0's in the original data stream.

The performance of the various coding schemes that will be introduced over the course of my studies will be analyzed over an additive white Gaussian noise (AWGN) channel. It is for this reason that the AWGN channel will be reviewed in this paper. The model of this channel will be introduced and the capacity of this type of channel will be looked at.

Once the channel model is fully understood, an overview of the channel encoder and decoder system will be reviewed. This system includes the modulation scheme, the channel encoder, the channel model and the decoder. Several modulation schemes will be looked at and their performance and trade-offs will be discussed.

Finally, once the channel capacity theory is understood, the difference between an un-coded modulation scheme and the channel capacity limit can be compared. To assist in this task, a simple coding scheme will be implemented and its benefits will be discussed.

II. AWGN CHANNEL

In this section, the additive white Gaussian noise channel will be discussed, and a model will be presented that accurately captures its performance. Important parameters of this channel include the signal-to-noise ratio (SNR), spectral efficiency, and the capacity, each of which will be analyzed later in this section. Before diving right into the definition of the channel model, let us take a step back and look at how the radio channel fits into the mobile communications system as a whole. Figure 1 shows the basic structure of a digital communications system. The channel in this system could be wired or wireless, and may be a multipath channel or a line-of-sight wireless channel. There are several different factors which affect a wireless mobile channel, and these factors may be modeled by Gaussian random variables, a Rayleigh distribution or a Rician distribution. For the purposes of my studies, I will focus on the Gaussian channel, as it provides a simple model that is understood and has been used in the analysis of channel coding schemes previously.

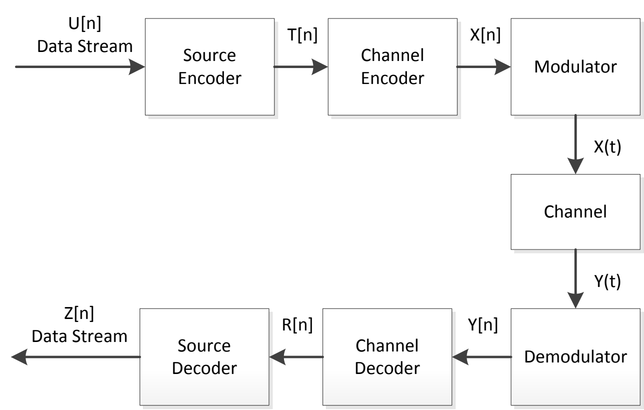
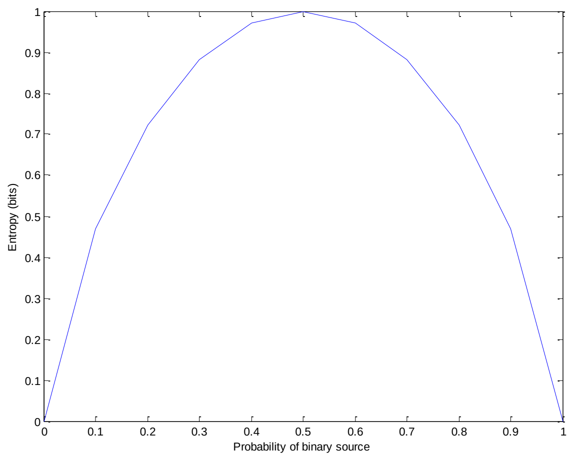


Figure 1: Structure of digital communication system

The transmitter of the system starts with a source encoder. This encoder serves an entirely different purpose than the channel encoder. The source encoder is designed to reduce the number of bits of the source, bringing it as close to the entropy as possible. This encoder may be lossy or lossless. The source encoder is almost the opposite of the channel encoder, in the sense that the source encoder is designed to remove redundant bits from the data source and the channel encoder is designed to add redundant bits. Next, the encoded data, X[n], is modulated and then sent through the communications channel. On the receive side, the reciprocal of these tasks are done to get the received data stream.

For the remainder of this course, it will be assumed that the source data stream into the channel encoder has entropy of 1. If we remember from information theory, the entropy is the amount of information carried by a data stream, telling us how many bits we need to send that information over a channel. Figure 2 shows the definition of the entropy of a data source and the graph of the entropy vs probability of the binary source.



$$I(\mathcal{X}) = -\sum_{i=1}^{M} P_{\mathcal{X}}(x_i) \cdot \log_2(P_{\mathcal{X}}(x_i))$$

Figure 2: Entropy of a binary source

Using this assumption, the input into the channel encoder has an equal probability of being a ‘0’ or a ‘1’. After this data stream goes through the channel encoder and the modulator, it finally goes through the communications channel. The additive white Gaussian noise channel can be described by the following model:

$$\mathbf{Y(t) = X(t) + N(t)} \tag{1}$$

Where Y(t) is the receive data stream, X(t) is the input waveform, described as an equal probability binary data source, and N(t) is a real white Gaussian noise process, independent from X(t). In a real system, the signal power of X(t), we will call S, would depend on the power amplifier in the system and on the modulation scheme. If the modulation scheme is said to be “constant envelope” than the peak-to-average power ratio is almost 1 and the power is not very dependent on the input data stream. Conversely, if the scheme is not constant envelope, than the signal power does depend on the transmit symbol values. This leads us to the definitions of the average signal power and the average noise power, which are shown below:

$$\mathbf{S} = \boldsymbol{E}\{\, \mathbf{X}^2 \}\tag{2}$$

$$\mathbf{N} = \boldsymbol{E}\{\, \mathbf{N}^2 \}\tag{3}$$

Where $E\{X^2\}$ is the expected value of the square of the input data stream, and $E\{N^2\}$ is the expected value of the square of the Gaussian noise values. These two values are important because they are the key variables when describing the channel capacity for an AWGN channel. These variables describe the signal to noise ratio, S/N, which is the main factor in the capacity of the AWGN channel, as seen in the channel capacity equation shown below:

$$C_{[\mathrm{b/s}]} = W \log_2(1 + \mathrm{SNR}) \quad \mathrm{b/s}.$$

The previous equation was theorized by Claude Shannon in the 1940’s to describe the maximum possible efficiency of a coding scheme over a communication channel. This shows that the capacity of a channel is based on the SNR (signal-to-noise ratio) of that channel and the bandwidth in which the data stream is using. This equation is very important in the field of error correction coding, as it provides an upper bound for the capacity of a given channel. This means that theoretically, a coding scheme could be devised such that its data rate, R, is less than or equal to the channel capacity, C, but not more than it. The relationship between the data rate and the channel capacity can then be used to gauge the efficiency of a particular coding scheme. This relationship will be used over the span of my studies to evaluate the performance of various coding schemes.

To understand the concept of channel capacity a little more clearly, a couple experiments were performed to help visualize the capacity vs different SNR values and the capacity vs different channel bandwidths. The results of those experiments are shown in the figures below.

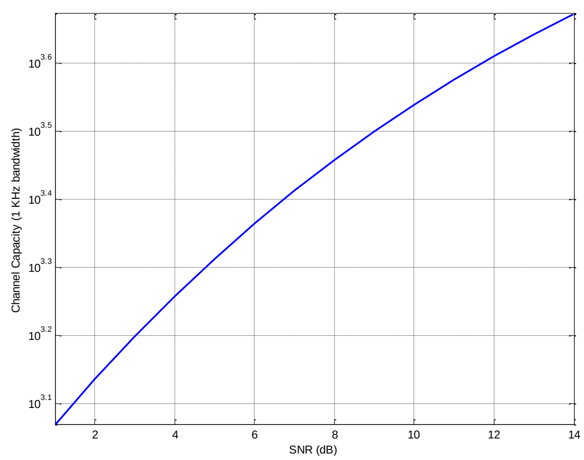


Figure 3: Channel capacity vs SNR

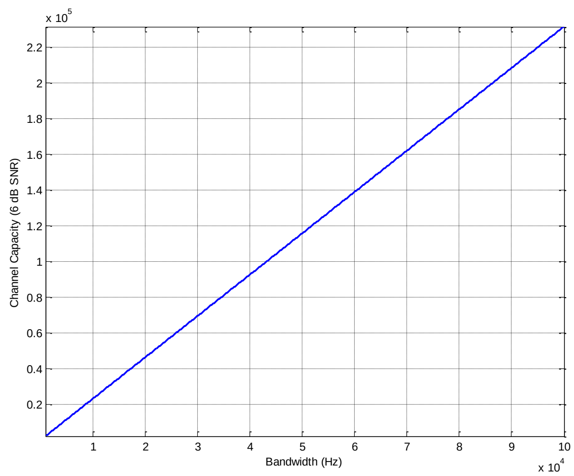


Figure 4: Channel capacity vs channel bandwidth

Now that a model of the AWGN channel has been presented and its capacity has been analyzed, it is time to review some of the theory behind channel coding.

III. Error Detection, Correction, and Decoding

The purpose of forward error correction coding is to increase the distance between codewords, so that the receiver can decode to the correct symbol with the presence of noise. Let us look at a system where the codewords are single bits. If one of the bits is switched from '0' to '1' because of the noise, then there is no real way of knowing what the transmitted bit was. Now if we have a system where there are still two codewords, but now they are '00000' and '11111'. If one of these bits gets flipped, the receiver should still get the correct symbol because of the redundancy. This is possible because the second set of symbols has a larger hamming distance.

If the received codeword contains errors, there must be a way to find the *closest* codeword to what was received. The way a receiver performs this task is by the use of *decoding rules*. One such rule is the *Maximum Likelihood Decoding (MLD) Rule*. This decoder will choose the value of c_x that is most likely to have been transmitted, in that it maximizes the forward channel probabilities. This is shown in the following equation:

$$\mathcal{P}(\mathbf{x}\text{ received} \mid \mathbf{c}_{\mathbf{x}}\text{ sent}) = \max_{\mathbf{c}\in C} \mathcal{P}(\mathbf{x}\text{ received} \mid \mathbf{c}\text{ sent}).$$

The main way to improve the detectability of coded symbols is to increase the Hamming distance. The hamming distance is the difference between two different symbols. The number of differences equates to the distance between those symbols. To elaborate further, we can look at an example. We can define three symbols as follows: x=10101, y=11111, z=00000. The hamming distance, or d(sym_1, sym_2), between x and y is d(x,y)=2. This is because x and y only have two bits that are different, the second bit and the fourth bit. Computing this distances for the other symbols gives us: d(x,z)=3, d(y,z)=5. Now suppose we receive a symbol r at the receiver, r=00101. Finding the distance between this received symbol and the other symbols in the code set gives us: d(r,x)=1, d(r,y)=3, d(r,z)=2. Using *minimum distance decoding*, the receiver should choose that the transmitted symbol was x, as it is the *closest* to the received symbol.

This example briefly shows how a symbol is detected at a receiver, even if some of its bits are in error. The noisier the channel, the more bits there will be that are incorrect, leading to some errors in symbol detection. A good forward error correcting code will minimize these errors while using the fewest number of redundant bits. In the next section, we will briefly look at a simple channel code.

IV. Sample Channel Code

As a case study into the channel coding field, we can look at a sample channel code to see the benefits of such a code. For this purpose, we will use a repetition code. This will be done by repeating the current transmit bit four times. The two code sets we will use are {0, 1} and {00, 11}. Doing this does have some side effects. For example, to use the same bandwidth we would need to increase the transmit power when using the second set in order to achieve the same SNR. Another option would be to increase the bandwidth to account for the overhead that is added by the redundant bits.

Forgetting about these tradeoffs for now, we will transmit 10^6 of each of the symbols over an AWGN channel and use minimum distance decoding at the receiver. A graph showing the bit error rates (BER) of the two coding schemes is seen in figure 5.

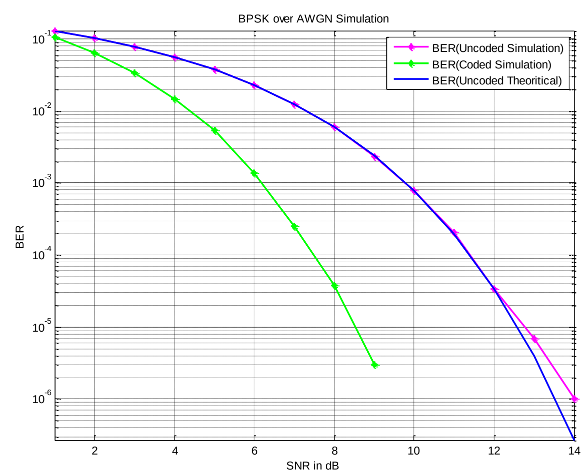


Figure 5: 1-bit codewords vs 4-bit codewords

It is seen in the figure above that the BER is reduced when using the 4-bit codewords. This simple example shows the benefits of using a forward error correction coding scheme over an AWGN channel.

V. Conclusion

In this paper, the AWGN channel was introduced and its characteristics were discussed. This channel was proposed to be used as the mobile communications channel for the duration of my studies because of its relative simplicity and its popularity in this field. The concepts of error detection, error correction and decoding were also discussed. Maximum likelihood decoding was reviewed and the Hamming distance between sample codewords was computed. This aided in showing the benefit of adding redundant bits, as doing so increases the Hamming distance between codewords. Finally, a simple example was presented, and an experiment was conducted to show how a channel code can reduce the BER of a data stream.

The next couple of weeks will be spend studying finite field theory, as it is the basis for many of the practical forward error correction coding schemes.

References

[1] San Ling, Chaoping Xing, *Coding Theory: A First Course*. Cambridge University Press, 2004

[2] Andre Neubauer, Jurgen Freudenberger, Volker Kuhn, *Coding Theory: Algorithms, Architectures, and Applications*. John Wiley & Sons Ltd., 2007

Appendix

```
%% Error correction coding : Weeks 1-2
close all;
clear all;
clc;

% create chart of entropy for a binary source
p=0:0.1:1;
ent=(-p.*log2(p))+(-(1-p).*log2(1-p));
% set y(1) & y(11) to 0 since they are undefined
%{log2(0)=undefined in matlab)
ent(1)=0; ent(11)=0;
% plot entropy vs probability of source
plot(p,ent);
xlabel('Probability of binary source');
ylabel('Entropy (bits)');

% Variables
SNRdB=1:1:14;                          %Signal to Noise Ratio in dB
SNR=10.^(SNRdB/10);                    %Signal to Noise Ratio in Linear Scale
Bit_Length=10^6;                       %No. of Bits Transmitted
BER_uncoded=zeros(1,length(SNRdB));    %Simulated Bit Error Rate for uncoded system
BER_coded=zeros(1,length(SNRdB));      %Simulated Bit Error Rate for coded system
C_snr = zeros(length(SNR),1);          %Channel Capacity over SNR
BW = 1000:100:100000;                  %Bandwidth for Simulation
C_bw  = zeros(length(BW),1);           %Channel Capacity over Bandwidth

% BPSK Transmission over AWGN channel - vary bandwidth
for k=1:length(BW);
    C_bw(k) = BW(k)*log2(1+SNR(6));
end
% BPSK Transmission over AWGN channel - vary SNR
for k=1:length(SNR);
    C_snr(k) = 1000*log2(1+SNR(k));
    x=(2*floor(2*rand(1,Bit_Length)))-1;
    x_coded=repmat(x,1,4);
    y=(sqrt(SNR(k))*x)+randn(1,Bit_Length);
    y_coded=(sqrt(SNR(k))*x_coded)+randn(1,(Bit_Length*4));

    BER_uncoded(k)=length(find((y.*x)<0));

%BER_coded(k)=length(find(((y_coded(1:Bit_Length)+y_coded(Bit_Length+1:2*Bit_Length)).*
x)<0));

BER_coded(k)=length(find(((y_coded(1:Bit_Length)+y_coded(Bit_Length+1:2*Bit_Length)+...
```

```
y_coded(2*Bit_Length+1:3*Bit_Length)+y_coded(3*Bit_Length+1:4*Bit_Length)).*x)<2));
end
semilogy(SNRdB,C_snr,'linewidth' ,2.0);
xlabel('SNR (dB)');
ylabel('Channel Capacity (1 KHz bandwidth)');
axis tight
grid
figure;
plot(BW,C_bw,'linewidth' ,2.0);
xlabel('Bandwidth (Hz)');
ylabel('Channel Capacity (6 dB SNR)');
axis tight
grid
figure;
BER_uncoded=BER_uncoded/Bit_Length;
BER_coded=BER_coded/Bit_Length;
semilogy(SNRdB,BER_uncoded,'m-*', 'linewidth' ,2.0);
hold on
semilogy(SNRdB,BER_coded,'g-*','linewidth',2.0);
semilogy(SNRdB,qfunc(sqrt(SNR)),'b-','linewidth',2.0);      %Theoritical Bit Error Rate
title('BPSK over AWGN Simulation');xlabel('SNR in dB');ylabel('BER');
legend('BER(Uncoded Simulation)','BER(Coded Simulation)','BER(Uncoded Theoritical)')
axis tight
grid
```