ELEC5507 Assignment

Vanush Vaswani - 308196465
Zhaoyan Liu - 440092733
Stephen Tridgell - 309205867

May 28, 2014

Contents

List of Figures

List of Tables

Project Introduction

The goal of error control coding is to encode messages for transmission with redundancy such that the receiver can correct the errors in the transmission and recover the original data. In this project, encoders and decoders will be simulated and analysed by using MATLAB. The first section involves the design and simulation of the BCH code over BSC and AWGN channels. The second section uses LDPC codes to meet the requirements of a power limited system with very high reliability. Similar simulations are run and compared with the results of the BCH code.

Section I - BCH code

Analysis

Background

Basics of BCH code

Implementation of BCH Decoder using Berlekamp's algorithm

Design

Blah blah blah

Implementation

See below. ..hahaa

Calculations and Results

Question 1

What is the generator polynomial for this code? Use MATLABs bchgenpoly function to verify your answer

The generator polynomial for a n = 31, k = 16, and t = 3 BCH code is $g(x) = x^{15} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^5 + x^3 + x^2 + x^1 + 1$. This was verified with

```
[gen, t] = bchgenpoly(31,16)
```

Question 2

What is the minimum distance of this code?

For a BCH code, $d_{min} \geq 2 * t + 1 = 7$, this was verified using

```
gfweight(double(gen.x), 31)
```

which gave a minimum distance calculation of 7.

Question 3

Construct the reduced syndrome lookup table for this code. You need to write a program to do this since it is difficult to do it by hand. You do not need to include the whole array in your report due to its large size. Instead, just show the sub-array consisting of the first 5 rows in your report, and include a separate text file (.txt) enumerating all the data in your electronic submission.*

Generate the syndrome lookup table by:

```
[h g k] = cyclgen(31, double(gen.x))
trt = syndtable(h)
```

The first 5 rows are:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The corresponding syndromes for these error patterns are:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	=	**0**
0	0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	=	**1**
0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	=	**2**
0	0	0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	=	**3**
0	0	0	0	0	0	0	0	0	0	0	0	**1**	0	0	=	**4**

The rest of the array is in "trt.csv".

Question 4

Based on the standard array you obtained in task 3, find out the weight distribution of the coset leaders.

```
weights = sum(trt')
A0 = sum(weights == 0)
A1 = sum(weights == 1)
A2 = sum(weights == 2)
A3 = sum(weights == 3)
A4 = sum(weights == 4)
A5 = sum(weights == 5)
```

Gives the weight distribution of $A_0 = 1$, $A_1 = 31$, $A_2 = 465$, $A_3 = 4495$, $A_4 = 13020$ and $A_5 = 14756$ where A_i is the number of weight i errors.

Question 5

Design and implement an encoder using the generator polynomial for this BCH code.

The encoder with the generator polynomial was implemented using the cyclic properties of the code. The message, $c(X)$, is shifted by $n-k$ and divided by the generator polynomial to get the parity check bits by $b(X) = remainder[\frac{X^{n-k}c(X)}{g(X)}]$. The parity

check bits are combined with the message for a systematic code. The function below shows our implementation.

```
function codeword = polBCHencoder(msg)
genPol = [1 0 0 0 1 1 1 1 1 0 1 0 1 1 1 1];
c = [msg zeros(1,15)];
[~, b] = gfdeconv(fliplr(c), fliplr(genPol));
b = [zeros(1,15 - length(b)) fliplr(b) zeros(1,16)];
codeword = b + [zeros(1,15) msg];
```

Question 6a

Use MATLAB defined functions (eg. the decode function) to decode the BCH code.

This decoder is implemented using MATLABs function *decode()*. The matrix *trt* is the syndrome decoding table calculated above using the *syndtable()* function.

```
function decoded_data = matlabBCHdecode(data_to_decode)
load trt
bchPol = [1 0 0 0 1 1 1 1 1 0 1 0 1 1 1 1];
decoded_data = decode(data_to_decode,31,16,'cyclic/fmt', bchPol, trt);
```

Question 6b

Design and implement a decoder using the syndrome decoding table in Task 3 for the BCH code.

The syndrome can be obtained by multiplying received codeword with the transposed parity-check matrix H. The corresponded error pattern is found by performing a look up in the trt table calculated above. The corrected code is then the codeword plus the error pattern. Our implementation is shown below.

```
function msg = syndLookupDecode(codeword)
load trt
H = [1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0 1 1 1 1 0 0 0
     0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0 1 1 1 1 0 0
     0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0 1 1 1 1 0
     0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0 1 1 1 1
     0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 1 0 0 1 1 1 1 1 1 1
     0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0 0 0 0 0 0 1 1 1
     0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 1 1 1 0 1 0 1 1 1 1 0 1 1
     0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 1 1 0 1 1 0 0 0 1 0 1
     0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1 0 1 1 0 0 0 1 1 0 1 0
     0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1 0 1 1 0 0 0 1 1 0 1
     0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 1 0 1 0 1 0 1 0 1 1 1 1 0
     0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 1 1 0 1 0 1 0 1 0 1 1 1 1 1
     0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 1 0 0 1 0 1 1 0 1 0 1 1 1
     0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 1 0 0 0 1 0 0 0 1 0 0 1 1
     0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0 1 1 1 1 0 0 0 1];
syndrome = bi2de(mod(codeword * H',2),'left-msb');
errLocations = trt(1+syndrome,:);
correctedcode = mod(errLocations+codeword,2);
msg = correctedcode(:,16:end);
```

Question 6c

Design and implement a decoder using the Berlekamps iterative procedure.

The implementation of Berlekamp's iterative procedure below used different notation to the textbook specified in *http://www.mathworks.com.au/help/comm/ug/error-detection-and-correction.html* allowing easier implementation of the procedure.

```
function [data, decoded_data, error_locations] = ...
                    berlekamp_decode(data_to_decode)

% Parameters of n = 31, k = 16, t = 3 BCH
n = 31;
t = 3;
k = 16;
m = 5; % 2^5 = 32

% GF field vars
zero = gf(0, m);
one = gf(1, m);
alpha = gf(2, m);

codeword = gf(data_to_decode, m);

% compute the syndromes
S =  codeword*(alpha.^([1:2*t]'*fliplr(0:30)))';

L = 0;
k = -1;
sigma = [one gf(zeros(1,(2*t - 1)), m)];
D = [zero one gf(zeros(1,(2*t - 2)), m)];

for n = 0:(2*t - 1)
    discrepency = fliplr(S((n-L + 1):(n + 1)))*sigma(1:(L + 1))';
    if discrepency ~= 0
        sigma_star = sigma - discrepency*D;
        if L < n - k
            L_star = n - k;
            k = n - L;
            D = sigma/discrepency;
            L = L_star;
        end
        sigma = sigma_star;
    end
    D = [zero D(1:(end-1))];
end


% Find roots of sigma
rootsOfSigma = [];
rootToTry = alpha.^(1:31);
for index = 1:length(rootToTry)
    % if sum is zero then is root
    if sigma*(rootToTry(index).^(0:(length(sigma) - 1)))' == zero
        rootsOfSigma = [rootsOfSigma index];
    end
end
degree = find(sigma ~= zero);
degree = degree(end) - 1;
error_locations = zeros(1,length(data_to_decode));
if length(rootsOfSigma) < degree
    % disp('WARNING: Cannot correct more than 3 errors')
```

```
    decoded_data = data_to_decode;
    data = decoded_data(16:end); % assuming a systematic code
    return
end
error_locations(rootsOfSigma) = ones(1,length(rootsOfSigma));
decoded_data = mod(data_to_decode + error_locations,2);
data = decoded_data(16:end); % assuming a systematic code
```

Sound analysis

Question 7

Simulate the implemented BCH encoder and decoder (using ANY method in Part 6) using the attached wave file (austinpowers.wav) in a BSC channel for different transition probabilities. Discuss the impact of changing different transition probability values.

The channel was simulated using the following function:

```
function playAudioOverBSC(pVal)
[data, Fs] = wavread('austinpowers.wav', 'native');
data = de2bi(data);
% add a row of zeros as it is not divisible by 16
data = double(reshape([data; zeros(1,8)],[],16));
errors = rand(length(data),31) < pVal;


encoded_data_no_error = encoder(data);
encoded_data = mod(encoder(data) + errors,2);

% Check whether error rate is correct :P
errors = encoded_data_no_error == encoded_data;
error_rate = length(find(errors == 0))/numel(errors)

decoded_data = reshape(matlabBCHdecode(encoded_data),[], 8);
% remove the last row of zeros again
decoded_data = bi2de(decoded_data(1:end-1,:));
soundsc(double(decoded_data),Fs);
figure
plot(double(decoded_data));
plot_title = sprintf('Audio␣over␣BSC␣channel␣with␣BCH␣decoding␣(p␣=␣%.2
    f)', pVal);
xlabel('Audio␣Sample␣number')
ylabel('Normalized␣audio␣magnitude')
title(plot_title)
```

Different transition probabilities were input into the function. With a transition probability of below 0.05 the sound could be heard with a little noise audible. As p increased above 0.15 the noise increased to the level where it could no longer be clearly heard. Above 0.2 it was difficult to hear anything at all other than noise.

The plots for different p-values are shown below as a qualitative indicator of noise levels.

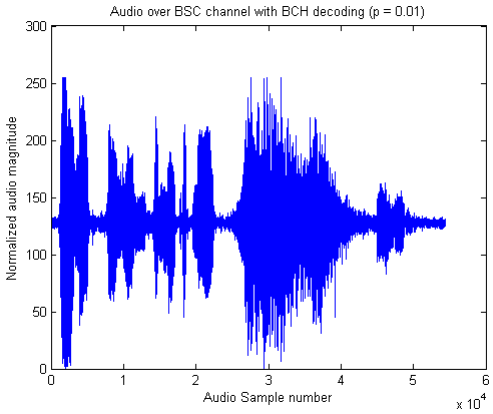

Figure 1: BCH Audio over BSC (p = 0.01)

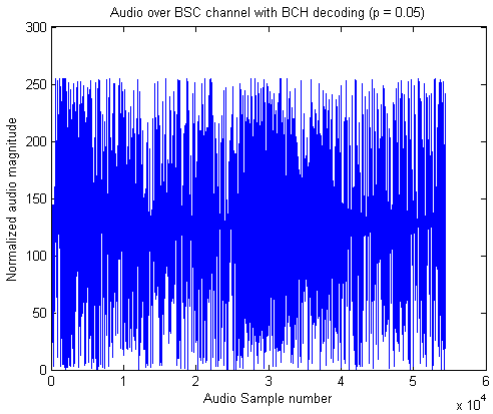

Figure 2: BCH Audio over BSC (p = 0.05)

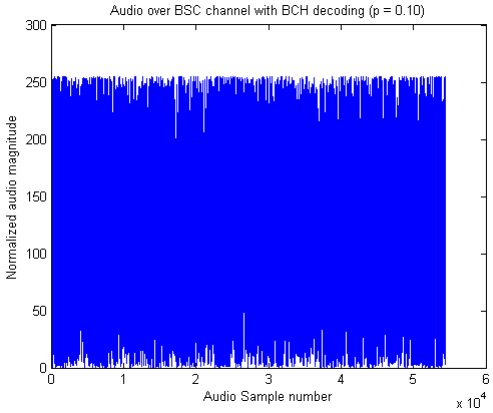

Figure 3: BCH Audio over BSC (p = 0.1)

BER analysis

Question 8a and 8b

Simulation over BSC Simulate the BCH code in a BSC channel (you are allowed to use MATLAB defined functions) and plot the BER versus transition probability for coded and uncoded systems on the same graph. Plot BER versus $\frac{E_b}{N_0}$ *for coded and uncoded systems on the same graph by assuming that the SNR (=* $\frac{E_b}{N_0}$ *) is related to the transition probability for the coded system via* $X_{dB,coded} = 10\log_{10}(\frac{[Q^{-1}(p)]^2}{R})$ *and that for uncoded system is* $X_{dB,uncoded} = 20\log_{10}(Q^{-1}(p))$*, where* $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{+\infty} e^{-\frac{z^2}{2}} dz$ *and* $0 < p < 0.5$*.*
Simulation over AWGN channel with BPSK modulation $\{1, 1\}$ *Simulate the BCH code in an AWGN channel (you are allowed to use MATLAB defined functions) and plot the BER versus the signal to noise ratio (SNR) for a BPSK coded and uncoded systems on the same graph by using a hard-decision demodulator and binary decoder.*

```
% simulations section 1) part 8)
% Find the BER over a BCH code
pValues = 0.001:0.001:0.35;
snrValues = -5:0.001:10;
repetitions = 500;

msg = rand(length(pValues)*repetitions, 16) > 0.5;
errors = rand(length(msg), 31) < ...
        repmat(reshape(repmat(pValues,repetitions,1),1,[])',1,31);
bch_encoded = encoder(msg);

bch_decoded = matlabBCHdecode(mod(bch_encoded + errors, 2));
bitErrs = reshape((sum(mod(bch_decoded' + msg',2))/16)', repetitions,
    []);
BER = sum(bitErrs)/length(bitErrs); % take the average

figure()
plot(pValues, BER);
title('Plot of BER in BCH codes in a BSC');
xlabel('P values');
ylabel('BER');

% Q(x) = 0.5 * ( 1 - erf(x/sqrt(2)))
% Q^(-1)(x) = sqrt(2)*erfinv(1 - 2x)

Xuncoded = 20.*log10(sqrt(2).*erfinv(1 - 2.*pValues));
Xcoded = Xuncoded - 10.*log10(16/31); % R = code Rate = k/n

figure()
plot([Xuncoded; Xcoded], BER);
title('Plot of BER in BCH codes in a BSC');
legend('Uncoded', 'Coded');
xlabel('SNR');
ylabel('BER');

% section 1) q 8)b)

% SNR = 10*log10((1/ampl)^2), ampl = 10^(-SNR/20)
msg = rand(length(snrValues)*repetitions, 16) > 0.5;
bch_encoded = encoder(msg);
```

```
noise = randn(length(bch_encoded), 31).* ...
      repmat(reshape(repmat(10.^(-snrValues./20),repetitions,1) ...
             ,1,[])',1,31);
signal = 1 - 2.*bch_encoded;
bch_decoded = matlabBCHdecode((signal + noise) < 0); % Hard decision on
     0
bitErrs = reshape((sum(mod(bch_decoded' + msg',2))./16)', repetitions,
    []);
BER = sum(bitErrs)/length(bitErrs); % take the average

noise = randn(length(msg), 16).* ...
      repmat(reshape(repmat(10.^(-snrValues./20),repetitions,1) ...
             ,1,[])',1,16);
signal = 1 - 2.*msg;
decoded = (signal + noise) < 0;
bitErrs = reshape((sum(mod(decoded' + msg',2))./16)', repetitions, []);
BER2 = sum(bitErrs)/length(bitErrs); % take the average

figure()
plot(snrValues, [BER2; BER]);
title('Plot␣of␣BER␣in␣BCH␣codes␣in␣an␣AWGN␣BPSK');
legend('Uncoded', 'Coded');
xlabel('SNR');
ylabel('BER');
```

This produced the following plots.

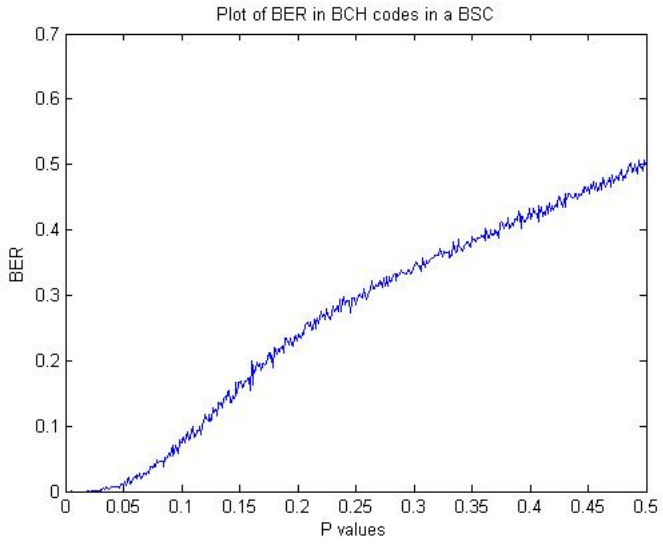

Figure 4: Bit Error Rate vs. transition probability for BCH over BSC

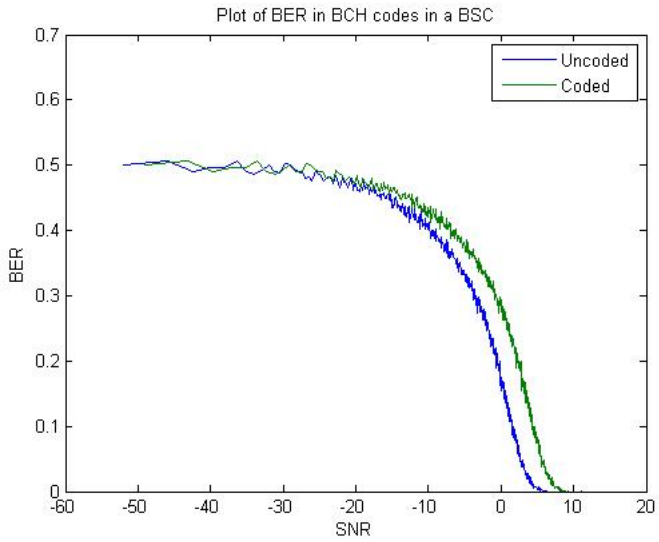

Figure 5: Bit Error Rate vs. SNR for BCH over BSC

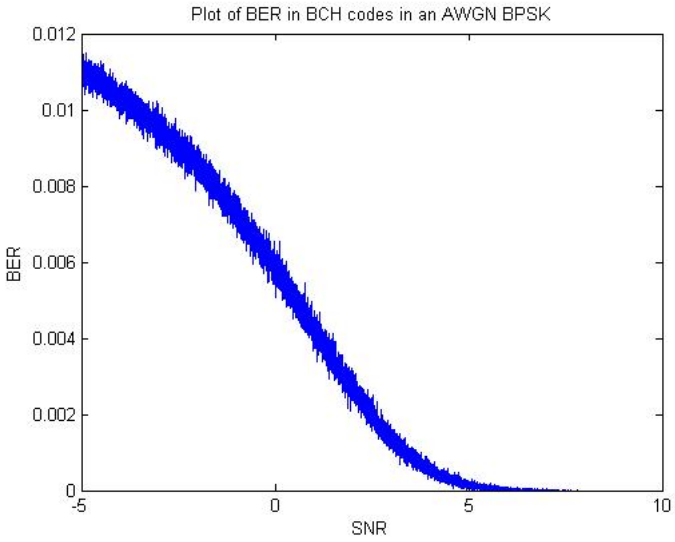

Figure 6: Bit Error Rate vs. SNR for BCH over AWGN

TODO: RESAVE AWGN plot

Question 8c

Draw a table detailing the coding gain for BER$= [10^2, 10^3, 10^4, 10^5, 10^6]$ *by reading the differences between the BER curves for BPSK coded and uncoded systems which have been obtained from simulations in part (a) and (b).*

BER	Coding Gain (Hard decision)	Coding Gain (BSC)
10^{-2}		
10^{-3}		
10^{-4}		
10^{-5}		
10^{-6}		

Table 1: Coding gain for BCH

Question 8d

Find the asymptotic coding gain when $\frac{Eb}{N0}$ is very large from the formula which is given in the lecture notes and compare with simulation results.

Using the formula $G = 10\log(R(t+1))$ in lecture 5 for large $\frac{Eb}{N0}$ where $R = 16/31$ and $t = 3$ results in a coding gain of $G = 3.1482$ dB. Comparing this to the simulation results **...**

Discussion

Answer here

Section II - LDPC

Analysis

Background

Basics of LDPC

LDPC Coding and Decoding using MATLAB Communications Toolbox

Design

Blah blah blah

Implementation

See below. ..hahaa sadas

Calculations and Results

You are an engineer whose job is to design and analyze the performance of error control codes for different clients. Client 2 requires a code for satellite transmission of digital TV. The satellite is power limited and very high reliability is required (as close to Shannon capacity as possible). Low decoding complexity is desired, but is not essential.

Question 1

Design the code (e.g. type of code, code parameters, and other relevant information) and justify its design

To meet these requirements an LDPC code is used

For client 2, it is clear that an LDPC code will be necessary and these can provide performance that is close to the Shannon limit for an AWGN channel, compared to other modern codes. However, it is known that this is the case for very long block codes, which require computationally complex [TODOREF]. An existing solution for digital satellite television is present in the DVB-S2 standard, which uses LDPC as part of concatenated code with BCH coding [1]. However, the drawback of this scheme is the high decoding complexity, though this can be reduced somewhat by the special structure of the code [TODOREF]. Another factor to consider is the power limitation of the satellite, since a large LDPC block code will also entail high coding complexity, more hardware gates and thus more power. An alternative code choice is the (1152, 2304) LDPC block code used in IEEE-802.16-2005, also known as WiMax [TODOREF]. This code is also irregular, which is known to exhibit superior performance over regular LDPC codes.

TODO: Expand on justification, summarize into table

Sound analysis

Question 2

Simulate the chosen code using the attached wave file (austinpowers.wav) in a BSC channel for different transition probabilities (you are allowed to use MATLAB defined functions. Discuss the difference in sound quality compared to the BCH code in Section I, for different transition probabilities.

The code was simulated for the transition probabilities p = $\{0.01, 0.05, 0.1\}$. As a qualitative tool, plots of the waveform are included. Figures 1-3 provide a visual indicator of the noise.

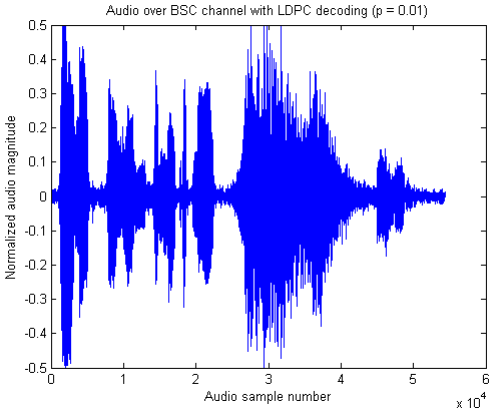

Figure 7: LDPC Audio over BSC (p = 0.01)

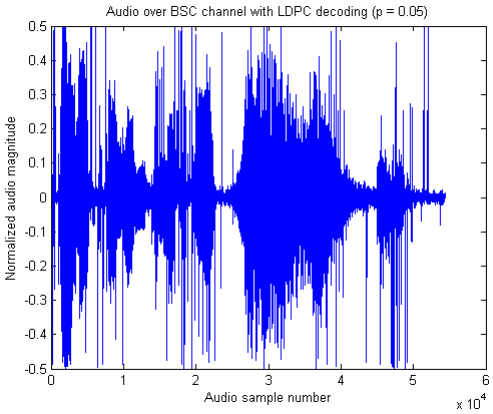

Figure 8: LDPC Audio over BSC (p = 0.05)

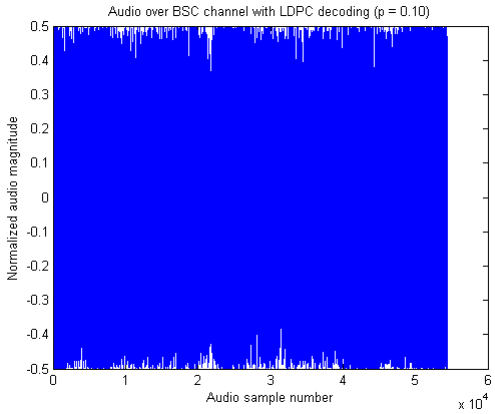

Figure 9: LDPC Audio over BSC (p = 0.1)

As the bit errors increase due to the transition probability, the noise becomes more prevalent. However, it is improved relative to BCH in the case of p = 0.05, showing LDPC has superior performance in this regard compared with BCH. However, there is not much difference in the case of p = 0.1, showing that the code has reached it's performance limitation for the particular channel, which is due to effective hard-decision decoding that is involved in a BSC channel. TODO: is it?

Question 3a

Simulate the chosen code in a BSC channel (you are allowed to use MATLAB defined functions) and plot the BER versus transition probability for coded and uncoded systems on the same graph. Plot BER versus $\frac{E_b}{N_0}$ *for coded and uncoded systems on the same graph by assuming that the SNR (*$= \frac{E_b}{N_0}$ *) is related to the transition probability for the coded system via* $X_{dB,coded} = 10\log_{10}(\frac{[Q^1(p)]^2}{R})$ *and that for uncoded system is* $X_{dB,uncoded} = 20\log_{10}(Q^1(p))$*, where* $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{+\infty} e^{-\frac{z^2}{2}} dz$ *and* $0 < p < 0.5$*.*

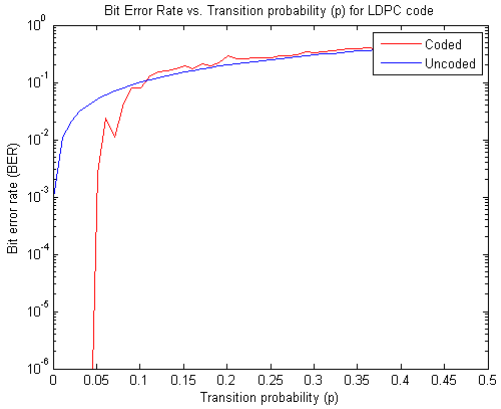

Figure 10: BER vs transition probability for LDPC over BSC

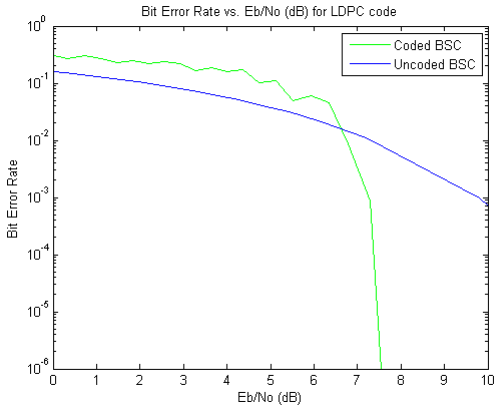

Figure 11: BER vs EbNo for LDPC over BSC

BER analysis

Question 3b

Simulate the chosen code in an AWGN channel (you are allowed to use MATLAB defined functions) and plot the BER versus the signal to noise ratio (SNR) for a BPSK coded and uncoded systems on the same graph by using a hard-decision demodulator and binary decoder. Simulate the chosen code in an AWGN channel and plot the BER versus SNR on the same graph for a BPSK coded system by using a soft-decision decoder.

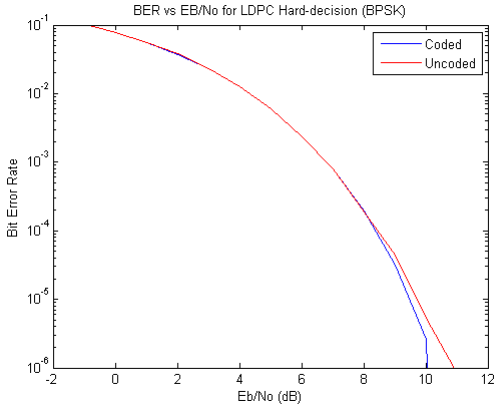

Figure 12: BER vs EbNo for Hard-decision demod and decoding (LDPC - AWGN)

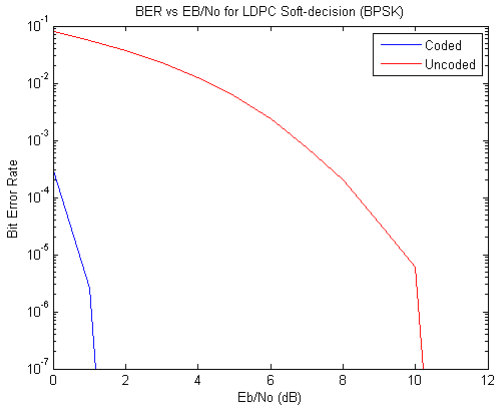

Figure 13: BER vs EbNo for Soft-decision demod and decoding (LDPC - AWGN)

Question 3c

Draw a table detailing the coding gain for BER$= [10^2, 10^3, 10^4, 10^5, 10^6]$ *by reading the differences between the BER curves for BPSK coded and uncoded systems which have been obtained from simulations in part (a) and (b).*

BER	Coding Gain (Hard decision)	Coding Gain (Soft decision)	Coding Gain (BSC
10^{-2}			
10^{-3}			
10^{-4}			
10^{-5}			
10^{-6}			

Table 2: Coding gain for LDPC

Question 3d

Find the asymptotic coding gain when $\frac{Eb}{N0}$ *is very large from the formula which is given in the lecture notes and compare with simulation results.*

Answer here

Discussion

Question 4

Discuss the advantages/disadvantages of the code chosen in this section with the BCH code in Section 1 (eg. complexity, coding gain, efficiency)

Answer here

References

[1] Alberto Morello and Vittoria Mignone. Dvb-s2: the second generation standard for satellite broad-band services. *Proceedings of the IEEE*, 94(1):210–227, 2006.