ELEC 5507 Error Control Coding
Group Project

Overview

In this project, students need to implement state-of-the-art forward error correcting (FEC) encoder(s) and decoder(s) in MATLAB, simulate and analyze the performance of the implemented FEC coding scheme(s).

The project will consist of two sections, corresponding to different types of FECs. Section I will be based on the BCH code, and is compulsory to all groups. This will be worth 60% of the total assignment. In Section II, each group will choose ONE client and code. This will be worth 40% of the total assignment, and is dependent on the particular code chosen. Please note that the percentage levels are not an accurate assessment on the difficulty levels or how much time should be spent on the projects. You should not treat this indicator as the sole basis of making your decision. Other factors should be as important, such as your work load from other UoS, your past performance, your target performance, your self study ability, your interest, etc.

1 Section I (60% of total assignment)

In Section 1, we will focus on the primitive binary BCH code with parameters $n = 31$, $k = 16$, and $t = 3$.

1. What is the generator polynomial for this code? Use MATLAB's 'bchgenpoly' function to verify your answer.

2. What is the minimum distance of this code?

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.

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

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

6. Decoding:

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

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

c. Design and implement a decoder using the Berlekamp's iterative procedure.

7. Simulations and sound analysis

   a. 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.

8. Simulations

   a. 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 E_b/N_0 for coded and uncoded systems on the same graph by assuming that the SNR ($= E_b/N_0$) is related to the transition probability for the coded system via $X_{dB,coded} = 10\log_{10}\left(\frac{[Q^{-1}(p)]^2}{R}\right)$ and that for uncoded system is $X_{dB,uncoded} = 20\log_{10}\left(Q^{-1}(p)\right)$, where $Q(x) = \frac{1}{\sqrt{2\pi}}\int_x^{+\infty} e^{-\frac{z^2}{2}}dz$ and $0 < p < 0.5$.

   b. 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**.

   c. 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). The table will thus look like:

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

   d. Find the asymptotic coding gain when E_b/N_0 is very large from the formula which is given in the lecture notes and compare with simulation results.

2 Section II (40% of total assignment)

You are an engineer whose job is to design and analyze the performance of error control codes for different clients. Each client has provided specifications for the type of code they require for their application. Each group is to design a SINGLE code for only ONE client.

- **Client 1 Specifications:** Client 1 requires an error control code for a satellite communication system. Legacy systems which contain efficient implementation of Viterbi decoders are to be used. Bandwidth is an issue, and hence at least half of the transmitted bit stream has to be equal to the total message length.

- **Client 2 Specifications:** 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.

The table lists below the types of code each group may choose in the project. Each group may only choose ONE code, and the code chosen will reflect on the percentage level of this section. Please note that the percentage levels are not an accurate assessment on the difficulty levels or how much time should be spent on the projects. You should NEVER treat this indicator as the sole basis of making your decision. Other factors should be as important, such as your work load from other UoS, your past performance, your target performance, your self study ability, your interest, etc.

Code No.	Code	% Total Assignment
1	BCH Code	40
2	Hamming Code	40
3	Convolutional Code	45
4	LDPC Code	50

For Code No. 3 and 4, the total percentage level is over 100%. If a student gets more than 100% in the project, the extra percentage will be reflected into bonus percentages of the course assessment. For example, if a student chooses Project No. 4 for Section 2, and gets full marks for both Section 1 and Section 2, he/she will get 110%. The extra 10% will be counted as bonus percentage points for the course assessment.

Please note that if you choose the projects with bonus percentages, you may need to spend significant time and effort in learning the materials not covered in the lectures/tutorials. However, that will definitely be beneficial to your telecom career in addition to the bonus points.

2.1 Tasks

1) For the client you have chosen

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

b. Justify its design.

2) Simulations and sound analysis

   a. 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)

   b. Discuss the difference in sound quality compared to the BCH code in Section I, for different transition probabilities.

3) Simulations and BER analysis

   a. Simulation over BSC

      * 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 E_b/N_0 for coded and uncoded systems on the same graph by assuming that the SNR ($= E_b/N_0$) is related to the transition probability for the coded system via $X_{dB,coded} = 10\log_{10}\left(\frac{[Q^{-1}(p)]^2}{R}\right)$ and that for uncoded system is $X_{dB,uncoded} = 20\log_{10}\left(Q^{-1}(p)\right)$, where $Q(x) = \frac{1}{\sqrt{2\pi}}\int_x^{+\infty} e^{-\frac{z^2}{2}}dz$ and $0 < p < 0.5$.

   b. Simulation over AWGN channel with BPSK modulation ($\{-1, 1\}$)

      * 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 systemby by using a **soft-decision decoder**.

   c. 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). The table will thus look like:

   d. Find the asymptotic coding gain when E_b/N_0 is very large from the formula which is given in the lecture notes and compare with simulation results.

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

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

3 When to Start

You should start to work on your project as early as possible. Please note that you should NOT wait until the relevant materials being covered in the lecture. Otherwise, you will not have enough time to finish your project. You should reserve enough time for debugging and documentation. If you are not familiar with communication system simulation in MATLAB, you should spend some time to get yourself familiar before your project allocation is finalised, and attend the tutorials.

Note that for some projects, you may need to teach yourselves ahead of the lecture schedule.

Confining yourselves within the coverage of the lectures will likely prevent you from completing the project satisfactorily. Self-study and research ability is one of the assessment components in this Unit of Study.

4 Project Report

A report has to be included in the project submission. The report should contain no more than 15 pages including references with 10pt font at minimum.

Generally, a report should have the following sections:

1) Introduction: gives an overview of your project;

2) Theoretical background: describes the theories/principles/algorithms involved in your project;

3) Calculation: answers the questions about the calculation of the performance etc. (if your project does not include any of the questions, you can skip this section);

4) Design & Implementation: describes the design and implementation details of the coding system in your project. This is the most important part of the report. It must describe the ideas included in your design and implementation, the structure of your codes, key variables/data structures, flow-charts of key codes/functions, and so on.

5) Sound Analysis and BER Analysis: presents the details of the analysis and comments.

6) Conclusion: describes your view of this project, the key difficulties, how the project fared in meeting the requirements, etc.

7) References: lists all references here.

5 What to Hand In

By the due date, you need to hand in the following materials:

1) Cover sheet signed by each member in your group;

2) Program Source Code

3) Assignment Report

4) A note about how your program can be run: the platform (Matlab version number), how to launch each program, the parameters to be set (if applicable), etc. You should also indicate how long it took you to run each program.

You need to submit by email in a zip format to mahyar.shirvanimoghaddam@sydney.edu.au by 6PM Friday30^{th} May 2014.

6 Presentation on Your Project

In the last week of this semester, each group has to do a presentation on their project. Each group will be given 10 minutes for their presentation and 5 minutes for question time. Each group can select one representative to do the presentation or they can partition the presentation task among members.

7 Important Dates

- Project Groups: 30^{th} April 2014 6PM
- Project Submission: 30^{th} May 2014 6pm
- Project Presentation: 4^{th} June 2014 6:00pm-8:00pm

8 Marking

- Project report: 50%
- Presentation: 40%
- Group mark: 10 % (Each group member will give a mark to the other members, which basically show the participation percentage in the project.)

9 Programming Language and Platform

In this project, you need to write program(s) in MATLAB to simulate the performance of some FEC coding schemes. [Please differentiate Matlab from Matlab Simulink. Simulink is not to be used in this project.] A familiarity with MATLAB programming is necessary. The Communications Toolbox provides off-the-shelf standard modules in communication systems, which includes random source, FEC encoders/decoders (CODECs), channels, modulations, etc.

If you are not familiar with MATLAB, there is a tutorial on MathWorks' website, http://www.mathworks.com/academia/studentcenter/ tutorials/ launchpad.html, which could help you develop the general skill in MATLAB programming. MATLAB is a widely used high-level language and interactive environment in engineering simulation and modelling. With its abundant functions, modules and toolboxes, you can develop your simulation systems in a quicker and more efficient way than with traditional programming language such as C/C++ and Fortran. Being proficient in MATLAB will be a great advantage for your future career.

10 About Plagiarism

On the Internet or from other resources, you might obtain documentations or program codes, which are similar or can be modified to be used in your project. The direct copy from these codes/resources is strictly prohibited. However, learning from reading others' designs/codes is not a bad thing. Please include a reference list in your report/document in that case.

When you hand in your completed project, a detailed report/document has to accompany the program codes. Meanwhile, the program codes your submitted need to be well commented.

The quality of your report/documents is one of the biggest factors that determine your mark for the project. The project presentation is another.

If there is any suspect of plagiarism, your group might need to attend an interview with the lecturer. If the plagiarism is confirmed, your group will get zero mark for the project and further heavy penalty might be imposed.

Please refer to the University's policy on plagiarism
$http://www.usyd.edu.au/senate/policies/Plagiarism.pdf$

11 Tips

1) *Programming representation of codewords*: A codeword can be represented by a vector of n-tuple $\mathbf{v} = (v_0, v_1, ..., v_{n-1})$ or a code polynomial $v(x) = v_0 + v_1 x + ... + v_{n-1}x^{n-1}$, where $\{v_i\}_{i=0,1,...,n-1}$ are binary digits. In a program, the above two representations might not be convenient and efficient. For a compact representation, we can convert the binary sequence $(v_0, v_1, ..., v_{n-1})$ to ONE decimal number. Thus, the above codeword can be represented as $v(2) = v_0 + v_1.2 + ... + v_{n-1}.2^{n-1}$, in which all operations

are in the real number field. For example, a codeword $(1, 0, 0, 1, 1, 1)$ can be represented as a decimal number 57. Note that the rightmost digit v_{n-1} in the codeword is the most significant bit (MSB). Alternatively, you can use the leftmost digit v_0 as the MSB as long as you are consistent everywhere and clearly indicate in your report which convention you use. The MATLAB's functions 'de2bi' and 'bi2de' are convenient tools to convert between binary sequences and decimal numbers.

2) *The digit order in a codeword:* In the lecture notes, we relate a codeword vector $\mathbf{v} = (v_0, v_1, ..., v_{n-1})$ with the code polynomial $v(x) = v_0 + v_1 x + ... + v_{n-1} x^{n-1}$. However, MATLAB uses a slightly different notation, where the codeword bits are reversed. That is the above codeword is expressed as $\mathbf{v} = (v_{n-1}, v_{n-2}, ..., v_0)$. This is the same case for message sequence. For a message polynomial $c(x) = c_0 + c_1 x + ... + c_{k-1} x^{k-1}$, the vector representation in MATLAB is $\mathbf{c} = (c_{k-1}, c_{k-2}, ..., c_0)$. You should use a consistent convention if you want to mix your own codes with MATLAB functions, such as 'bchenc', 'bchdec', etc.

3) *Conversion between GF(2) arrays and real arrays:* In MATLAB, messages and codewords are represented as Galois array of symbols in GF(2). Assuming that you have a Galois array ga and a real array ra, you can convert from one to another as follows:

$$ga = gf(ra) \text{ and } ra = double(ga.x).$$

4) *LDPC:* The LDPC code in WiMAX (Standard: IEEE Std 802.16-2004 / IEEE Std 802.16e-2005) is based on a set of fundamental LDPC codes. Each of the fundamental codes is a systematic linear block code. By using code rate and block size adjustment, the fundamental codes can accommodate various code rates and packet sizes. If you have chosen LDPC, you can consider one fundamental LDPC code used in WiMAX. A rate-1/2 LDPC code with parameters $k = 1152$, $n = 2304$ bits. The parity check matrix of this code is an $(n-k)$ by n matrix $\mathbf{H}$. It is defined through an $(n_b - k_b)$ by n_b base model matrix $\mathbf{H}_{bm}$, where $n_b = 24$, $k_b = 12$. The base matrix is given by:

-1	94	73	-1	-1	-1	-1	-1	55	83	-1	-1	7	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	27	-1	-1	-1	22	79	9	-1	-1	-1	12	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	24	22	81	-1	33	-1	-1	-1	0	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1	-1
61	-1	47	-1	-1	-1	-1	-1	65	25	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1	-1
-1	-1	39	-1	-1	-1	84	-1	-1	41	72	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	46	40	-1	82	-1	-1	-1	79	0	-1	-1	-1	-1	0	0	-1	-1	-1	-1	-1
-1	-1	95	53	-1	-1	-1	-1	-1	14	18	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1	-1
-1	11	73	-1	-1	-1	2	-1	-1	47	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1	-1
12	-1	-1	-1	83	24	-1	43	-1	-1	-1	51	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1	-1
-1	-1	-1	-1	-1	94	-1	59	-1	-1	70	72	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	-1
-1	-1	7	65	-1	-1	-1	-1	39	49	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0
43	-1	-1	-1	-1	66	-1	41	-1	-1	-1	26	7	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	0

The base model matrix is expanded to the parity check matrix as follows. The expansion factor $z = n/nb = 2304/24 = 96$. Each element in $\mathbf{H}_{bm}$ is replaced by a z by z sub-matrix. An element of -1 is replaced by a z by z zero matrix. An element of 0 is replaced by a z by z identity matrix. An element of $p(p > 0)$ is replaced by a z by z identity matrix circularly right shifted p times.