ECEN 604: Course Project

Due Thursday 12/12/13

Reading:

- For project ideas, read below and browse textbook.

Overview:

The purpose of the class project is to expose to you to more advanced practical and theoretical issues in channel coding. The project should be completed in groups of two. The report is due on the day after finals and should be about 5 pages of typeset text including the project description, results, figures, and references. I suggest typesetting the report using LaTeXor LyX (a simplified front-end for LaTeX), but this is not required. Students are allowed to choose their topic of the project as long as it has a close connection to the material presented in this class. For those interested in a well-defined assignment, there is also a standard project that involves the concatenation of an outer Reed-Solomon code with an inner convolutional code.

Choices:

1. (Concatenating a Reed-Solomon and Convolutional Code)
   For students interested in practical applications, I suggest simulating a concatenated coding system with an outer Reed-Solomon code and an inner convolutional code (e.g., data is encoded first with a RS code then a convolutional code). This is a very common coding system that has been used by NASA/JPL and in the Digital Video Broadcasting (DVB) standard. For example, a common choice of component codes is a (255,239,17) RS code and a rate-1/2 convolutional code with constraint-length 7. The 8-bit words of the RS code are interleaved (with depth from 2-8) before encoding by the convolutional code. Decoding is done using the Viterbi algorithm for the convolutional code and any appropriate decoder for the RS code. We will cover the details of the Viterbi algorithm when we study convolutional codes.

2. (Exploring Iterative Decoding)
   To a large extent, modern communication systems now use codes based on sparse random graphs and iterative decoding. Two prototypical examples of these codes are turbo codes [1] and low-density parity-check (LDPC) codes [2, 3]. It turns out that there is a natural iterative decoder that can be applied to any binary linear code (e.g., see Chap. 15 in Moon). In this project, the goal is to implement the iterative decoder for arbitrary binary linear code and use it to make some comparisons. For example, you could compare iterative and algebraic decoding for BCH codes of moderate length (e.g., $n = 255$ to $n = 1023$). You might also compare the performance with sparse random code with the same parameters (n, k).

3. (Generalized Product Codes) [4, 5]
   Let $\mathcal{C}_1$ be an (n_1, k_1) linear code and $\mathcal{C}_2$ be a (n_2, k_2) linear code. The $n_1 \times n_2$ matrix $\underline{X}$ is a

codeword of the *product code* $\mathcal{C}$ formed by $\mathcal{C}_1$ and $\mathcal{C}_2$ if every column is a codeword in $\mathcal{C}_1$ and every row is a codeword in $\mathcal{C}_2$. For parity-check matrices $\underline{H}_1, \underline{H}_2$, we can define the product code by $\underline{H}_1 \cdot \underline{X} = \underline{0}$ and $\underline{X} \cdot \underline{H}_2^T = 0$. Due to a low-complexity syndrome-updating decoder, these codes are now becoming popular for systems that require high data rates and high code rates. The goal of this project is to implement an iterative algebraic decoder for the product of two binary BCH codes and to test its performance. For example, one might consider the product of two $(63, 51)$ $t = 2$ error-correcting BCH codes. For $t = 2$ error-correcting BCH codes, there are some tricks that allow very fast decoding based on lookup tables (LUTs). For this project, you need to implement your own BCH decoder based either on the fast LUT approach or the general purpose Berlekamp-Massey algorithm. Once the basic decoder is working, there are also a number of interesting variations that can be tried fairly easily.

4. (Alternate Project)
   For students who would like to pick their own project, the task is to find a topic that is closely related to the material in this class. For ideas, try reading ahead in the book and considering how coding might be integrated into your research. The topic must be approved in advance by the instructor.

Tasks:

1. Take some time to understand the problem, related algorithms, and their analysis.

2. Perform a minimal simulation that evaluates its performance and effectiveness. This may be completed in any programming language, but MATLAB is typically the easiest.

3. Write a short report describing the technique and application (e.g., 2-3 pages per person). It is often helpful to think of this as a short tutorial designed to explain what you've done to your fellow classmates. Do not list the simulation source code in your report.

4. Submit a printed copy of your report and e-mail a zip file containing an electronic copy of your report and all source code. One should be able to execute a main program that generates all the results plotted in your report.

Note: Plagiarism is a very serious offense in Academia. Any figures in the paper not generated by you should labeled "Reproduced from [...]". Any portions of any simulation code (e.g., MATLAB, C, etc...) not written by you be clearly marked in your source files. The original source of any mathematical derivation or proof should be explicitly cited.

References

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in *Proc. IEEE Int. Conf. Commun.*, vol. 2, (Geneva, Switzerland), pp. 1064–1070, IEEE, May 1993.

[2] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. 8, pp. 21–28, Jan. 1962.

[3] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399–431, March 1999.

[4] R. M. Pyndiah, "Near-optimum decoding of product codes: Block turbo codes," *IEEE Trans. Commun.*, vol. 46, pp. 1003–1010, Aug. 1998.

[5] K. Zigangirov, A. J. Feltstrom, M. Lentmaier, and D. Truhachev, "Encoders and decoders for braided block codes," in *Proc. IEEE Int. Symp. Inform. Theory*, (Seattle, WA), pp. 1808–1812, July 2006.