Gallager codes

Information Theory (APMA 1710), Fall 2011

In this assignment, you will implement and test the Gallager codes. (Suggestion: For this assignment, Matlab is probably the easiest language to use.)

(I) Visualize parity check and generator matrices

Included with this assignment is a Matlab function called `construct_gallager.m` that takes arguments c, d, m (where c = number of ones per column, d = number of ones per row, and m = number of rows in the parity check matrix) and returns a Gallager code parity check matrix H and a corresponding generator matrix G. (Note: m must be a multiple of c.) There is also an optional 4th argument, *visualize*, that displays the matrices H and G when set to 1.

Using $c = 3$ and $d = 7$, and choosing *visualize* $= 1$, call `construct_gallager.m` with the following values of m: $m = 75$, $m = 300$, $m = 600$, and $m = 900$. (You do not need to submit anything for this part.)

(II) Implement the decoding algorithm

Also included with the assignment is pseudocode for the decoding algorithm, in the file `pseudocode.pdf`. Implement the decoding algorithm.

(III) Implement code to evaluate performance

We will evaluate the Gallager codes' performance in essentially the same way as we did for Hamming codes. You can reuse your code from Hamming for steps (1), (2), (3), and (4) below. Implement code for (5).

(1) Randomly sample a binary sequence $s = s_1 s_2 \cdots s_K$, where each s_i is drawn from a Bernoulli(1/2) distribution. This s will be our source message. (For convenience, we will assume that K is a multiple of the block length k.)

(2) Encode the source sequence s using the generator matrix G, producing a "transmitted" sequence $t = t_1 t_2 \cdots t_N$ where $N = nK/k$. (Suggestion: By arranging s in a $k \times (K/k)$ matrix,

the encoding can be done with a single matrix multiplication. In fact, I would encourage you to represent all of the sequences s, t, u, r, and $\hat{s}$ (defined below) in matrix form.)

(3) Randomly sample a binary sequence $u = u_1 u_2 \cdots u_N$, where each u_i is drawn from a Bernoulli(α) distribution. This u will represent the noise in the channel.

(4) Compute the binary sequence $r = r_1 r_2 \cdots r_N$ such that $r_i \equiv t_i + u_i$ (where $\equiv$ denotes congruence mod 2). This r represents the "received" sequence.

(5) For each block of n symbols in the received sequence r, compute the parity check vector z by multiplying by H mod 2. (As above, this can be done with a single matrix multiplication.) Using your implementation of the decoding algorithm from part (II), perform error correction on each such parity check vector z, and use the results to produce the decoded sequence $\hat{s} = \hat{s}_1 \hat{s}_2 \cdots \hat{s}_K$.

(IV) Evaluate the performance of Gallager codes

(1) Choose the following parameter values: $c = 3$, $d = 7$, $\alpha = 0.05$, and $K = 24000$. For each of $m = 75$, $m = 300$, $m = 600$, and $m = 900$, run your code from part (III) and print the following quantities:

$$m \quad n \quad k \quad R \quad \hat{p}_b$$

where $R = k/n$ is the rate, and as in the Hamming codes assignment, we estimate the probability of bit error p_b using

$$\hat{p}_b = \frac{1}{K} \sum_{i=1}^{K} I(\hat{s}_i \neq s_i).$$

(2) Repeat (1) using $\alpha = 0.025$.

(3) Describe the trends you see in $\hat{p}_b$.

(4) What is the capacity C of this channel, for each of these values of α?

(5) The values of c and d above were chosen so that the rate of these codes is the same as the rate of the $(7, 4)$ Hamming code. Recall that for the $(7, 4)$ Hamming code, $p_b \approx 9\alpha^2$. For the values of α above (0.05 and 0.025), how does the performance (in terms of p_b) of the Gallager codes above compare with the performance of the $(7, 4)$ Hamming code (using the approximation $p_b \approx 9\alpha^2$ for the $(7, 4)$ Hamming code)?

(Extra Credit) Sampling a Gallager code parity check matrix using MCMC

Learn about the Metropolis algorithm for MCMC (Markov chain Monte Carlo), and explain why the function `binary_matrix_MCMC.m` produces a sample approximately from the uniform distribution on $m \times n$ binary matrices with all column sums c and all row sums d.