Simulating Low-Density Parity-Check Codes

Elena Martínez, Miliano Mikol, Dr. Edward Mosteig

Frank R. Seaver College of Science and Engineering, Loyola Marymount University, Los Angeles, CA

Loyola Marymount University McNair Scholars Program

Abstract

All communication channels are subject to corruption, which can have implications on how high performance technology transmits data. The need for efficient and accurate encoding and decoding has inspired new methods of protecting data. Low-density parity-check codes (LDPCs) incorporate sparse redundancy to protect the original message bits without incurring too much decoding complexity. We seek to empirically show whether LDPC schemes designed deterministically improve upon the accuracy of LDPC schemes designed using randomization. A binary erasure channel was programmed using Python 3 to simulate three classes of LDPCs: Gallager, MacKay Neal, and a unique method developed throughout the course of this research. We compare and report the accuracy of the three classes in recovering an original message from a corrupted message.

Background

- Noise refers to interference with a message that may cause the signal received to differ from the signal transmitted. No communication channel is truly noiseless, so we must figure out how to transmit messages in the face of this corruption.
- Parity bits are combined with the bits of an original message to create a codeword, which is then transmitted across a communication channel. Increasing the number of parity bits leads to greater redundancy, which offers protection to the original message bits. However, the greater the number of bits transmitted, the more "expensive" the design is. The amount of redundancy added needs to provide enough protection for the message bits and not be too expensive.
- Low-density parity-check codes are constructed so the number of parity bits is not overwhelmingly large.

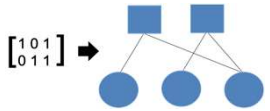


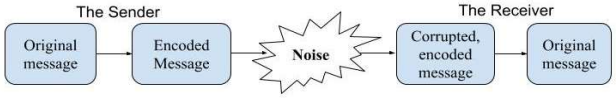
Figure 1. LDPC parity-check matrices may be represented by a Tanner graph where squares correspond to check equations (rows), circles correspond to bits in a codeword (columns), and edges correspond to ones.

Research Question

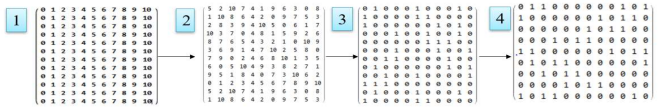
Could deterministic LDPC encoding schemes improve upon the efficiency and accuracy of current schemes?

Methodology

- Tools: Python 3, Wolfram Mathematica, MATLAB R2019a.
- A communication channel with an encoder, noise, and decoder was modeled in Python 3.
- A Binary Erasure Channel (where each bit has a fixed probability of being erased) was created to corrupt codewords using rate ¼ LDPCS (there are three parity-check bits for every single message bit).
- The parity-check matrices simulated are 9x12. Generator matrices were produced for encoding. The following decoding schemes were simulated: Gallager, MacKay-Neal, and a unique encoding scheme developed over the course of this research.
- Accuracy in recovering the pre-corrupted codeword were compared for each encoding scheme.



A Deterministic Approach



The goal is to create a 9x12 parity-check matrix. Here is an example using our deterministic approach:

1. Begin with a 11x12 matrix with identical rows filled from 0-10 across. This represents the current positions of the numbers in every row.
2. The positions of the numbers in each row are mapped according to f(x) = 5x+8 mod 13.
3. Digits 0, 1, 2 are replaced by 1 and all digits 3, 4, 5, 6, 7, 8, 9, 10, 11 are replaced by 0.
4. An all zero column (column 5) is produced and must be removed along with the last column (column 11). The resulting 12x9 matrix is transposed to produce a 9x12 parity-check matrix.

Proposition (M_, M_, M_): The deterministic approach we have constructed will always result in at least one all-constant row in Step 3

Proof:

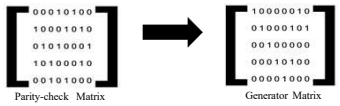
Define $m \equiv b(1-a) \bmod p$. We will demonstrate that $f(ma+b) \equiv ma + b$ which is equivalent to $f(ma+b) - (ma+b)$. This follows from the observation

$$\begin{aligned} f(ma+b) - (ma+b) &\equiv a(ma+b) + b - (ma+b) \\ &\equiv a\big(b - m(1-a)\big) \\ &\equiv a\big(b - b(a-1)^{-1}(1-a)\big) \\ &\equiv a(b-b) \\ &\equiv 0 \end{aligned}$$

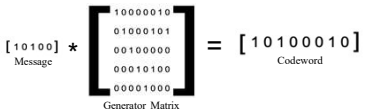
The example above can be generalized by permuting rows using various maps of the form f(x) = ax + b mod p, where p is a prime number and a and b are fixed constants. The form used should produce a matrix of distinct rows.

Simulation

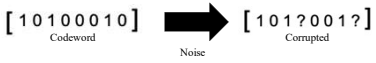
To encode and decode messages, we need a generator matrix and a parity-check matrix, respectively. After a parity-check matrix is constructed (using Gallagher, MacKay Neal, or our own method above), one can construct a corresponding generator matrix using an algorithm that uses Gaussian elimination.



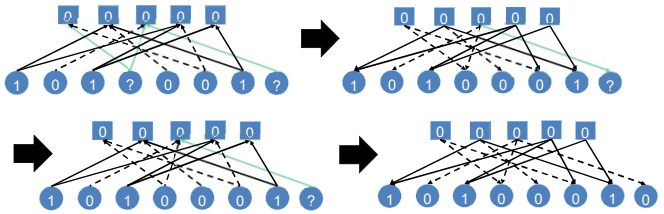
Step 1. The generator matrix is multiplied (vector-matrix multiplication) by a given message to generate a codeword (the generator matrix encodes the message).



Step 2. The code is sent through a Binary Erasure Channel (BEC) and corrupted as a result.



Step 4. The corrupted message is decoded using a belief propagation algorithm and the parity-check matrix as illustrated in a Tanner graph†.



System of check equations

Check Equation = C_i
Codeword Bit = B_i

$C_1 = B_4 \oplus B_6$
$C_2 = B_1 \oplus B_5 \oplus B_7$
$C_3 = B_2 \oplus B_4 \oplus B_8$
$C_4 = B_1 \oplus B_3 \oplus B_7$
$C_5 = B_3 \oplus B_5$

†
Solid Arrow = 1
Dashed Arrow = 0
Green Arrow = Unknown

Results

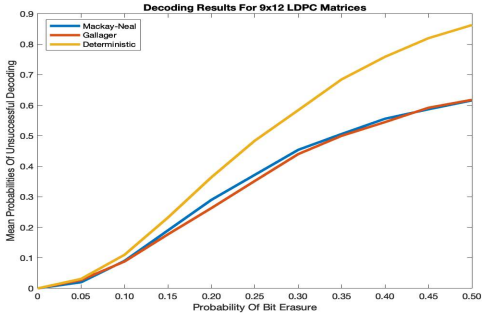


Figure 2. For each scheme, 100,000 test cases were created in the following manner: 10 random parity matrices were constructed. 10,000 simulations were run in total for each matrix with 1,000 simulations per probability of bit erasure.

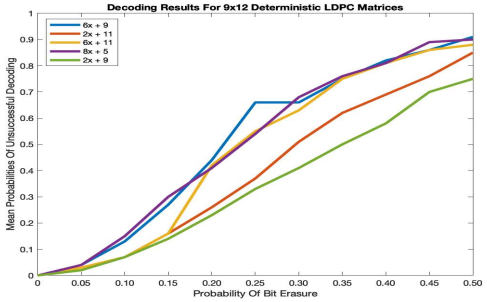


Figure 3. Each scheme is labeled by the row permutation equation as the method of permuting matrix elements appears to alter performance. A single matrix was generated for each scheme. 10,000 simulations were run in total for each matrix with 1,000 simulations per probability of bit erasure

Conclusion

The sample of 10 deterministic matrices analyzed, although comparable, on average produced a higher rate of error than the existing LDPC codes (Gallager and MacKay Neal). However, when the deterministic matrices were compared to each other, there was a significant range in performance. Certain equations produced matrices that were closer in performance to the Gallager and MacKay Neal matrices than others. Further research could be performed to find a method to systematically select equations for the deterministic matrices that perform closer to and potentially better than the existing LDPCs. A shortcoming of this method is that in order to produce a 9x12 matrix (for this case), another column, besides the all zero column, had to be removed. The last column was selected, but this removes protection from certain bits and not others. Removing another column could perhaps lead to different results. Overall, it is worth exploring this approach of producing deterministic low-density parity-check codes in comparison to LDPCs that contain a component of randomization.

References

1. S.J. Johnson, Introducing Low-Density Parity-Check Codes, University of Newcastle, Australia, 2006.
2. D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," IEEE Trans. Inform. Theory, vol. 45, no. 2, p399–431, March 1999.
3. D. J. C. MacKay and Rp.. M. Neal, "Near Shannon limit performance of low density parity check codes," Electron. Lett., vol. 32, no. 18, pp. 1645–1646, March 1996, reprinted Electron. Lett, vol. 33(6), pp. 457– 458, March 1997.
4. R. G. Gallager, Low-Density Parity-Check Codes. Cambridge, MA: MIT Press, 1963.

Acknowledgements

Thank you Dr. Edward Mosteig for serving as our research advisor, the McNair Scholars Program for providing the funds that made our research possible, and Loyola Marymount University for providing an avenue for undergraduates to experience research.