Information Theory and Communication
Channel Capacity

Ritwik Banerjee
rbanerjee@cs.stonybrook.edu
Roadmap

- Types of Channels
- Jointly Typical Sequences
- Channel Coding Theorem
- Zero-Error Codes
- Fano’s Inequality
- Converse to the Channel Coding Theorem
- Hamming Codes
Randomization is the method used to prove the theorem, but it’s not how signals are communicated through a channel.

We proved that the average over all codes of block length $n$ has a small probability of error.

The best code within this set can be found by an exhaustive search.

This means that revealing the best code $C^*$ to both the sender and the receiver doesn’t require a channel.

Both simply agree to use the best $(2^{nR}, n)$ code for the channel.

The theorem shows the existence of good codes with small error probability.

But it doesn’t tell us how to construct such a code.

That brings us to **Hamming codes**, one of the simplest “error-correcting” codes.
Fano’s Inequality

**Theorem**

*Fano’s Inequality* For a discrete memoryless channel with a codebook $C$ and the input message $W$ uniformly distributed over $2^{nR}$, we have

$$H(W|\hat{W}) \leq 1 + P_e^{(n)} n R$$

**Theorem**

Let $Y^n$ be the result of passing $X^n$ through memoryless channel of capacity $C$. Then,

$$I(X^n;Y^n) \leq nC \quad \forall \ p(x^n)$$

(proof shown in class)
Zero-Error Codes

Theorem

If the average error probability is zero, then the rate is bounded above by the channel capacity, i.e.

\[ P_e^{(n)} = 0 \implies R \leq C \]

(proof shown in class)
Linear Error Correcting Codes

- Shannon’s channel coding theorem tells us that if the redundancy is above a certain threshold, then the error probability can be made arbitrarily small by increasing the blocklength $n$ of the message.
- The proof is based on randomization, so we can’t really use it’s approach to design good codes.
- We will need some structure in our encoding and decoding functions.
- For this, we will study a simple class of codes called linear block codes.
- Linearity leads to a scenario where the analysis of the code’s error correcting ability is easier.
- The use of matrices to encode/decode messages is much easier than a random codebook.
Linear Error Correcting Codes

- **Error detection** means that the code can detect errors but we don’t know where the errors are in a received sequence.

- **Error correction** means we know where the errors are and can correct the bit positions in error.
  
  - If a code can correct up to, say, $t$ errors, then, if the sequence contains more than $t$ errors, the decoder may decode incorrectly.

- For linear codes, **encoding** is a linear transformation $c$ that maps a message $m$ to a codeword $c(m)$.

- In **block codes**, all codewords $c(m)$ have the same length, which we will denote by $n$.

- Let $k$ be the length of information bits that we encode. Then there are $n - k$ redundancy bits in each codeword. These are called **parity check bits**, which are used for error correction.

- The pair $(n, k)$ specifies the parameters for a block code.
Hamming Distance

- The **Hamming Distance** $d_H$ between two strings is the number of positions in which the two strings differ. E.g., $d_H(101101, 100110) = 3$.
- Let $d^*$ be the minimum Hamming distance between any two distinct codewords in $C$. That is,
  $$d^* = \min_{c_i \neq c_j} d_H(c_i, c_j)$$
- A code with minimum distance $d^*$ between codewords can detect $d^* - 1$ errors and can correct $(d^* - 1)/2$ errors. That is, to correct one error, the minimum distance must be 3.
- For binary sequences, a linear code means that the encoding function $c$ is linear over $\mathbb{Z}_2$.
  $$c(m + m') = c(m) + c(m'),$$
  where the addition is modulo 2.
- The **Hamming weight** $w(s)$ of a binary string $s$ is defined as the sum of its non-zero entries.
A linear code is completely defined by all the codewords for messages of weight 1. For example, to encode 01101, we simply add the basis vector codewords:

\[ c(01101) = c(01000) + c(00100) + c(00001) \]

Because we can use just the basis vectors, there is no need for a codebook with \(2^k\) entries.

All we need is a matrix that defines the linear transformation of all the weight-1 messages. This is a \(k \times n\) matrix \(G\) called the generator matrix.

For a message \(m\), the encoding is \(c = mG\).