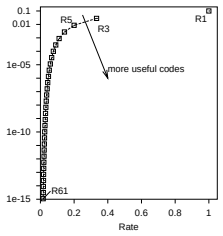
Lecture 17:

Proving Shannon's Second Theorem

November 8, 2006

How Good Are Simple Codes? 1

- Shannon's noisy coding theorem says we can get the probability of error in decoding a block, p_B, arbitrarily close to zero when transmitting at any rate, R, below the capacity, C — if we use good codes of large enough length, N.
- For repetition codes, as N increases, $p_B \to 0$, but $R \to 0$ as well.
- For Hamming codes, which we will study next week, as N increases, $R \to 1$, but $p_B \to 1$ as well, since there's bound to be more than one error in a really big block.



Good Codes Aren't Easy to Find 2

- In the 60 years since Shannon's noisy coding theorem, many schemes for creating codes have been found, but most of them *don't* allow one to reach the performance promised by theorem.
- They can still be useful. For example, error correction in computer memory necessarily works on fairly small blocks (eg, 64 bits). Performance on bigger blocks is irrelevant.
- But in other applications — computer networks, communication with spacecraft, digital television — we could use quite big blocks if it would help with error correction.
- How can we do this in practice?

Getting to Capacity for the BEC 3

- We *can* get near-error-free transmission for the binary erasure channel, at any rate below capacity, using a practical method.
- We use a linear $[N, K]$ code, defined by a set of $M = N - K$ parity-check equations:

$$\begin{aligned} c_{1,1}\, v_1 + c_{1,2}\, v_2 + \cdots + c_{1,N}\, v_N &= 0 \\ c_{2,1}\, v_1 + c_{2,2}\, v_2 + \cdots + c_{2,N}\, v_N &= 0 \\ &\vdots \\ c_{M,1}\, v_1 + c_{M,2}\, v_2 + \cdots + c_{M,N}\, v_N &= 0 \end{aligned}$$

- For the BEC, any bit received as 0 or 1 is guaranteed to be correct. To decode, we fill in these known values in the equations above, and then try to solve for the unknown values, where the bit was received as an erasure ('?').
- This can be done either iteratively (as you will do in the upcoming assignment) or via more computationally intensive methods.

When Will BEC Decoding Succeed?

- If the probability of an erasure is f, and N is large, there will very likely be around Nf erasures in the received data, assuming the parity checks are independent (this is just the Law of Large Numbers).
- So the decoder will be solving M equations in U unknowns, where U is very likely to be near Nf
- These equations will be *consistent*, since the correct decoding is certainly a solution.
- The correct decoding will be the *unique* solution — which the decoder is guaranteed to find — as long as U out of the M equations are independent.

Picking the Code at Random

- Suppose we pick a code — specified by the parity-check coefficients, c_{ij} — *at random*. (!!!)
- How likely is it that the equations that we need to solve to decode a transmission that has U erasures will have a unique solution?
- Imagine randomly picking the parity-check equations *after* we receive the transmission with U erasures. How many equations would we expect to have to pick to get U independent equations?
- Once we have i independent equations, the probability that the next equation picked will be dependent on these will be

$$\frac{2^i}{2^U} = \frac{1}{2^{U-i}}$$

since there are 2^i ways of combining the previous equations, and 2^U possible equations.

Picking the Code at Random (Continued)

- The expected number of dependent equations picked before we get U independent ones is

$$\sum_{i=0}^{U-1} \frac{1}{2^{U-i}} \left(1 - \frac{1}{2^{U-i}}\right)^{-1} = \sum_{i=0}^{U-1} \frac{1}{2^{U-i} - 1}$$

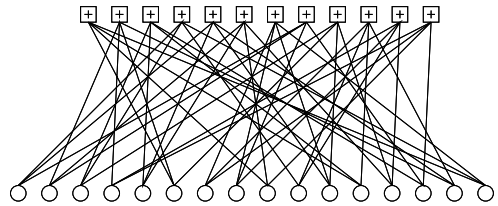
Reordering the terms, we can see that this is small:

$$1 + 1/3 + 1/7 + \cdots < 1 + 1/2 + 1/4 + \cdots < 2$$

- Hence, we likely need M to be only slightly larger than U, which is likely to be no more than slightly larger than Nf.
- So with a random code, we will be likely to correct all erasures when N is large as long as $f < M/N = (N-K)/N = 1 - R$. In other words, as long as $R < 1-f$.
- As we saw before, the capacity of the BEC is equal to $1-f$, so we've achieved the promise of Shannon's theorem.

What about the BSC?

- A similar argument using randomly-chosen codes is used in the proof of Shannon's noisy coding theorem for the BSC.
  We'll look at a sketch of this proof.
- But unlike the random codes for the BEC (which are practical), the random codes used in this proof are completely impractical.
- This was why we needed to consider random codes of a different kind, whose parity-check matrices are mostly zeros. These are the "Low Density Parity Check Codes", which can be used in practice, and allow near-error-free transmission at close to capacity.



Shannon's Noisy Coding Theorem for the BSC

- Consider a BSC with error probability $f < 1/2$.
  This channel has capacity $C = 1 - H_2(f)$.
- **Theorem**: For any desired closeness to capacity, $\eta > 0$, and for any desired limit on error probability, $\epsilon > 0$, there is a code of some length N whose rate, R, is at least $C - \eta$, and for which the probability that nearest neighbor decoding will decode a codeword incorrectly is less than ϵ.
- We can now give a proof of this, which more-or-less follows the proof for general channels in Chapter 10 of MacKay's book.
- The idea is based on showing that a *randomly chosen code* performs quite well and hence that there must be *specific codes* which also perform quite well.

Strategy for Proving the Theorem

- Rather than showing how to construct a specific code for given values of f, η, and ϵ, we will consider choosing a code of a suitable length, N, and rate $\log_2(M)/N$, by picking M codewords *at random* from Z_2^N.
- We consider the following scenario:
  1. We randomly pick a code, $\mathcal{C}$, which we give to both the sender and the receiver.
  2. The sender randomly picks a codeword $\mathbf{x} \in \mathcal{C}$, and transmits it through the channel.
  3. The channel randomly generates an error pattern, $\mathbf{n}$, and delivers $\mathbf{y} = \mathbf{x} + \mathbf{n}$ to the receiver.
  4. The receiver decodes $\mathbf{y}$ to a codeword, $\mathbf{x}^*$, that is nearest to $\mathbf{y}$ in Hamming distance.
- If the probability that this process leads to $\mathbf{x}^* \neq \mathbf{x}$ is $< \epsilon$, then there must be some specific code with error probability $< \epsilon$.

Rearranging the Order of Choices

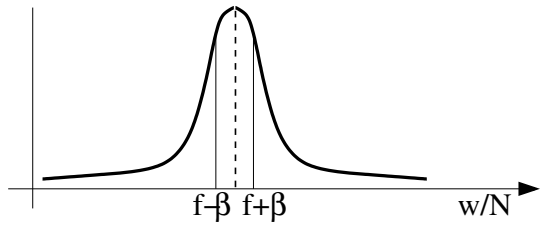
- It will be convenient to rearrange the order in which random choices are made, as follows:
  1. We randomly pick *one* codeword, $\mathbf{x}$, which is the one the sender transmits.
  2. The channel randomly generates an error pattern, $\mathbf{n}$, that is added to $\mathbf{x}$ to give the received data, $\mathbf{y}$. Let the number of transmission errors (ie, ones in $\mathbf{n}$) be w.
  3. We now randomly pick the other $M-1$ codewords. If the Hamming distance from $\mathbf{y}$ of all these codewords is greater than w, nearest-neighbor decoding will make the correct choice.
- The probability of the decoder making the wrong choice here is the same as before. Sneaky huh?

The Typical Number of Errors

- If N is large, we expect that close to Nf of the N bits in a codeword will be received in error. In other words, we expect the error vector, $\mathbf{n}$, to contain close to Nf ones.
- Specifically, the Law of Large Numbers tells us that for any $\beta > 0$, there is some value for N such that:

$$P(f - \beta < w/N < f + \beta) \;\geq\; 1 - \epsilon/2$$

  where w is the number of errors in $\mathbf{n}$. We'll say that error vectors, $\mathbf{n}$, for which $f - \beta < w/N < f + \beta$ are "typical".



How Many Typical Error Vectors Are There? 12

- How many error vectors, $\mathbf{n}$, are ther for which $f - \beta < w/N < f + \beta$?
- If $\mathbf{n}$ is such a typical error pattern with w errors, then

$$P(\mathbf{n}) \;=\; f^{w}(1-f)^{N-w} \;>\; f^{N(f+\beta)}(1-f)^{N(1-f-\beta)}$$

- Let J be the number of typical error vectors. Since the total probability of all these vectors must not exceed one, we must have

$$J f^{N(f+\beta)}(1-f)^{N(1-f-\beta)} \;<\; 1$$

and hence

$$J \;<\; f^{-N(f+\beta)}(1-f)^{-N(1-f-\beta)}$$

- Equivalently,

$$\begin{aligned} J &< 2^{N(-(f+\beta)\log_2(f)-(1-f-\beta)\log_2(1-f))} \\ &< 2^{N(H_2(f)+\beta\log_2((1-f)/f))} \end{aligned}$$

Decoding with Typical Error Patterns 13

- The probability that the codeword nearest to $\mathbf{y}$ is the correct decoding will be at least as great as the probability that the following sub-optimal decoder decodes correctly:

  If there is exactly one codeword $\mathbf{x}^*$ for which $\mathbf{n} = \mathbf{y} - \mathbf{x}^*$ has a typical number of ones, then decode to $\mathbf{x}^*$, otherwise declare that decoding has failed.

- This sub-optimal decoder can fail in two ways:
  - The correct decoding, $\mathbf{x}$, may correspond to an error pattern, $\mathbf{n} = \mathbf{y} - \mathbf{x}$, that is not typical.
  - Some other codeword, $\mathbf{x}'$, may exist for which the error pattern $\mathbf{n}' = \mathbf{y} - \mathbf{x}'$ is typical.

Bounding the Probability of Failure (I) 14

- The total probability of decoding failure is less than the sum of the probabilities of failing in these two ways.
  We will try to limit each of these to $\epsilon/2$.
- We can choose N to be big enough that

$$P(f - \beta < w/N < f + \beta) \;\geq\; 1 - \epsilon/2$$

  This ensures that the actual error pattern will be non-typical with probability less than $\epsilon/2$.
- We now need to limit the probability that some other codeword also corresponds to a typical error pattern.

Bounding the Probability of Failure (II) 15

- The number of typical error patterns is

$$J \;<\; 2^{N(H_2(f)+\beta\log_2((1-f)/f))}$$

- For a random codeword, $\mathbf{x}$, other than the one actually transmitted, the corresponding error pattern given $\mathbf{y}$ will contain 0s and 1s that are independent and equally likely.
- The probability that one such codeword will produce a typical error pattern is therefore

$$J/2^{N} \;<\; 2^{-N(1-H_2(f)-\beta\log_2((1-f)/f))}$$

- The probability that *any* of the other $M-1$ codewords will correspond to a typical error pattern is bounded by M times this. We need this to be less than $\epsilon/2$, ie

$$M\,2^{-N(1-H_2(f)-\beta\log_2((1-f)/f))} \;<\; \epsilon/2$$

Finishing the Proof

- Finally, we need to pick β, M, and N so that the two types of error have probabilities less than $\epsilon/2$, and the rate, R is at least $C - \eta$.
- We will let $M = 2^{\lceil (C-\eta)N \rceil}$, and make sure N is large enough that $R = \lceil (C-\eta)N \rceil / N < C$.
- With this value of M, we need

$$2^{\lceil (C-\eta)N \rceil}\, 2^{-N(1-H_2(f)-\beta \log_2((1-f)/f))} < \epsilon/2$$
$$\Rightarrow 2^{-N(1-H_2(f)-\lceil (C-\eta)N \rceil/N-\beta \log_2((1-f)/f))} < \epsilon/2$$

- The channel capacity is $C = 1 - H_2(f)$, so that $1 - H_2(f) - \lceil (C-\eta)N \rceil / N \;=\; C - R$ is positive.
- For a sufficiently small value of β, $1 - H_2(f) - \lceil (C-\eta)N \rceil / N - \beta \log_2((1-f)/f)$ will also be positive. With this β and a large enough N, the probabilities of both types of error will be less than $\epsilon/2$, so the total error probability will be less than ϵ.

Good Codes and Minimum Distance

- Recall that for a code to be guaranteed to correct up to t errors, it's minimum distance must be at least $2t + 1$.
- What's the minimum distance for the random codes used to prove the noisy coding theorem?
- A random N-bit code is very likely to have minimum distance $d \leq N/2$ — if we pick two codewords randomly, about half their bits will differ. So these codes are likely *not guaranteed* to correct patterns of $N/4$ or more errors.
- A BSC with error probability f will produce about Nf errors. So for $f > 1/4$, we expect to get more errors than the code is guaranteed to correct. Yet we know these codes are good!
- **Conclusion:** A code may be able to correct *almost all* patterns of t errors even if it can't correct *all* such patterns.