1 The Story So Far

To date, we’ve almost completely resolved Shannon theory (errors occur at random.) Shannon showed an upper bound and matching existential lower bound with his seminal coding theorem, and last week we saw Forney’s algorithmic result for actually constructing optimal codes in polynomial time.

Unfortunately, we haven’t been so lucky with Hamming theory. There’s a reasonably-sized gap between our MRRW upper bound and GV existential lower bound, and our Forney/Justensen constructions are lower still. Even worse, using these codes to correct errors cuts their utility in half.

TODO: Insert diagram from class notes

Today we’re going to explore list-decoding, which relaxes our notion of perfect decoding. This will allow us to shorten the gaps considerably.

2 Definition

The essence of list decoding is that the decoder is allowed to produce a list of possible decodings. If there aren’t too many errors during transmission, this list is required to contain the correct decoding among its members.

Which of the above models are we using here? We’re concerned with correcting errors, so it might seem like Shannon – however, our analyses will assume that errors are produced maliciously (like Hamming.)

Let’s define this more formally. An encoding function $E : \Sigma^k \to \Sigma^n$ is $(p, L)$ list-decodable if there exists a list-decoding function $D : \Sigma^n \to (\Sigma^k)^L$ such that for $m \in \Sigma_k$, $\zeta \in \text{Ball}(0, pn)$, $m \in D(E(m) + \zeta)$.

We can also look at this pictorially. Each dot in the following diagram represents a member of a language $C \in \Sigma^n$. The surrounding circles have radii $pn$.

TODO: Diagram

The code $C$ is $(p, L)$ list-decodable if $\forall y \in \Sigma^n$, $|\text{Ball}(y, pn) \cap C| \leq L$. 

3 Characteristics

There are a couple of interesting points to note before analyses.

- The above definition is combinatorial, rather than computational. In particular, $D$ is not required to be efficient – this allows us to use the brute-force searching strategies replete in earlier classes.

- We usually fix $L$ to be poly-sized; we want to analyze how $R$ varies as $p$ changes for $(p, \text{poly})$ list-decodable codes.

4 Matching Bounds

With this relaxation, we can find matching upper and lower bounds. The upper bound is just an application of Shannon’s Converse Coding Theorem.

**Theorem 1** The rate of a $(p, \text{poly})$ list-decodable code is at most $1 - H(p)$.

**Proof** If we had a list-decodable code that violated this, we could convert it into a Shannon-style code by randomly picking an element in the list. This new code would have inverse-polynomial error probability – Shannon’s theorem claims that this error rate would have to be inverse-exponential.

We’re going to show two different non-constructive proofs of a matching lower bound. This bound is due to [Zyablov, Pinskei, Blinovskii].

**Theorem 2** There exist $(p, \text{poly})$ list-decodable codes with rate $R = 1 - H(p) + \epsilon$ for any $\epsilon$.

**Proof** We show that a random code has the property we need. Pick $C \subseteq 0, 1^n$ with $|C| = 2^{Rn}$ codewords at random.

For any fixed $y$, the probability that the $i$th codeword is in $\text{Ball}(y, pn)$ is approximately:

$$\frac{2^{H(p)n}}{2^n} = 2^{(H(p)-1)n}$$

Note that this probability is exponentially small in $n$! The probability that there exist $L$ codewords in this ball, then, is just:

$$2^{(H(p)-1)n} \times \binom{2^{Rn}}{L} \leq (2^{(H(p)-1)n} \times 2^{Rn})^L$$

Finally, the probability that there exists some value of $y$ with $L$ codewords within its ball is at most the following, via the Union bound:

$$2^n (2^{(H(p)-1)n} \times 2^{Rn})^L$$

Using the fact that the first term is exponentially small, we arrive at

$$2^n (2^{-\epsilon n})^L$$

This goes to 0 if $L = \Omega(\frac{1}{\epsilon})$
5 Linear Lower Bound

The above proof also works (with modification) for linear codes: in that case, the second term above would be bounded by the probability that there exist log $L$ independent codewords in $Ball(y, pn)$; carrying everything through, it works if $L \geq 2^\frac{1}{\epsilon}$.

We can do even better, though. Consider the following greedy construction:

- Let $C_i$ (the codewords to date) equal span\{b_1, \ldots, b_i\}
- Let $n_i(y) = |Ball(y, pn) \cap C_i|$. For any $y$, $n_i(y)$ measures the current number of codewords within radius $pn$.
- Define a potential function $\Phi_i = E_y[2^{n_i(y)}]$. This function uses exponential weighting to discourage proximity.
- Pick a linearly independent $b_{i+1}$ to minimize $\Phi_{i+1}$ given $b_1, \ldots, b_i$
- Stop when $\Phi_k = 2$, returning $C_k$

Note that we’re going to cheat a bit in the following analyses: we assume that $b_{i+1}$ was picked uniformly at random. Not sure how to fix this.

**Theorem 3** $C_k$ is $(p, n)$ list decodable

**Proof** If not, there would exist a value of $y$ with more than $n$ codewords within its radius. This would contribute more than $2^{n+1}/2^n = 2$ to the potential function, which would be a contradiction.

How many steps will this procedure take? We can bound the gain of the potential function. (Note: not entirely clear on first lemma; will revise soon)

**Lemma 4** $E_{b_{i+1}}[\Phi_{i+1}] \leq \Phi_i^2$

**Proof**

\[
\Phi_{i+1} = \frac{1}{2^n} \sum_y 2^{n_i(y) + n_i(y + b_{i+1})}
= \frac{1}{2^n} \sum_y 2^{n_i(y)} 2^{n_i(y + b_{i+1})}
\]

\[
E_{b_{i+1}}[\Phi_{i+1}] = \frac{1}{2^n} \frac{1}{2^n} \sum_{b_{i+1}} \sum_y 2^{n_i(y)} 2^{n_i(y + b_{i+1})}
= \frac{1}{4^n} \sum_z 2^{n_i(z)} \sum_y 2^{n_i(y)}
= \Phi_i^2
\]
Theorem 5 For $k = (1 - H(p))n - \epsilon$, this construction takes $O\left(\frac{1}{\epsilon}\right)$ steps.

Proof First, let’s calculate the value of $\Phi_0$. Note that even at the beginning, the zero vector is part of the code – thus, the value of $\Phi_0$ is $1 + \frac{2^{H(p)n}}{2n}$.

Using the above lemma, $\Phi_k \leq (1 + \Phi_0)^{2^k}$. If $k$ is sufficiently small, we can use the $(1 + x)^n = 1 + xn$ approximation and arrive at $\Phi_k \leq 1 + 2^k \Phi_0^2$. Thus, the maximum number of steps taken is $O\left(\frac{1}{\epsilon}\right)$.

6 Johnson Bound (unfinished)

How do codes we’ve seen earlier in the class fit into this new setting? The analyses turn out to be fairly difficult. Better if we could relate known work on these codes (e.g. their distance) to list-decodability. The Johnson bound does just that.

Theorem 6 Given an $[n, k, d]_q$ code $C$, $C$ is $(1 - \sqrt{1 - \frac{d}{n}}, \text{poly})$ list-decodable.

Proof Will arrive soon.. ■