Lecture 8: More Results on List Decoding

Error-Correcting Codes (Spring 2016)
Rutgers University
Swastik Kopparty
Scribes: Nathan Fox and Malihe Alikhani

1 The Johnson Bound

Recall the *Johnson bound* from last time:

Proposition 1. *If a code C has minimum relative distance at least δ, then C is (ρ, L) list decodable for $\rho = 1 - \sqrt{1-\delta}$ and $L = n$.*

This raises the question of whether we can list-decode *efficiently* from a ρ-fraction errors.

We have the following improvement of the Johnson bound over the binary alphabet:

Theorem 2. *If a code $C \subseteq \{0,1\}^n$ has minimum relative distance δ, then C is (ρ, L) list decodable with $\rho < \frac{1-\sqrt{1-2\delta}}{2}$ and $L = n + 1$.*

Before we prove this result, let us examine the asymptotics of ρ as δ approaches its extreme values, namely 0 and $\frac{1}{2}$.

The Taylor expansion $\sqrt{1-x}$ about $x = 0$ is $1 - \frac{x}{2} - \frac{x^2}{8} - \cdots$. So, as $\delta \to 0$,

$$\rho \to \frac{1 - \left(1 - \frac{2\delta}{2} - \Theta\left(\delta^2\right)\right)}{2} = \frac{\delta}{2} + \Theta\left(\delta^2\right).$$

If $\delta = \frac{1}{2} - \epsilon$, then

$$\rho < \frac{1 - \sqrt{2\epsilon}}{2} = \frac{1}{2} - \frac{\sqrt{\epsilon}}{\sqrt{2}}.$$

We will now prove this version of the Johnson bound.

Proof. Let C be a binary code with minimum relative distance δ. Suppose $x \in \{0,1\}^n$ is arbitrary and $c_1, c_2, \ldots, c_L \in C$ are codewords such that for all i, $\Delta(x, c_i) < \rho n$. Since each c_i is a codeword, we have that $\Delta(c_i, c_j) \geq \delta n$ for all $i \neq j$.

Let $\Phi : \{0,1\}^n \to \mathbb{R}^n$ be the map sending 0 to 1 and 1 to -1. Define $u = \Phi(x)$, and, for all i, define $v_i = \Phi(c_i)$. Let $\bar{u} = \frac{1}{\sqrt{n}} u$ denote the normalization of u; similarly, for all i, let $\bar{v_i} = \frac{1}{\sqrt{n}} v_i$.

Consider the following inner products. We see that, for all i,

$$\langle \bar{u}, \bar{v_i} \rangle = \frac{n - 2\Delta(x, c_i)}{n} \geq 1 - 2\rho.$$

Also, for all $i \neq j$,

$$\langle \bar{v_i}, \bar{v_j} \rangle = \frac{n - 2\Delta(c_i, c_j)}{n} \leq 1 - 2\delta.$$

Let $\beta > 0$ be a parameter (to be chosen later). We observe that, for $i \neq j$,

$$\langle \bar{v_i} - \beta u, \bar{v_j} - \beta u \rangle \leq (1 - 2\delta) - 2\beta(1 - 2\rho) + \beta^2 = (\beta - (1 - 2\rho))^2 - (1 - 2\rho)^2 + (1 - 2\delta).$$

It $\rho < \frac{1}{2}$, we can take $\beta = 1 - 2\rho > 0$. Doing so makes

$$\langle \bar{v_i} - \beta u, \bar{v_j} - \beta u \rangle \leq -(1 - 2\rho)^2 + (1 - 2\delta)$$

for all $i \neq j$. If $-(1 - 2\rho)^2 + (1 - 2\delta) < 0$, all the vectors $\bar{v_i} - \beta\bar{u}$ have pairwise negative inner products. This means that there are at most $n + 1$ of them.

Finally, we observe that solving the condition $-(1 - 2\rho)^2 + (1 - 2\delta) < 0$ for ρ gives $\rho < \frac{1-\sqrt{1-2\delta}}{2}$. So, C is $(\rho, n + 1)$ list decodable for $\rho < \frac{1-\sqrt{1-2\delta}}{2}$, as required. □

1.1 The Elias-Bassalygo Bound

We will now use the improved Johnson bound to derive a new bound on rate-distance tradeoff for binary codes. The resulting bound, known as the Elias-Bassalygo bound, will improve upon both the volume packing bound and the Plotkin bound.

Let $C \subseteq \{0,1\}^n$ be a code with rate R and relative distance δ. By the Johnson bound, C is $\left(\frac{1-\sqrt{1-2\delta}}{2}, O(n)\right)$ list decodable. Draw balls of radius ρ around every codeword. Every point of $\{0,1\}^n$ is covered at most $n+1$ times. So, $|C| \cdot |B(\rho)| \leq 2^n(n + 1)$. This implies that $|C| \leq \frac{2^n(n+1)}{|B(\rho)|}$. This gives

$$R \leq 1 - H(\rho) + \frac{\log(n)}{n} = 1 - H\left(\frac{1 - \sqrt{1 - 2\delta}}{2}\right) + o(1).$$

Let us analyze the asymptotics of this bound as δ approaches its extreme values of 0 and 1. When $\delta \to 0$,

$$R \leq 1 - H\left(\frac{\delta}{2} + O\left(\delta^2\right)\right) + o(1) \sim 1 - H\left(\frac{\delta}{2}\right).$$

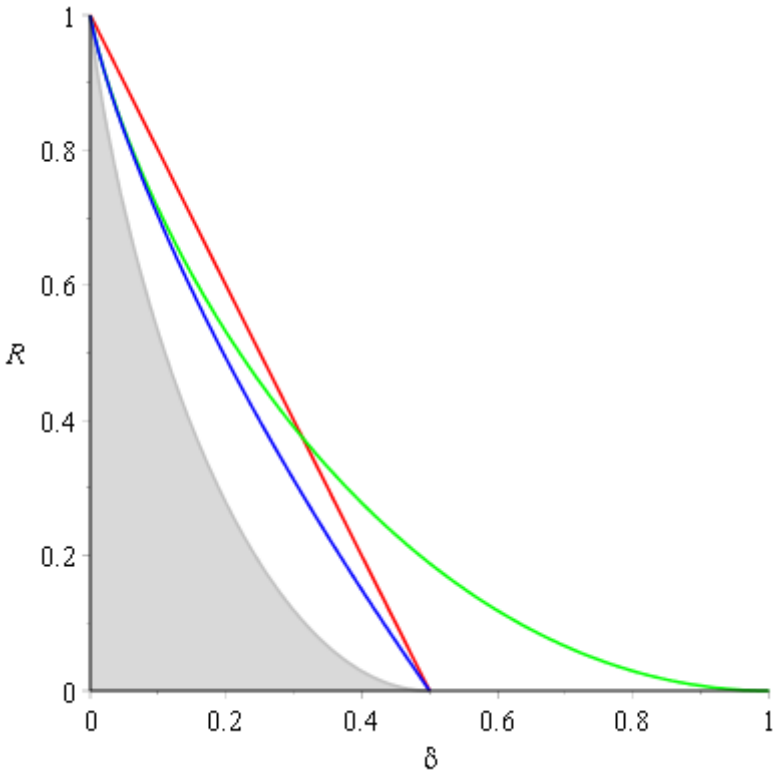
This is like the volume packing bound.

When $\delta = \frac{1}{2} - \epsilon$,

$$R \leq 1 - H\left(\frac{1}{2} - \sqrt{\frac{\epsilon}{2}}\right) = 1 - (1 - \Theta(\epsilon)) = \Theta(\epsilon).$$

This is like the Plotkin bound. In particular, it is very far from the $\Theta\left(\epsilon^2\right)$ Gilbert-Varshamov bound.

Our picture now looks like this:



The grey area is the region below the Gilbert-Varshamov bound, where we know codes exist. The green curve is the volume packing bound. The red curve is the Plotkin bound. The blue curve, which improves both of these, is the Elias-Bassalygo bound.

To summarize, for codes of relative distance $\frac{1}{2} - \epsilon$,

- there exist codes with rate $R > \Omega\left(\epsilon^2\right)$.
- all codes must have $R < O\left(\epsilon\right)$.
- we know how to construct codes with rate $R > \Omega\left(\epsilon^3\right)$, namely Reed-Solomon codes concatenated with a brute-force search.

Later, we will see the linear programming bound, which gives $R < O\left(\frac{\epsilon^2}{\log\left(\frac{1}{\epsilon}\right)}\right)$.

2 List Decoding Reed-Solomon Codes

We want to efficiently list decode Reed-Solomon codes. Eventually, we will be able to do so all the way up to the Johnson bound, though our first algorithm will not be as powerful. Our setting:

- $\mathbb{F}$ is a field.

- $S \subseteq \mathbb{F}$ has cardinality n.
- U is a fixed positive integer.
- Given $r : S \to \mathbb{F}$, we want to find all polynomials $P(x) \in \mathbb{F}[x]$ of degree at most d such that $\Delta(r, P) \leq U$.

At a high level, our algorithm will first try to understand r. Then, it will use this understanding to list decode. The key will be to look at this problem as living in $\mathbb{F}^2$. Here is the first version of our algorithm:

Algorithm 1 (Sudan's Algorithm, Version 1)**.**

1. Find a low degree bivariate polynomial $Q(x, y) \in \mathbb{F}[x, y]$ such that the curve $Q(x, y) = 0$ passes through all the points of r.

   Let the coefficients of Q be undetermined:

   $$Q(x, y) = \sum_{i=0}^{\sqrt{n}} \sum_{j=0}^{\sqrt{n}} a_{ij} x^i y^j.$$

   The number of coefficients in Q is $(\sqrt{n}+1)^2 > n$. Asking that Q vanish at a given point imposes one *homogeneous* linear constraint on the coefficients of Q. We are asking for n vanishings, and, since there are more than n coefficients, such a nonzero Q exists.

2. Now, suppose $P(x)$ is such that $\Delta(r, P) < U$. Define $h(x) = Q(x, P(x))$. Whenever $a \in S$ is such that $r(a) = P(a)$, then $h(a) = Q(a, P(a)) = Q(a, r(a)) = 0$. So, h vanishes at at least $n - U$ points. We observe that $\deg(h) \leq \sqrt{n} + \sqrt{n} \cdot d = \sqrt{n}(d+1)$. If $\deg(h) < n - U$, then $h(x) \equiv 0$. We see that $\sqrt{n}(d+1) < n - U$ if and only if $U < n - (d+1)\sqrt{n}$. So, if $U < n - (d+1)\sqrt{n}$, then every P that we are interested in satisfies $Q(x, P(x)) = 0$. This occurs if and only if $y - P(x)$ divides $Q(x, y)$.

   So, all we need to do is factor $Q(x, y)$ and look for factors of the form $y - P(x)$. We then output all such $P(x)$. The above analysis guarantees that we will find all $P(x)$ such that $\Delta(r, P(x)) \leq U < n - (d+1)\sqrt{n}$. For $d = o(\sqrt{n})$, this decodes from $1 - o(1)$ fraction errors.

For an alternative analysis of Step 2, we could have used Bezout's Theorem, which states that if $A(x, y)$ and $B(x, y)$ are coprime polynomials of degrees E and F, then the total number of intersections between the curves $A(x, y) = 0$ and $B(x, y) = 0$ is at most $E \cdot F$. We see that, for any $P(x)$ we care about, the curves $Q(x, y) = 0$ and $y - P(x) = 0$ both pass through the points we are interested in. Q has degree at most n; $y - P(x)$ has degree at most d. So, if these curves have at least $d\sqrt{n}$ common points, which they do in our case, then $y - P(x)$ and $Q(x, y)$ must have a common factor. But, $y - P(x)$ is irreducible, so it must divide $Q(x, y)$.

Algorithm 2 (Sudan's Algorithm Version 2)

The idea is that we balance X, Y degree so that Y degree is lower and then let Y degree be equal to t.

1. Find $Q(X,Y) \in \mathbb{F}[X,y]$ degrees $n/t, t$ respectively such that $Q(X,Y)$ vanishes on all $(a, r(a))_{a \in S}$. We can do this since the number of coefficients is equal to $(n/t+1)(t+1) > n$.

2. Taken $P(x)$ such that $\Delta(r,\rho) < U$. Let $h(X) = Q(X, P(X))$. And degree of h be $n/t + d.t$ and h has at least $n - U$ roots.

   If $n/t + dt < n - U$ then $h(X) = 0 \Rightarrow Q(X, P(X) = 0 \Rightarrow Y - P(X)$ is a factor of $Q(X,Y)$. If $U < n - n/t - dt$, then we can find $P(X), t = n/d$ and then we can take $U < n - 2\sqrt{nd}$. We can now decode $n - 2\sqrt{nd}$ errors.

   RS decodes in $(1 - (1 - (1 - d/n))^{1/2}n)$ distance decodable. Therefore, list decoding from $n - \sqrt{nd}$ errors gives polynomial size lists. So we can decode from $n - \sqrt{nd}$ errors efficiently for $d << n$, this is basically the same but for $d = \Omega(n)$ there is a difference.

Algorithm 3 (Sudan's Algorithm Version 3)

The idea here is to optimize the monomials in $Q(X,Y)$. Ultimately, we care about the degree of $Q(X, P(X))$. Consider all the monomials with $(1,d)$ degree at most D. $(1,d)$-weighted degree of $X^i Y^j equals i + dj$. So the number of monomials is about $\frac{1}{2} \cdot \frac{d}{D} . D = \frac{D^2}{2d}$. we want $\frac{D^2}{2d} > n, D = \sqrt{2nd}$.

1. Take $Q(X,Y)$ of the form $\sum_{i,j} a_{i,j} X^i Y^j$, where $i + dj \leq D$ non-zero vanishing on all $(X, r(x))_{X \in S}$. We can do this since the number of monomials is greater than $\frac{D^2}{2d} = n$.

2. For any $P(x)$ with $\Delta(r,P) < U, h(X) = Q(X, P(X))$ has $n - U$ roots and $deg(h) \leq D$. If $U < n - D = n - \sqrt{2nd}$ then $Q(X, P(X)) = 0$. Hence the algorithm list-decodes from $n - \sqrt{2nd}$ errors.

Definition (Derivative over finite fields)

–$(X^i)' = iX^{i-1}$

–$(X^i)^j = \frac{i(i-1)(i-2)\ldots(i-j+1).X^{i-j}}{j!} = \binom{i}{j} X^{i-j}$

This extends by linearity and we have an alternative definition:

For $P(x) \in \mathbb{F}[x]$, define $P^{(j)}(X)$ to be the coefficient of Z^j in $P(X+Z)$. Then

$$P(X+Z) = P(X) + P^{(1)}(X)Z + \cdots + P^{(j)}(X)Z^j + \ldots$$

Definition (Multivariate Hasse derivatives)

$P(X+Z_1, Y+Z_2) = \sum_{j_1,j_2} P^{j_1,j_2}(X,Y)(Z_1)^{j_1}(Z_2)^{j_2}$.

$P(X_1, \ldots, X_m)$ vanishes at $(a_1, \ldots a_m)$ with multiplicity M if

$$P^{j_1,j_2,\ldots,j_m}(a_1, \ldots, a_m) = 0 \quad \forall j_1, j_2, \ldots, j_m, \quad \sum_{i=1}^{m} j_i < M.$$

$P(X,Y)$ vanishes at (a,b) if

$$P(a,b) = 0, P^{(1,0)}(a,b) = 0 \ldots P^{(0,1)}(a,b) = 0.$$

Lemma

Suppose $Q(X,Y)$ vanishes at (a,b) with multiplicity M. Suppose $P(X)$ is such that $b = P(a)$. Then $h(x) = Q(X, P(X))$ vanishes at a with multiplicity M.

Algorithm 4 (Guruswami-Sudan Algorithm)

The idea is to add more restrictions on the bi-variate polynomial $Q(X,Y)$ which results in the increment of constraints along with the number of roots.

1. Understand r with high multiplicity.

   In the first step let M be large parameter to be chosen. Find Q(X,Y) with $(1,d)$ with degree at most D that vanishes at each point $x, r(x)$ where $x \in S$ with multiplicity M.

   This can be done as long as the number of free variables is greater than the number of homogeneous linear coefficients. Asking that $Q(X,Y)$ vanishes at a single point with multiplicity M is asking that $Q(a,b) = 0, Q^{(1,0)}(a,b) = 0, \ldots Q^{(j_1,j_2)}(a,b) = 0 \quad \forall j_1, j_2 \quad j_1 + j_2 < M$

   The total number of homogeneous coefficients is equal to $n.(\frac{M(M+1)}{2})$ and the total number of free variables of $(1,d)$ w.t.d. less than D is equal to $\frac{D^2}{2d}$. If $\frac{D^2}{2d} > n.(\frac{M(M+1)}{2})$ then we can find such Q. $D^2 > dnM^2$ and $D > \sqrt{nd}.M$.

2. Use it.

   In step 2, Let $P(X)$ be such that $\Delta(r,p) < U$ look at $h(X) = Q(X, P(X))$ degree of h is at most D. If a is such that $P(a) = r(a)$ then $Q(a, P(a)) = h(a) = Q(a, r(a)) = 0$.

By the lemma, h vanishes at a with high multiplicity. There are $n-U$ such a. If $(n-U)M > D$ then $h(X) \equiv 0$. Therefore, $Q(X, P(X)) = 0$ and $Y - P(X)$ is a factor Q(X,Y). Next, Factor $Q(X, Y)$. Take all $P(X) s.t. Y - P(X) | Q(X, Y)$. It finds all $P(X)$ with $\Delta(r, p) < U < n - \frac{D}{M}$ and $\frac{D}{M}$ can be taken to equal $\sqrt{nd}$.

Note: we needed to take M, D large so that $M\frac{M+1}{2} \sim \frac{M^2}{2}$ is justified.