CS 229r: Essential Coding Theory	Feb 19, 2020
Lecture 7	
Instructor: Madhu Sudan	*Scribe: Gal Kaplun*

1 Concatenated Codes

1.1 Motivation

Recall from last lecture the Reed-Solomon Code:

Proposition 1. *Reed-Solomon Code. Let $k < n \leq q$ with q a prime number. Then, there exists a linear code C (The Reed-Solomon Code) with $[n, k, n-k+1]_q$.*

As we saw in class, for any given n, k this distance is optimal. The main concern about this code is that we require $n \leq q$. We will try to work around this (strong) requirement with the hope of not loosing too much distance in the process. Moreover, for small alphabet size, we saw an existential guarantee about codes on the line $R = 1 - H(\delta)$ (Figure 1) but we never saw a constructive code that achieves this performance.

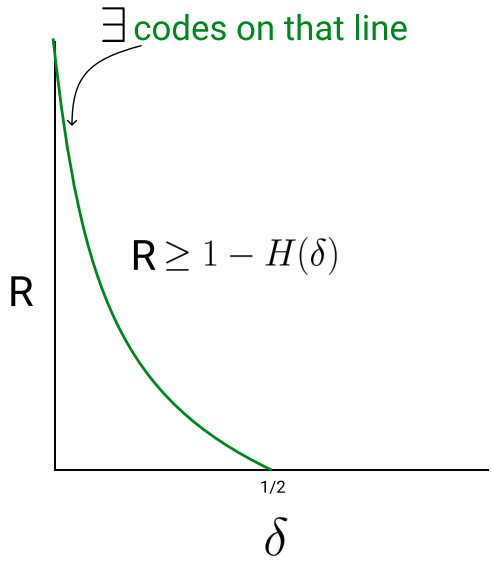

Figure 1: We saw that there are codes on that line.

1.2 Constructability

One caveat with the term "constructive" is that it is not well defined in the sense that we can go over all possible codes and find one that achieves this performance. The problem with that approach is that it is inefficient, i.e., it takes exponential time to compute. Thus, we define two notions of constructability that we would want in order to be able to work with a code.

Definition 2. *Weak constructability. A code is weakly constructible if the encoding function,* $E : \{0,1\}^k \to \{0,1\}^n$ *runs in* $poly(n)$ *time.*

Definition 3. *Strong constructability. A linear code with generator matrix* $G \in \mathbb{F}_q^{k \times n}$ *is strongly constructible if for any given* $(i,j) \in [k] \times [n]$*, outputting* $G_{i,j}$ *runs in* $poly(\log n)$ *time.*

Intuitively, we want the matrix G to be explicit, that is, have it in random access memory. Above, the $\log n$ factor is due to the bit encoding of indices. We would like a code that is both strongly constructible and asymptotically good, e.g. see Figure 2.

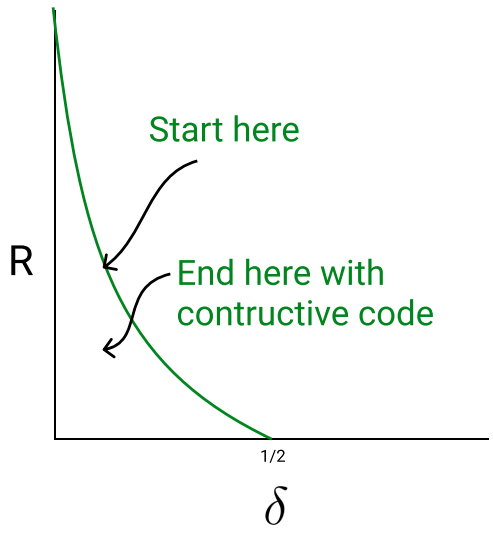

Figure 2: We want to move from existential results to constructive yet asymptotically good codes.

1.3 Concatenation

To achieve both strongly constructible and asymptotically good codes, we will be using concatenated codes. First, we see how to reduce alphabet size.

Proposition 4. *Every code with parameters* $(n,k,d)_q$ *can be converted to* $(n \log_2 q, k \log_2 q, d)_2$ *(or more formally* $(n\lfloor \log_2 q \rfloor, k\lceil \log_2 q \rceil, d)_2$*) code.*

Proof. This is simply done by associating q with $\log_2 q$ bits. Then, $\Sigma \cong \{0,1\}^{\log q}$ and the encoding is from $\{0,1\}^{\log qk} \to \{0,1\}^{\log qn}$. □

Exercise 5. *Verify that the distance of the above code is d.*

In the following, we discuss if there is a better way of encoding q in a larger number of bits, say, $10 \log_2 q$ instead of $\log_2 q$ bits. More specifically, assume that we have a code $(10 \log q, \log q, \log q)_2$ (positive constant rate) and an $(n, k, d)_q$ code (think of Reed-Solomon) code where we think of $n \approx q$, can we combine them into a one good code?

We start with weak constructability. Note that we can be exponential in $\log q$ as this corresponds to being polynomial in n, thus, using the greedy/Varshamov approach we can construct a code with $(10 \log q, \log q, \log q)_2$. Again, an approximately optimal $(n, k, d)_q$ code can be constructed with the Reed-Solomon codes. In Figure 3 we illustrate how the concatenation works. To combine the two we use the following lemma:

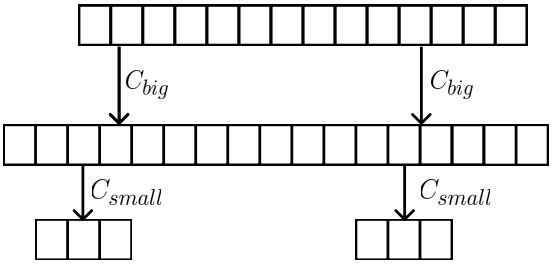

Figure 3: Concatenation of codes. We first encode using C_{big} then we encode the result using C_{small}.

Lemma 6. *Concatenation lemma. Assume we have codes C_{big} that achieves $(N, K, D)_Q$ and C_{small} that is an $(n, k, d)_q$ code where $Q = q^k$. Then, the code of their concatenation C_{concat} achieves $(nN, kK, dD)_q$.*

Exercise 7. *Prove that the distance of the code is dD.*

Exercise 8. *Assuming both C_{big} and C_{small} are linear and achieve $[N, K, D]_Q$ and $[n, k, d]_q$ respectively. Show that C_{concat} is a $[nN, kK, dD]_q$ code.*

For example, if we instantiate the above example with Reed-Solomon: C_{big} that satisfies $(N, \frac{N}{2}, \frac{N}{2})_N$ (rate $\frac{1}{2}$) and Greedy: $C_{small\ greedy}$ that satisfies $(10 \log N, \log N, \log N)_2$, then we have poly time (weakly) constructible code C_{concat} that achieves $(10M, \frac{M}{2}, \frac{M}{2})_2$ where $M = N \log N$. Note that the code is indeed weakly constructible—the time is takes to encode the Reed-Solomon code takes $O(N^3)$ as it is just calculating matrix by vector multiplication with the Vandermonde matrix. Subsequently, the greedy codes runs in exponential time in $\log_2 N$ which translates to polynomial time in N. Since composition of polynomial times is still polynomial we have that the concatenation code is weakly constructible.

We now move on to multivariate polynomials codes (Reed-Muller) as a tool to get strong constructability.

2 Multivariate Polynomials (Reed-Muller) Codes

2.1 Reed-Muller codes

We will work with the following parameters: the field $\mathbb{F}_q$, the number of variables M and the degree r. The messages space is $\{c_{\bar{e}}|\bar{e} = (e_1, ..., e_m), \sum e_i \leq r, 0 \leq e_i < q\}$ and each message is associated with a polynomial $p(x_1, ..., x_m) = \sum c_{\bar{e}} x_1^{e_1} ... x_m^{e_m}$. Also, the encoding is done by evaluating polynomials on the elements of the field $\{p(\alpha_1, ..., \alpha_m)|(\alpha_1, ..., \alpha_m) \in \mathbb{F}_q^m\}$. In order to evaluate the distance of this encoding we will use the following Lemma:

Lemma 9. *Schwartz–Zippel Lemma. if $0 \neq p \in \mathbb{F}[x_1, ..., x_m]$ has degree $\leq r$, then for every set $S \subseteq \mathbb{F}$,*

$$\frac{|\{\alpha_1, ..., \alpha_m) \in S^m : \ p(\alpha_1, ..., \alpha_m) = 0\}|}{|S|^m} \leq \frac{r}{|S|} \tag{1}$$

Note the lack of dependence on m that will be useful to us. The idea behind the proof is to look at the polynomial as a univariate polynomial $p_{\alpha_1,...,\alpha_{m-1}}(x_m)$ and separating between two cases: 1) $p_{\alpha_1,...,\alpha_{m-1}}(x_m)$ is 0 everywhere. 2) $p_{\alpha_1,...,\alpha_{m-1}}(x_m)$ is the non-zero polynomial. If we randomize over $\alpha_1, ..., \alpha_{m-1}$, the first case only appears w.p at most $\frac{r-t}{|S|}$ where t is the degree of the univariate polynomial (inductive hypothesis). On the other hand, there are at most t (the degree) solutions for the second case for each such polynomial.

Proof sketch. We rewrite

$$p(x_1, ..., x_m) = p_0(x_1, ..., x_{m-1}) + p_1(x_1, ..., x_{m-1})x_m + ... + p_t(x_1, ..., x_{m-1})x_m^t$$

s.t. $p_t \neq 0$ and $deg(p_t) \leq r - t$. We then have (inductive hypothesis):

$$\mathbb{P}_{\alpha_1,...,\alpha_m}[p_t = 0] \leq \frac{r-t}{|S|}$$

and

$$\mathbb{P}_{\alpha_1,...,\alpha_m}[p_t(\alpha_1, ..., \alpha_{m-1})\alpha_m^t + ... + p_0(\alpha_1, ..., \alpha_{m-1})|p_t \neq 0] \leq \frac{t}{|S|}$$

and using induction we can conclude the proof.

2.2 Justesen codes

We move on to Justesen codes that will also utilize the above construction. Here we again assume a large alphabet size, i.e., $n = q = 2^t$. The messages are in $\mathbb{F}_q^{\frac{n}{2}} \cong \mathbb{F}_2^{\frac{nt}{2}}$, and $M(x) \in \mathbb{F}_2^{<\frac{n}{2}}[x]$ are polynomials with degree $< \frac{n}{2}$. We encode a message in the following way:

$$(M(\alpha_1), \alpha_1 M(\alpha_1), M(\alpha_2), \alpha_2 M(\alpha_2), ..., M(\alpha_2), \alpha_q M(\alpha_q))$$

where $\mathbb{F}_q = \{\alpha_1, ..., \alpha_q\}$. We will view this as a long encoding over bits $\in \mathbb{F}_2^{2tq}$. The code is $[2tq, \frac{tq}{2}, \frac{q}{2}]_2$ as there are at most $\frac{q}{2}$ roots to the polynomial. This encoding scheme can be viewed

as "concatenation" of two encoding schemes but the second encoding is not a unique encoding but rather an ensemble of codes. Most of those codes are good, and if we assume that all of them are good then we have a good fully constructible code: The inner code is $([2t, t, 1 - H^{-1}(\frac{1}{2}) \cdot 2t]$. So we have a $[2tq, \frac{tq}{2}, 1 - H^{-1}(\frac{1}{2}) \cdot 2tq]_2$ code. Thus, the ratio is $R = \frac{1}{4}$ and the distance is $\delta > 0$. Note that for $m = 1$ we have the Reed-Solomon codes and for arbitrary $m, q = 2, r = 1$ we have $[2^m, m + 1, 2^{m-1}]_2$ code.

Exercise 10. *Contrast this result with the Plotkin bound and analyze the running time of access of each coordiante.*