Notes for Lecture 14

In this lecture we will show how to concatenate the Hadamard and Reed-Solomon codes to obtain a code where the number of corrupted bits can get arbitrarily close to $\frac{1}{2}$. We also present Reed-Muller codes together with a sublinear-time unique decoding algorithm.

1 Concatenation of Reed-Solomon and Hadamard codes

Let us consider a Reed-Solomon code on the field $\mathbb{F}$:

$$\mathbf{RS} : \mathbb{F}^k \to \mathbb{F}^n, \quad n \leq |\mathbb{F}|$$

By Sudan's algorithm (see previous lecture), given a corrupted encoding with $\geq 2\sqrt{kn} + 1$ non-errors, we can reconstruct in polynomial time the list of all codewords that agree with the given input in at least $\geq 2\sqrt{kn} + 1$ positions.

Now consider a Hadamard code

$$\mathbf{H} : \{0,1\}^k \to \{0,1\}^{2^k}.$$

By the Goldreich-Levin algorithm, given a corrupted encoding with $\geq (\frac{1}{2} + \epsilon)2^k$ non-errors, in time poly$(k, \frac{1}{\epsilon})$ we can reconstruct all messages whose encoding has agreement $\geq (\frac{1}{2} + \epsilon)2^k$ with the input. From the analysis of the algorithm it also follows that given $y \in \{0,1\}^{2^k}$, there are at most $O(\frac{1}{\epsilon^2})$ codewords that agree in $\geq (\frac{1}{2} + \epsilon)2^k$ bits with y. By using Fourier analysis we can get $\frac{1}{4\epsilon^2}$ codewords.

We want to produce a code such that if the proportion of errors in the output is less but arbitrarily close to $\frac{1}{2}$, then we can find in polynomial time all the codewords that are close to the output.

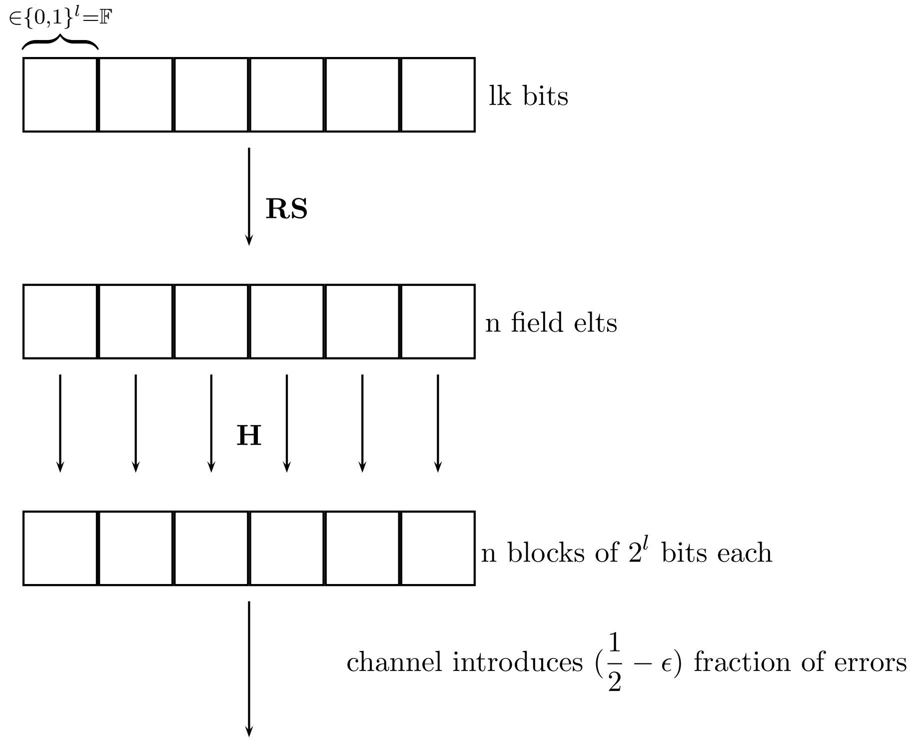
Now suppose $n = |\mathbb{F}| = 2^l$. As before we have

$$\mathbf{RS} : \mathbb{F}^k \to \mathbb{F}^n \; \mathbf{H} : \{0,1\}^k \to \{0,1\}^{2^k},$$

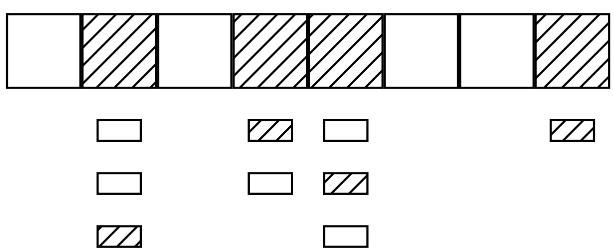
which gives

$$\mathbf{RS} \circ \mathbf{H} : \{0,1\}^{lk} \to \{0,1\}^{n2^l}.$$

If $\geq (\frac{1}{2} + \epsilon)n2^l$ of the bits in the output are correct, then an easy calculation shows there exist $n\epsilon/2$ blocks in which at least $\geq (\frac{1}{2} + \frac{1}{\epsilon})n$ bits are correct. We now apply the Hadamard list decoding algorithm with radius $(\frac{1}{2} - \frac{\epsilon}{2})n$ to each block individually. By a previous argument, there are at most ϵ^2 codewords in each list.



Now pick a random element from each list and construct a new binary string. For at least $\frac{\epsilon}{2}n$ of the blocks, the correct field codeword is contained in its list and there are at most $1/\epsilon^2$ elements in each list, therefore this random assignment will, on average, correctly decode at least $\frac{\epsilon^3}{2}$ of the blocks.



Think of the blocks as elements of $\mathbb{F}$. We have an **RS** encoding where the proportion of non-errors is at least $\frac{\epsilon^3}{2}$. If $n\frac{\epsilon^3}{2} > 2\sqrt{nk}$, then by Sudan's algorithm we are done. But

$$n\frac{\epsilon^3}{2} > 2\sqrt{nk} \Leftrightarrow n \geq \frac{16k}{\epsilon^6}.$$

Since $l = \log n$, we get $l = \log\left(\frac{16k}{\epsilon^6}\right)$ and our encoding becomes

$$\mathbf{RS} \circ \mathbf{H} : \{0,1\}^{k\log\left(\frac{16k}{\epsilon^6}\right)} \to \{0,1\}^{\frac{256k^2}{\epsilon^{12}}}.$$

The following theorem is therefore true:

Theorem 1 *For any k, ϵ, there is a code $\boldsymbol{C} : \{0,1\}^k \leftarrow \{0,1\}^n$, where $n = poly(k, \frac{1}{\epsilon})$, computable in polynomial time, such that given $y \in \{0,1\}^n$, we can find in time polynomial in $(k, \frac{1}{\epsilon})$ a list of size $poly(\frac{1}{\epsilon})$ that contains all codewords with agreement $\geq (\frac{1}{2} + \epsilon)n$ with y.*

2 Reed-Müller codes

Reed-Müller codes are an encoding of the type

$$\mathbf{RM} : \mathbb{F}^{h^m} \to \mathbb{F}^{q^m}.$$

Fix a subset $H \subseteq \mathbb{F}$, such that $|H| = h$. Given a message M of length h^m, we think of M as the list of values of a function

$$M : H^m \to \mathbb{F}.$$

Claim 2 *We can always find a polynomial $P_M : \mathbb{F}^m \to \mathbb{F}$ which has degree $\leq h$ in each variable such that*

$$P_M(x) = M(x), \forall x \in H^m.$$

Proof: This can be done by using the standard Lagrange inversion formula and induction on m. □ The encoding of M is then the list of values of $P_M(\cdot)$ at all points in $\mathbb{F}^m$.

Now suppose we have two different messages M and M'. Then their encodings correspond to two different polynomials and the distance between the two codewords would be

$$\text{length of encoding} \cdot \Pr_{x \in \mathbb{F}^m}[P_M(x) \neq P_{M'}(x)] \geq |\mathbb{F}|^m \left(1 - \frac{hm}{|\mathbb{F}|}\right).$$

This is an easy consequence of the following theorem:

Theorem 3 (Schwartz-Ziepel) *If $p : \mathbb{F}^m \to \mathbb{F}$ is a non-zero degree d polynomial, then $\mathbf{Pr}_{x \in \mathbb{F}^m}[p(x) = 0] \leq \frac{d}{|\mathbb{F}|}$.*

We therefore need $|\mathbb{F}| \geq 2hm$ to get an encoding with relative distance $\frac{1}{2}$, in which case we will transform strings of length $k = h^m$ into strings of length $h^m(2m)^m = k(2m)^m$.

When m is large, take $h = k^{1/m}$. In this case, the encoding becomes more wasteful, but the efficiency actually increases, as the decoding running-time depends only on h.

Now let us describe the decoding procedure. Let $P_M : \mathbb{F}^m \to \mathbb{F}$ be the degree $d = hm$ encoding of the message $M : H^m \to \mathbb{F}$, with $|\mathbb{F}| \geq 5d = 5hm$. Suppose that the output $f : \mathbb{F}^m \to \mathbb{F}$ differs from P_M in at most $1/10$ of the total number $|\mathbb{F}^m|$ of entries. Given $x \in H^m$, we need to compute $p(x)$.

We use the following algorithm:

RM $-$decode(x)

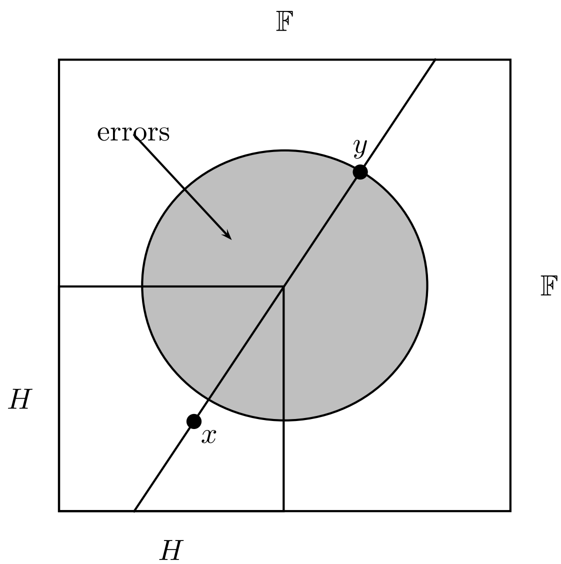
Choose uniformly at random $y \in \mathbb{F}^m$

Take the line $l(t) = ty + (1-t)x$

Let $F(t)$ be the result of the unique decoding of Reed-Solomon codes algorithm applied to $f(l(t))$ as a function of t

Return $f(0)$.

It is easy to see that since P_M is a polynomial in x, $P_M(l(t))$ is a composition of two polynomials, and therefore a polynomial in t, of the same degree d. Let $P_M(l(t)) = p(t)$. We have $P_M(x) = P_M(l(0)) = p(0)$. Therefore recovering p is enough for recovering $M(x)$.



Now if y is uniformly distributed, then $a \cdot y$ is uniformly distributed for any constant a, so $ty+(1-t)x$ is uniformly distributed for any fixed value of t (remember we are choosing y uniformly at random). Therefore with probability ≥ 0.9, $l(t)$ is correct (i.e.$f(l(t)) = p(t)$) for any fixed value of t. Also, on average 0.9 of the points on $l(t)$ are correct.

By Markov's inequality, $\mathbf{Pr}[|\{t|p(t) = f(l(t))\}| \geq 0.7|\mathbb{F}|] \geq 2/3$. Using the decoding algorithm of Reed-Solomon from 2 lectures ago, we can find the unique polynomial $p(t)$ which agrees with $f(l(t)$ in at least 0.6 of the positions. We can do this since $d \leq 0.2|\mathbb{F}|$, and thus $0.4|\mathbb{F}| \leq (|\mathbb{F}| - d)/2.\square$

Note: It is possible to get the probability of error arbitrarily close to 1 by a method similar to that of the Goldreich-Levin algorithm.

$$P_M : \mathbb{F}^m \to \mathbb{F} \qquad M : H^m \to \mathbb{F} \qquad h = |H|$$

Then P_M is of degree $d = hm$ and $f : \mathbb{F}^m \to \mathbb{F}$ differs from P_M for at most $0.9|\mathbb{F}|^m$ inputs. Now look at the line through x and y, where x and y are chosen uniformly at random. Apply Sudan's list-decoding algorithm to find all polynomials of degree at most d that agree with f on the line in at least 0.05 of the points. If the list does not contain a unique polynomial q with $q(0) = f(x)$, then return an error. Otherwise output $P_M(y) = q(1)$.