Introduction to Coding Theory **CMU: Spring 2010**

Notes 10: List Decoding Reed-Solomon Codes and Concatenated codes

April 2010

Lecturer: Venkatesan Guruswami *Scribe: Venkat Guruswami & Ali Kemal Sinop*

DRAFT

Last class, we saw a toy version of recovering from a mixture of two Reed-Solomon codewords the two polynomials in question. Now we turn to list decoding arbitrary received words with a bounded distance from the Reed-Solomon code using Sudan's [6] algorithm. This algorithm decodes close to a fraction 1 of errors for low rates. Then we will see an improvement by Guruswami and Sudan [4] which list decoded up to the Johnson radius.

1 List Decoding Reed-Solomon Codes

Let C_{RS} be a $[n, k+1, n-k]_q$ Reed-Solomon code over a field $\mathbb{F}$ of size $q \geq n$ with a degree k polynomial $p(X) \in \mathbb{F}[X]$ being encoded as $(p(\alpha_1), p(\alpha_2), \ldots, p(\alpha_n))$. We can reduce the list-decoding problem to a *polynomial reconstruction* problem with agreement parameter t.

Remark 1 *If $t > \frac{n+k}{2}$, then the answer (if it exists) is unique.*

Remark 2 *Recall that Johnson Bound states that a code of distance $k+1$ can be list decoded up to $n - \sqrt{nk}$ errors. Our goal is to try solve this for $t > \sqrt{kn}$.*

However it is an open question whether if one can do it for (say) $t = 0.9\sqrt{kn}$. Another open question is what can be done if we allow super polynomial list size for $t = \frac{n}{k}$.

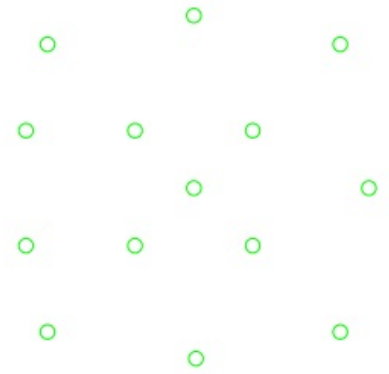
1.1 Polynomial Reconstruction

Given parameters n,t,k and n pairs $(\alpha_i, y_i) \in \mathbb{F}^2$, we want to find all degree $\leq k$ polynomials $f \in F[X]$ such that $\left|\left\{i \middle| f(\alpha_i) = y_i\right\}\right| \geq t$. Here we will present an algorithm due to Sudan [6] for $t \geq \sqrt{2kn}$. Later we will see how to remove $\sqrt{2}$.

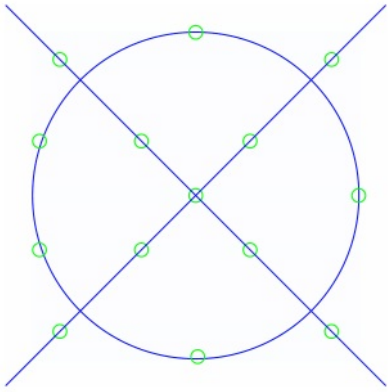
Figure for comparison of relative distances.

1.1.1 Geometric Intuition

Example 1 *Consider the following point configuration on which we want to find all the lines passing through at least 5 points.*



By inspecting the figure, we can see that there are only two such lines, $f_1(x) = x$ and $f_2(x) = -x$:



Claim 3 *Any total degree 4 polynomial $Q(x, y)$ such that $Q(\alpha_i, y_i) = 0$ for all i, must have these two lines as factors.*

PROOF: *Both lines must intersect $Q(x, y) = 0$ in at least 5 distinct points. Let $Q(\alpha_j, f_i(\alpha_j)) = 0$ for 5 different points. Since $Q(x, f_i(x))$ is a degree-4 polynomial in x, we have $Q(x, f_i(x)) \equiv 0$. Therefore $y - f_i(x) \mid Q(x, y)$.* $\square$

Hence a possible factorization for these points is given by $Q(x, y) = (x - y)(x + y)(x^2 + y^2 - 1)$.

1.1.2 Sudan's Algorithm

Armed with the above intuition, consider the following algorithm:

1. Find a low degree $Q(x, y) \not\equiv 0 \in \mathbb{F}[X, Y]$ such that $Q(\alpha_i, y_i) = 0$ for all i.
2. Find all factors of the form $y - f(x)$ of $Q(x, y)$. Output those with agreement at least t.

There are three questions we need to answer in order to show that this algorithm is correct. First we need to make sure that such Q exists. Second we need to find out when $y - f(x)$ is a factor of $Q(x, y)$. Third is how we can find such factors efficiently. We will touch upon a randomized algorithm for this at the end of this section.

Lemma 4 *Given positive integers d_x, d_y, $n < (d_x + 1)(d_y + 1)$ and n points $\{(\alpha_i, y_i)\}_{i=1}^n \subset \mathbb{F}^2$, the following holds: There exists $Q(X, Y) \not\equiv 0 \in \mathbb{F}[X, Y]$ such that $Q(\alpha_i, y_i) = 0$ for all i and $\deg_x(Q) \leq d_x, \deg_y(Q) \leq d_y$. Moreover such Q can be found by solving a linear system.*

PROOF: Notice that Q with $\deg_x(Q) \leq d_x, \deg_y(Q) \leq d_y$ has $(d_x + 1)(d_y + 1)$ many coefficients. For each i, $Q(\alpha_i, y_i) = 0$ is a homogeneous linear equation in terms of Q's coefficients. There are n such equations and $(d_x + 1)(d_y + 1)$ many unknowns. For $(d_x + 1)(d_y + 1) > n$, a non-zero solution is guaranteed to exist. □

Having made sure that Q exists, we need to answer when $y - f(x) \mid Q$.

Lemma 5 *Given a polynomial $f(x) \in \mathbb{F}[X]$ with $f(\alpha_i) = y_i$ for t different points $(\alpha_i, y_i) \in \mathbb{F}^2$: $Q(\alpha_i, y_i) = 0$, if $t > \deg_x(Q) + \underbrace{\deg_x(f)}_{k} \deg_y(Q)$, then $y - f(x) \mid Q(x, y)$.*

PROOF: Let $R(x) := Q(x, f(x))$. We have $R(\alpha_i) = Q(\alpha_i, f(\alpha_i)) = Q(\alpha_i, y_i) = 0$. Therefore $R(x)$ has t different roots. But $R(x)$ has degree at most $\deg_x(Q) + \deg_y(Q) \deg_x(f) < t$, which implies $R(x) \equiv 0 \implies Q(x, f(x)) \equiv 0$. Hence $y - f(x) \mid Q(x, y)$. □

Remember, $Q(x, y)$ can have at most $\deg_y(Q)$ many different factors of the form $y - f(x)$. Hence $\deg_y(Q) = \ell$ is our list size. By Lemma 4, we can take $d_x = \lfloor \frac{n}{\ell} \rfloor$. By Lemma 5, we need $t > \frac{n}{\ell} + \ell k$. This is minimized for $\ell = \sqrt{\frac{n}{k}}$, which yields $t > 2\sqrt{nk}$.

In order to get rid of a factor of $\sqrt{2}$, we notice that the important quantity is $d_x + k \cdot d_y$. As long as all non-zero monomials $q_{ij} x^i y^j$ of $Q(x, y)$ has $i + kj < t$, Lemma 5 will hold.

Definition 6 ($(1, k)$-weighted degree) *For a polynomial $Q(X, Y) \in \mathbb{F}[X, Y]$, its $(1, k)$-weighted degree is defined to be the maximum value of $i + kj$ over all non-zero monomials $x^i y^j$ of Q.*

Lemma 7 *Given positive integer D and $kn < \binom{D+2}{2}$ points $\{(\alpha_i, y_i)\}_{i=1}^n \subset \mathbb{F}^2$, the following holds: There exists $Q(X, Y) \not\equiv 0 \in \mathbb{F}[X, Y]$ such that $Q(\alpha_i, y_i) = 0$ for all i and $(1, k)$-weighted degree of Q is D. Moreover such Q can be found by solving a linear system.*

PROOF: The number of Q's coefficients is:

$$\sum_{j=0}^{\lfloor D/k \rfloor} (D - jk + 1) = (D + 1) \left(\left\lfloor \frac{D}{k} \right\rfloor + 1 \right) - \frac{k}{2} \left\lfloor \frac{D}{k} \right\rfloor \left(\left\lfloor \frac{D}{k} \right\rfloor + 1 \right) \geq \frac{(D + 1)(D + 2)}{2k}$$

Rest of this argument is exactly the same with Lemma 4. □

Lemma 8 *Given a degree-k polynomial $f(x) \in \mathbb{F}[X]$ with $f(\alpha_i) = y_i$ for t different points $(\alpha_i, y_i) \in \mathbb{F}^2 : Q(\alpha_i, y_i) = 0$, if $(1,k)$-weighted degree of Q is $< t$, then $y - f(x) \mid Q(x,y)$.*

PROOF: As usual, let $R(x) := Q(x, f(x))$ and note that $R(x)$ has t different roots. Now degree of R is at most $\max_{ij:q_{ij}\neq 0} i + \deg_x(f)j \leq \max_{ij:q_{ij}\neq 0} i + kj < t$. Here q_{ij} is the coefficient of monomial $x^i y^j$ of $Q(x,y)$. Hence $y - f(x) \mid Q(x,y)$. $\square$

Putting these together, we obtain:

Theorem 9 (6) *Given parameters n,t,k and n pairs $(\alpha_i, y_i) \in \mathbb{F}^2$, if $t > \sqrt{2kn}$, we can find all degree $\leq k$ polynomials $f \in F[X]$ such that $\left|\{i | f(\alpha_i) = y_i\}\right| \geq t$ in time polynomial in n, k and $|\mathbb{F}|$. Moreover there are at most $\sqrt{2n/k}$ many such f's.*

PROOF: Let $D \leftarrow t - 1$. Then $\frac{(D+1)(D+2)}{2k} = \frac{t^2+t}{2k} > \frac{2nk+1}{2k} > n$. By Lemma **??**, a Q with $(1,k)$-weighted degree $\leq t-1$ exists. By Lemma 8, we know that if f has $\geq t$-agreement, $y - f(x) \mid Q(x,y)$. Assuming that we can find such factors in polynomial time, we are done. $\square$

1.1.3 Finding factors of $Q(X,Y)$ of the form $Y - f(X)$

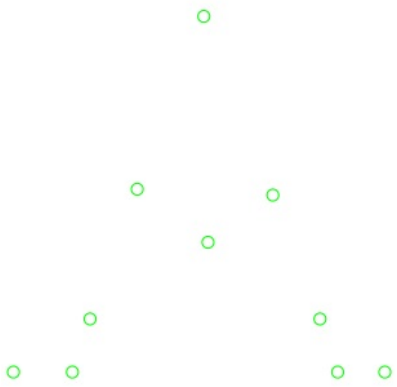
We will describe a simple randomized algorithm given in [1, Section 4.2]. We are given a bivariate polynomial $Q(X,Y) \in \mathbb{F}[X,Y]$ with $q = |\mathbb{F}|$ and we want to find factors $y - f(x)$ with $\deg_x(f) \leq k$. If we can find an algorithm that either finds such polynomial $f(x)$ or concludes that none exists, we are done. We can run this algorithm, and if it finds $f(x)$, we can recurse on the quotient $Q(x,y)/(y - f(x))$.

Let $E(X) \in \mathbb{F}[X]$ be an irreducible polynomial with degree $\geq k+1$. An explicit choice is $E(x) = x^{q-1} - \gamma$ where γ is a generator of $\mathbb{F}$. As we have seen in Homework 2, this is irreducible. Given such $E(x)$, we can view $Q(x,y) \mod E(x)$ as a polynomial, $\widetilde{Q}(Y) \in \widetilde{\mathbb{F}}[Y]$ with coefficients in the extension field $\widetilde{\mathbb{F}} = \mathbb{F}[X]/(E(X))$ of degree $q-1$ over $\mathbb{F}$. The problem reduces to finding roots of $\widetilde{Q}(Y)$ in $\widetilde{\mathbb{F}}$, which can be done using Berlekamp's algorithm in time polynomial in $\deg\left(\widetilde{Q}\right)$ and q.

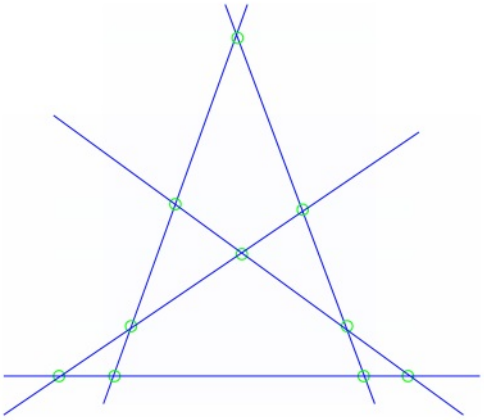
2 Method of Multiplicities

In this section, we will remove the factor $\sqrt{2}$ and obtain a list decoding algorithm with $t > \sqrt{kn}$ due to Guruswami and Sudan [4]. First we will see an example showing that we can't hope to improve the above parameters.

Example 2 *Consider the following point configuration of $n = 10$ points. For $k = 1$ (lines), having $t \geq \sqrt{nk}$ implies $t > 3$. So we want to find lines passing through ≥ 4 points.*



Next figure shows set of all lines going through at least 4 points:



If we want to output all these 5 lines, Q must have total degree at least 5. But the agreement parameter $t = 4$ is smaller than 5, so $Q(x, f(x))$ (for one of these lines; $y = f(x)$) might not be $\equiv 0$.

Note that in this example, each point has two lines crossing. Thus any $Q(x, y)$ containing these lines as factors must also contain each point twice.

Now we will try to understand what it means for $Q(x, y)$ to contain each point twice.

Example 3 *If $Q(X, Y) \in \mathbb{F}[X, Y]$ has r zeroes at point $(0, 0)$, then it has no monomials of total weight less than r.*

Since we can translate Q by any $(\alpha, \beta) \in \mathbb{F}^2$, we can generalize the above example to a formal definition:

Definition 10 (Multiplicity of zeroes) *A polynomial $Q(X, Y)$ is said to have a zero of multiplicity $w \geq 1$ at a point $(\alpha, \beta) \in \mathbb{F}^2$ if $Q_{\alpha,\beta} := Q(x + \alpha, y + \beta)$ has no monomial of total degree less than w.*

Armed with this definition and an idea on how to put additional constraints on Q, we will look at what happens to Lemmas 7 and 8.

Corollary 11 *Given positive integers $D, w_1, \ldots, w_n$ such that $k\sum_i \binom{w_i+1}{2} < \binom{D+2}{2}$, and points $\{(\alpha_i, y_i)\}_{i=1}^n \subset \mathbb{F}^2$, the following holds: There exists $Q(X,Y) \not\equiv 0 \in \mathbb{F}[X,Y]$ such that $Q(x,y)$ has a zero of multiplicity w_i at (α_i, y_i) for all i, and $(1,k)$-weighted degree of Q is D. Moreover such Q can be found by solving a linear system.*

Proof: Notice that coefficient of $x^i y^j$ in $Q_{\alpha,\beta}$ is

$$\sum_{i' \geq i, j' \geq j} \binom{i'}{i}\binom{j'}{j}\alpha^{i'-i}\beta^{j'-j} q_{i',j'}.$$

Hence we can express w_i-multiplicity condition as $\binom{w_i+1}{2}$ homogeneous linear equations. By previous calculation, we know that the number of Q's coefficients is $\geq \frac{(D+1)(D+2)}{2k}$. Thus a non-zero solution exists if $n\sum_i \binom{w_i+1}{2} \leq \binom{D+2}{2}$ $\square$

Lemma 12 *Suppose $f(\alpha) = \beta$ if $Q(x,y)$ has r-zeroes at (α, β). Then $(x-\alpha)^r \mid Q(x, f(x))$.*

Proof: We have

$$R(x) := Q(x, f(x)) = Q_{\alpha,\beta}(x-\alpha, f(x)-\beta) = Q_{\alpha,\beta}(x-\alpha, f(x)-f(\alpha))$$

Here $Q_{\alpha,\beta}$ has no monomials of degree $< r$. Also $(x-\alpha) \mid (f(x)-f(\alpha))$. For any non-zero monomial $x^i y^j$ of $Q_{\alpha,\beta}$, we have $(x-\alpha)^{i+j} \mid (x-\alpha)^i (f(x)-f(\alpha))^j$. Since $i+j \geq r$, we have $(x-\alpha)^r \mid Q_{\alpha,\beta}(x-\alpha, f(x)-\beta) = Q(x, f(x))$. $\square$

Lemma 13 *If a degree k polynomial f has $f(\alpha_i) = y_i$ for $\geq t$ values of i and $\sum_{i:f(\alpha_i)=y_i} w_i > D$, then $y - f(x) \mid Q(x,y)$ assuming Q has $(1,k)$ weighted degree $\leq D$.*

Proof: If $f(\alpha_i) = y_i$ and $Q(x,y)$ had a multiplicity of w_i zeroes at (α_i, y_i) then $(x-\alpha_i)^{w_i} \mid Q(x, f(x))$ by previous lemma. Therefore $\prod_{i:f(\alpha_i)=y_i}(x-\alpha_i)^{w_i} \mid Q(x, f(x))$. If $(1,k)$ weighted degree of $Q(x,y)$ is D, then $Q(x, f(x)) \equiv 0$ and $y - f(x) \mid Q(x,y)$. $\square$

Putting all together, we obtain:

Theorem 14 *The algorithm finds all polynomials $f(x)$ such that $\sum_{i:f(\alpha_i)=y_i} w_i > \sqrt{2k\sum_i \binom{w_i+1}{2}}$. In particular, if $w_i = w$, then we can decode f up to $n-t$ errors with $t > \sqrt{nk\left(1+\frac{1}{w}\right)}$.*

Corollary 15 (4) *For a Reed-Solomon code of rate R, we can list decode up to $1-\sqrt{R}$ fraction of errors in time polynomial in the block length and the field size.*

3 Soft Decoding

One reason we picked each w_i individually is because we "encode" likelihood information. One might think of w_i's representing a confidence value on the received symbol and placing more emphasis on smaller values. This connection has been made formal by Koetter-Vardy [5], where it was shown how to choose weights optimally based on channel observations and channel transition probabilities.

One important point is that this application requires dealing with non-negative real weights. This is made precise in the following Theorem which appears in [2, Chapter 6]:

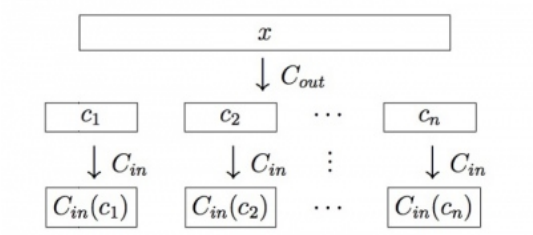
Theorem 16 *For all non-negative rationals* $\{w_{i,\alpha}\}_{i\in[n],\alpha\in\mathbb{F}}$ *and non-negative real* ϵ*, there is a* poly $\left(n, |\mathbb{F}|, \frac{1}{\epsilon}\right)$*-time algorithm to find all degree* k *polynomials* f *such that*

$$\sum_{i=1}^{n} w_{i,f(\alpha_i)} \geq \sqrt{k \sum_{i,\alpha} w_{i,\alpha}^2} + \epsilon w_{\max}.$$

4 List Decoding of Binary Concatenated Codes

The main shortcoming of the above methods is that the alphabet size is polynomial in block length. We will use concatenated codes to reduce alphabet size.

Assume we are given a Reed-Solomon code as our outer code $C_{\text{out}} : \mathbb{F}_2^n \to \mathbb{F}_2^{n/R}$ with rate R and an inner code $C_{\text{in}} : \mathbb{F}_2^m \to \mathbb{F}_2^{m/r}$ with rate $r = \text{rate}(C_{\text{in}})$. Consider the basic composition:



This code has rate $R_0 = R/r$. First we will start with the most naive idea and work our way up. Our main goal in this section is to get a binary code which is close to Zyablov's bound.

4.1 Uniquely Decoding Inner Codes

Consider uniquely decoding each inner codes and then applying list decoding algorithm on the outer code. There will be $\geq \frac{\text{h}^{-1}(1-r)}{2}$ fraction of errors if an inner block is decoded to a wrong code word. The list decoding capacity of C_{out} is $1 - \sqrt{\frac{R}{r}} = 1 - \sqrt{R_0}$. Hence this algorithm can list decode up to $\left(1 - \sqrt{R_0}\right) \frac{\text{h}^{-1}(1-r)}{2}$-fraction of errors.

However this is quite bad, as we picked up a factor of $\frac{1}{2}$ that we were trying to avoid by using list decoding at the first place.

Lemma 17 *Given positive reals $0 < r, R < 1$, there exists a binary code which can be list-decoded up to $\left(1 - \sqrt{\frac{R}{r}}\right) \frac{\mathrm{h}^{-1}(1-r)}{2}$ fraction of errors in polynomial time.*

4.2 List Decoding Inner Codes

Instead of uniquely decoding inner codes, consider list decoding them, by –say– brute force. Recall that by Johnson Bound, any binary code can be list-decoded up to $\mathrm{h}^{-1}(1-r-\epsilon)$-fraction of errors with a list size of $\ell = \ell(\epsilon) \leq O(1/\epsilon)$.

A slight complication arises on how to apply Reed-Solomon list-decoding on a message where each symbol is itself another list. During list-decoding algorithm, we only assumed all (α_i, y_i) pairs were distinct, and everything worked fine even if $\alpha_i = \alpha_j$ for different i, j as long as $y_i \neq y_j$. Using this observation, we can apply our list decoding algorithm on the set $\{(\alpha_i, y_{ij})\}_{i,j}$. This will return us a list of messages which agrees with $t > \sqrt{kN}$ points where $N < n\ell$. Therefore this method will give us list decoding up to $\left(1 - \sqrt{\frac{R}{r}}\right) \mathrm{h}^{-1}(1-r-\epsilon)$ fraction of errors.

Lemma 18 *Given positive reals $0 < r, R < 1$ and for any real $0 < \epsilon < r$, there exists a binary linear code which can be list-decoded up to $\left(1 - \sqrt{\frac{R}{r}}\right) \mathrm{h}^{-1}(1-r-\epsilon)$ fraction of errors.*

In order to get a binary code list decodable up to $\frac{1-\gamma}{2}$ fraction of errors, we can take $\ell = 1/\gamma^2$ and $r = O(\gamma^2)$, which implies $\epsilon = O(\gamma^2)$. By taking $R = r^5$, we obtain:

Theorem 19 *For any real $0 < \gamma < 1$, there exists a linear binary code of rate $\Omega(\gamma^6)$ list-decodable up to $\frac{1-\gamma}{2}$ fraction of errors with a list size of $O\left(\frac{1}{\gamma^3}\right)$.*

Remark 20 *Compare this to the optimal parameter of rate $\Omega(\gamma^2)$ and list size $O(1/\gamma^2)$.*

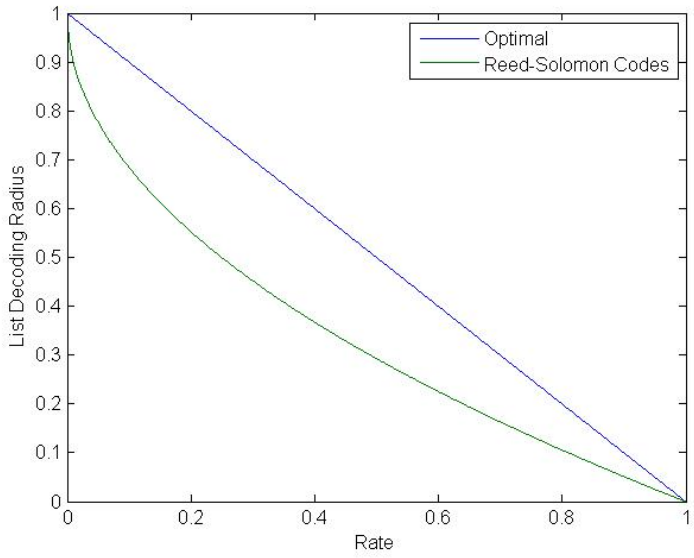
4.3 Weighted List Decoding Inner Codes

One room for improvement to the previous algorithm is that, instead of treating each code in the list equally, assigning different weights. If we remember Theorem 16, we can obtain a list decoding with a (weighted) agreement of $\sum_{i=1}^{n} w_{i,f(\alpha_i)} \geq \sqrt{k \sum_{i,\alpha} w_{i,\alpha}^2} + \epsilon w_{\max}$. In order to minimize this, we need to ensure that the weights of inner codes around a point should decay in ℓ_2 norm rapidly. Although it is not known how to get this for general binary codes, Hadamard codes have this property.

5 Going beyond Reed-Solomon codes & Johnson radius

For Reed-Solomon codes, we showed that one can efficiently list decode up to a radius (fraction of errors) equal to $1 - \sqrt{R}$. However we know that list decoding up to a fraction $1 - R - \epsilon$ of errors

is possible non-constructively. In this section, we will construct better variants of Reed-Solomon codes. The code we will describe was given by Guruswami and Rudra [3].



The main difficulty in Reed-Solomon codes is that we need to be able to deal with *any* pattern of e errors. Instead, we will "pack" parts of codeword together and decrease the error patterns we have to handle considerably.

5.1 Folded Reed-Solomon Codes

Recall that a Reed-Solomon code $\mathrm{RS}_{\mathbb{F},\mathbb{F}^*}[n,k]$ gives an encoding of a degree-k polynomial $f(X) \in \mathbb{F}[X]$ as

$$\begin{aligned} f(x) &\mapsto \left(f(\alpha)\big|_{\alpha \in \mathbb{F}^*} \right) \\ &\mapsto \left(f(1), f(\gamma), \ldots, f(\gamma^{n-1}) \right) \end{aligned}$$

where $n = |\mathbb{F}| - 1 = q - 1$ and γ is a generator of $\mathbb{F}$.

Folded Reed-Solomon code $\mathrm{FRS}^{(s)}_{\mathbb{F}}[k]$ (a code over $\mathbb{F}^s$) is the s-folded version of the above (for convenience assume $s \mid n$):

$$f(x) \mapsto \left(\begin{bmatrix} f(1) \\ f(\gamma) \\ \vdots \\ f(\gamma^{s-1}) \end{bmatrix}, \begin{bmatrix} f(\gamma^s) \\ f(\gamma^{s+1}) \\ \vdots \\ f(\gamma^{2s-1}) \end{bmatrix}, \ldots, \begin{bmatrix} f(\gamma^{n-s}) \\ f(\gamma^{n-s+1}) \\ \vdots \\ f(\gamma^{n-1}) \end{bmatrix}, \right).$$

The received word looks like following:

	1	γ	γ^s	$\dots$	γ^{n-s+1}
1	y_1	y_{s+1}			y_{n-s+2}
γ	y_2	y_{s+2}			$\vdots$
γ^2	y_3	y_{s+3}			$\vdots$
$\vdots$				$\ddots$	
γ^{s-1}	y_s	$\cdots$			y_n

$N = n/s$ many columns

If we think of Reed-Solomon decoding as interpolating a polynomial over a plane (which led to the $\sqrt{R}$ bound on agreement required), it might seem possible to decode with agreement fraction about $R^{s/(s+1)}$ by interpolating in $(s+1)$-dimensions.

Problem 21 *We want to find a polynomial* $Q(X, Y_1, Y_2, \dots, Y_s) \not\equiv 0 \in \mathbb{F}[X, Y_1, Y_2, \dots, Y_s]$ *such that* $Q(\gamma^{is}, y_{is+1}, y_{is+2}, \dots, y_{(i+1)s}) = 0$ *for* $0 \leq i < n/s$.

6 References

1. S. Ar, R. Lipton, R. Rubinfield and M. Sudan, "Reconstructing algebraic functions from mixed data," *SIAM Journal on Computing,* vol. 28, no. 2, pp. 488–511, 1999.

2. V. Guruswami, "List decoding of error-correcting codes," *Lecture Notes in Computer Science,* no. 3282, Springer, 2004.

3. V. Guruswami and A. Rudra, "Explicit Codes Achieving List Decoding Capacity: Error-Correction With Optimal Redundancy," *IEEE Transactions on Information Theory* 54(1): 135-150 (2008).

4. V. Guruswami and M. Sudan, "Improved decoding of Reed-Solomon and algebraic-geometric codes," *IEEE Transactions on Information Theory,* vol. 45, pp. 1757–1767, 1999.

5. R. Koetter and A. Vardy, "Algebraic soft-decision decoding of Reed-Solomon codes," *IEEE Transactions on Information Theory*, 49(11): 2809-2825 (2003)

6. M. Sudan, "Decoding Reed-Solomon codes beyond the error-correction bound," *Journal of Complexity,* vol. 13, no. 1, pp. 180–193, 1997.